Capítulo 5

Realización de controladores Digitales

Introducción

La acción de un controlador PID analógico ha sido exitosa en muchos sistemas de control y su funcionamiento es conocido, la función de transferencia de un PID analógico paralelo está dada por:

$$m(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(t).dt + T_d \frac{de(t)}{dt} \right)$$

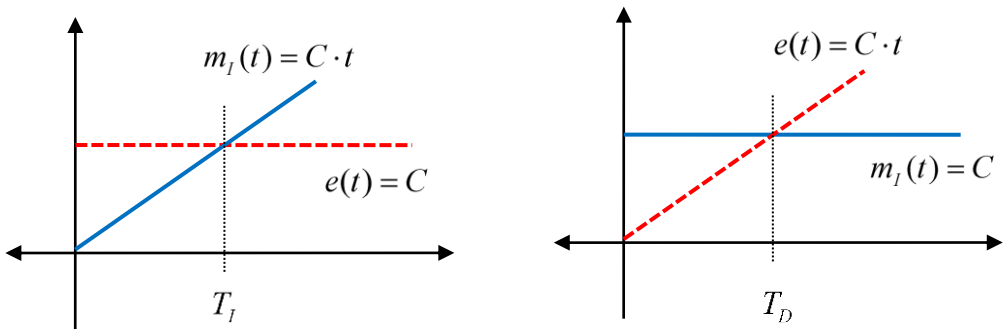
Donde $e(t)$ es la señal error o sea la entrada al controlador en cascada, y $m(t)$ es la variable manipulada, entrada al actuador. K es la ganancia proporcional,

Donde T_I es el denominado *tiempo de Reset* y es el tiempo que la salida el integrador tarda en alcanzar el valor del error ante un error constante

Sea $e(t) = C = cte$ luego $m_I(t) = C \cdot t / T_I$ y cuando $t = T_I \Rightarrow m_I(t) = C$

A T_D se le suele llamar *constante de tiempo derivativa* y es el tiempo que tarda la salida en alcanzar el error cuando este varía de forma lineal

Sea $e(t) = C \cdot t$ luego $m(t) = T_D C = cte$ y cuando $t = T_D \Rightarrow e(t) = C$



(a) (b)

Figura 1: (a) Tiempo integral o de Reset, (b) Constante de tiempo derivativa

Muchos de los procesos de tiempo continuo se pueden controlar mediante la versión digital de un PID o sus variantes (PI, PD), Que consisten en algoritmos, siempre que el periodo de muestreo sea lo suficientemente chico comparada con la dinámica del sistema, ya sea de lazo abierto o cerrado, la que sea más restrictiva
En este capítulo se desarrollan los métodos para implementar controladores digitales en forma de algoritmos, y también algunos de los distintos métodos para convertir controladores analógicos originalmente diseñados en tiempo continuo a su versión de tiempo discreto.

Controlador PID Digital

La función de transferencia discreta de un controlador PID genérico es como se vio

$$M(z)=\left[K_P+\frac{K_I}{1-z^{-1}}+K_D(1-z^{-1})\right]E(z)$$

donde $e(kT)\overset{Z}{\longleftrightarrow}E(z)$ es la señal error muestreada o sea la entrada al controlador en cascada, y $m(kT)\overset{Z}{\longleftrightarrow}M(z)$ es la salida del controlador que entra a un ZOH o equivalente

Implementación PID paralelo

Antitransformando la ecuación de definición y llevándola al dominio del tiempo resulta

$$m(kT)=K_P e(kT)+K_I\left[\sum_{n=0}^{k}e(nT)\right]+K_D\left(e(kT)-e[(k-1)T]\right)$$

Lo anterior requiere almacenar en memoria el error anterior

$$e\left[(k-1)T\right]$$

Y el error acumulado

$$ea(kT)=\sum_{n=0}^{k}e(nT)$$

El esquema de programacion se muestra en la figura 1

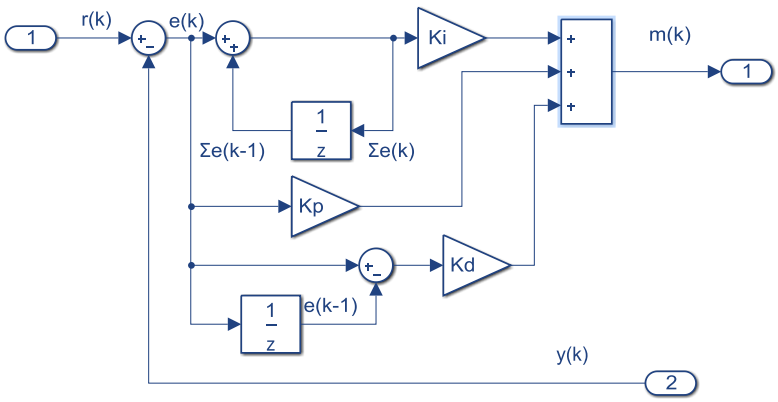


Figura 2: PID paralelo forma de posición

Se puede reescribir como

$$m(kT)=\left(K_P+K_D\right)e(kT)+K_I\left[\sum_{n=0}^{k}e(nT)\right]-K_D e[(k-1)T]$$

Lo que evita realizar una resta en el algoritmo de cálculo en tiempo real cambiándolo por una suma al inicio

Estas formas se denominan **algoritmo de posición**. También existen otras formas de implementación

Ejemplo 1: Forma paralelo a partir de polo cero

Sea el controlador del capítulo anterior en su forma factorizada

$$C(z)=1.5\frac{(z-0.2865)(z-0.68)}{z(z-1)}$$

Y se desea convertirlo a la forma paralelo

$$C(z)=K_P+\frac{K_I}{1-z^{-1}}+K_D(1-z^{-1})$$

La cual se puede escribir como

$$C(z) = K\frac{z^2 - bz + c}{z(z-1)}$$

con

$$K = K_P + K_D + K_I \quad b = \frac{K_P + 2K_D}{K} \quad c = \frac{K_D}{K}$$

$$C(z) = 1.5\frac{(z^2 - 0.96z + 0.195)}{z(z-1)}$$

Luego

$$K_D = 1.5 \cdot 0.195 = 0.3$$

$$b = \frac{K_P + 2K_D}{K} \Rightarrow K_P = Kb - 2K_D = 0.84$$

$$K = K_P + K_D + K_I \Rightarrow K_I = 1.5 - 0.84 - 0.3 = 0.36$$

En la Tabla 1 se muestra el código para implementar el PID de ambas formas y

Tabla 1: Código Matlab comparativa

```
CPID=0.84+0.36*tf([1 0],[1 -1],0.125)+0.3*tf([1 -1],[1 0],0.125)
Cd=zpk([0.2865 0.68],[0 1],[1.5],0.125)
tf(Cd),tf(CPID)
```

Implementación como filtro Digital o algoritmo de velocidad

Anteriormente se obtuvo que

$$\frac{M(z)}{E(z)} = K\frac{z^2 - bz + c}{z(z-1)}$$

Con

$$K_P + K_D + K_I = K \quad b = \frac{K_P + 2K_D}{K} = c_1 + c_2 \quad c = \frac{K_D}{K} = c_1 c_2$$

Entonces

$$\frac{M(z)}{E(z)} = K\frac{z^2 - bz + c}{z(z-1)}\frac{z^{-2}}{z^{-2}} = K\frac{1 - bz^{-1} + cz^{-2}}{1 - z^{-1}}$$

Pasando términos

$$(1 - z^{-1})M(z) = K(1 - bz^{-1} + cz^{-2})E(z)$$

Antitransformando

$$m(kT) = m[(k-1)T] + K\left\{e(kT) - \cdot b \cdot e[(k-1)T] + \cdot c \cdot e[(k-2)T]\right\}$$

El cual se denomina algoritmo de velocidad y requiere guardar el error pasado y el anterior, así como la acción anterior, un esquema se muestra en la figura 3, donde el switch representa una sentencia condicional *if* que suma o no el valor anterior, según la implementación pretendida

Es particularmente útil cuando el actuador el del tipo incremental, o sea tiene memoria como por ejemplo un motor paso a paso o algunas válvulas, ya que solo hay que enviarle la diferencia y se evita el termino de suma realimentada, lo que equivale a cambiar el switch en la figura, esto evita toda acumulación y no hay riesgo de overflow

$$\Delta m(kT) = Ke(kT) - K \cdot b \cdot e[(k-1)T] + K \cdot c \cdot e[(k-2)T]$$

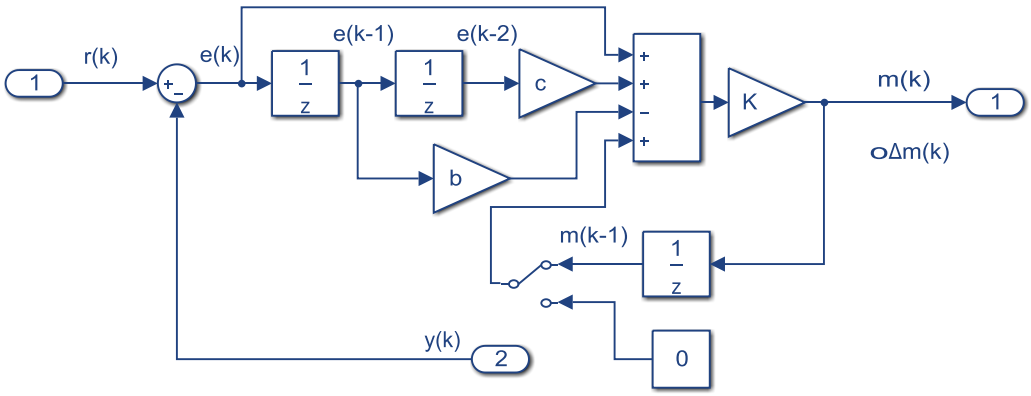


Figura 3: PID forma de velocidad, común o incremental

Forma directa II o variable de estado

Haciendo

$$\frac{M(z)}{E(z)} = K\frac{1 - bz^{-1} + cz^{-2}}{1 - z^{-1}}\frac{X(z)}{X(z)}$$

Donde $X(z)$ es una variable auxiliar, conocida como variable de estado

Luego se puede poner igualando numeradores y denominadores

$$M(z) = K\left(1 - bz^{-1} + cz^{-2}\right)X(z) = K\left(X(z) - bz^{-1}X(z) + cz^{-2}X(z)\right)$$

$$E(z) = \left(1 - z^{-1}\right)X(z) = X(z) - z^{-1}X(z) \Rightarrow X(z) = E(z) + z^{-1}X(z)$$

En el dominio del tiempo

$$M(kT) = K\left(x(kT) - bx[(k-1)T] + cx[(k-2)T]\right)$$

$$x(kT) = e(kT) + x[(k-1)T]$$

Lo cual se puede expresar en un diagrama de bloques como en la figura 4

La forma de variable de estado puede ser más eficiente para los cálculos en línea, pero se pierde el sentido de las acciones de control y son más difíciles de sintonizar manualmente.

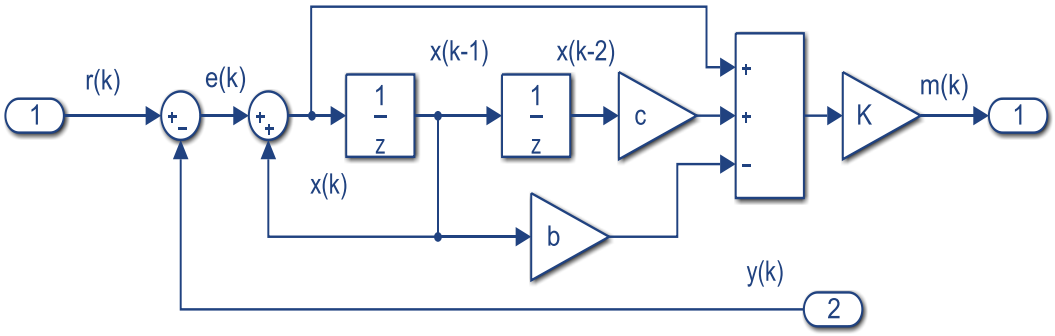


Figura 2: PID en Variable de estados

En resumen el PID estructuralmente es un caso especial de filtro digital de segundo orden y para su implementación aplica toda la teoría de dichos filtros, la cual no será abordada aquí

Ubicación de las acciones

También hay distintas estructuras de ubicación de las acciones tal como en tiempo continuo, aclarando que algunas solo valen para entradas de tipo escalón (regulación), así es usual derivar la salida en lugar del error para bajar la patada derivativa, básicamente hay 3 estructuras: PI+PD PI+D e I+PD

- Siempre la acción integral debe estar afectando al error, o sea en la cadena directa

Forma serie

PI+PD

Consiste en un PI en la rama directa y un PD en la realimentación con ganancia unitaria, se pone de esta manera para que uno de los ceros del PID (si no cancela un polo) no aparezca en la FT e lazo cerrado como se vio en el capítulo anterior, supóngase el PID en la forma polo cero y que ambos son reales, se puede expresar como

$$C(z) = K\frac{(z-c_D)(z-c_R)}{z(z-1)}\frac{1-c_R}{1-c_R}$$

Se podía descomponer en 2 terminos como se vio, uno en la rama de realimentacion que contenga unos de los ceros, en este caso se le llamara c_R , aquí hay 3 opciones para elegir el cero que va en la realimentacion

- Si no hay diseño por cancelación se pone en la realimentación el mas cercano a los polos dominantes
- Si hay cancelación debe ir el cero que no cancela el polo como se vio
- Si el cero es doble es indistinto

La condición es que tenga ganancia unitaria para no alterar la realimentación en régimen entonces el controlador en la realimentación $C_R(z)$ será un PD, y el restante $C_D(z)$ un PI en la cadena directa como se muestra en la figura

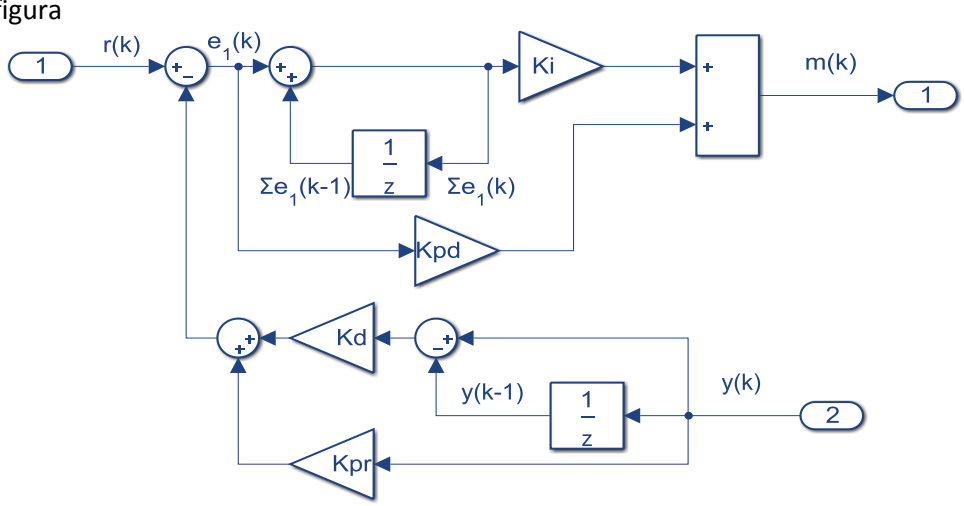


Figura PI+PD

$$C_R(z) = \frac{1}{1-c_R}\frac{(z-c_R)}{z} = K_{PR} + K_D\frac{z-1}{z}$$

$$C_D(z) = K(1-c_R)\frac{(z-c_D)}{(z-1)} = K_{PD} + K_I\frac{z}{z-1}$$

De aquí se pueden despejar las acciones de ambas ramas usando las formulas del PI y PD

- Observese que el polo en el origen del PD aparece a la salida como cero, es decir habrá un retardo puro

Formas paralelo

Por otro lado si se restringe la entrada a escalones espaciados en el tiempo (regulación) teniendo en cuenta que

$$e(kT) = r(kT) - y(kT)$$

La ley PID se podría reescribir como

$$m(kT) = K_P\left(r(kT) - y(kT)\right) + K_I\left[\sum_{n=0}^{k} e(nT)\right] + K_D\left(r(kT) - r[(k-1)T] - y(kT) + y[(k-1)T]\right)$$

PI+D

Considerando que la referencia es constante, excepto en el instante en que cambia

$$r(kT) - r[(k-1)T] = 0$$

La ley de control se puede implementar como

$$m(kT) = K_P e(kT) + K_I\left[\sum_{n=0}^{k} e(nT)\right] - K_D\left(y(kT) - y[(k-1)T]\right)$$

Un esquema se muestra en la figura

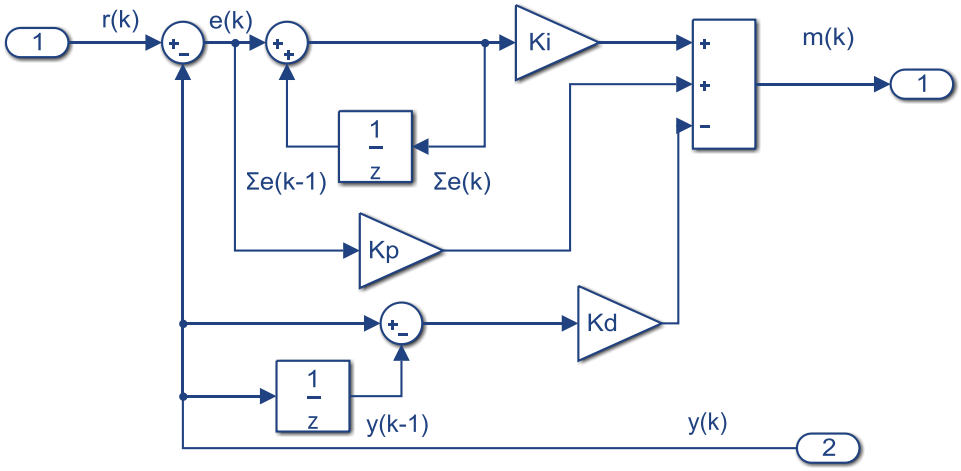


Figura 4: Forma PI+D

Aquí aparece solo la patada proporcional y hace más suaves las acciones de control
Pero es más lento que el PID tradicional

I+PD

Efectúa la acción integral sobre el error y la proporcional y derivativa sobre la salida, la que al no crecer bruscamente ante, por ejemplo un escalón, en la referencia hace más suaves las acciones de control y morigera las patadas derivativas y proporcionales

Si se desprecia el término

$$K_P r(kT)$$

Queda el algoritmo I+PD

$$m(kT) = -K_P y(kT) + K_I\left[\sum_{n=0}^{k} e(nT)\right] - K_D\left[y(kT) - [y(k-1)T]\right]$$

Lo cual hace el control más lento, pero a su vez la acción de control se vuelve más suave y se evitan tanto la patada derivativa como la proporcional

Su implementación se muestra en la figura y es mas lento que el anterior

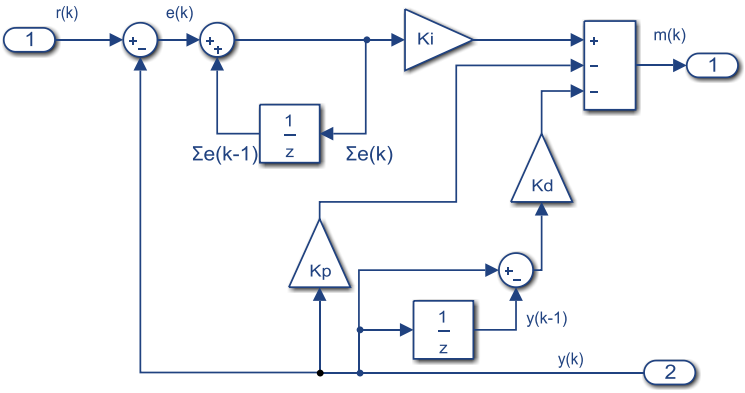


Figura 3: Forma I+PD

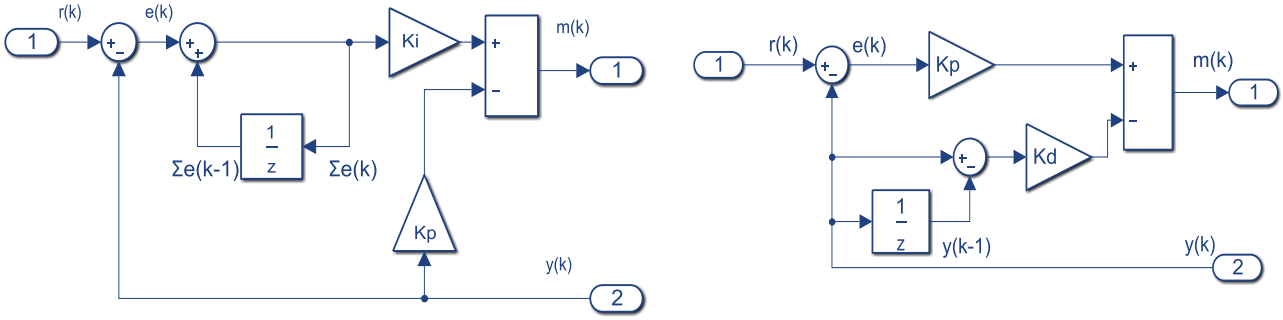
Variantes PI y PD

Para el caso de un PD se puede usar el llamado P+D que calcula la parte proporcional del error y la derivativa de la salida, sería un PI+D con ganancia integral nula

$$m(kT) = K_P e(kT) - K_D\left[y(kT) - [y(k-1)T]\right]$$

Para un PI se puede usar el I+P que toma la integral del error y la salida cambiada de signo, sería un I+PD con ganancia derivativa nula

$$m(kT) = -K_P y(kT) + K_I\left[\sum_{n=0}^{k} e(nT)\right]$$



(a) (b)

Figura Formas (a) I+P, (b) P+D

Implementación Controladores de Adelanto y atraso

El controlador de adelanto o de atraso es

$$C(z) = K\frac{z-a}{z-b}$$

Implementacion directa

Se puede reescribir como

$$C(z) = K\frac{z-a}{z-b}\frac{z^{-1}}{z^{-1}} = K\frac{1-az^{-1}}{1-bz^{-1}} = \frac{M(z)}{E(z)}$$

$$M(z)(1-bz^{-1}) = K(1-az^{-1})$$

Luego

$$m[kT] = b \cdot m[(k-1)T] + K\left(e[kT] - a \cdot e[(k-1)T]\right)$$

Es decir hay que guardar el error y la salida anteriores como se muestra en la figura

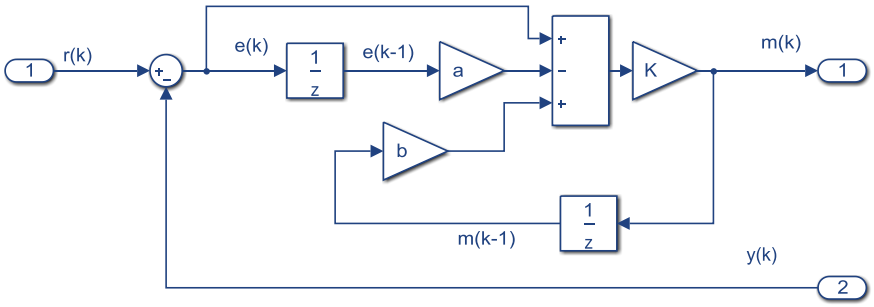


Figura 5: Controlador de Adelanto / Atraso forma normal

Implementación en VE

Partiendo de la ecuación anterior y procediendo como en el PID

$$\frac{M(z)}{E(z)} = K\frac{1-az^{-1}}{1-bz^{-1}}\frac{X(z)}{X(z)}$$

Luego igualando

$$M(z) = K(1-az^{-1})X(z)$$
$$E(z) = (1-bz^{-1})X(z)$$

Pasando al dominio del tiempo

$$m[kT] = K\left(x[kT] - a \cdot x[(k-1)T]\right)$$
$$x[kT] = e[kT] + b \cdot x[(k-1)T]$$

En la figura se muestra el diagrama donde puede notarse que hace falta solo un elemento de memoria

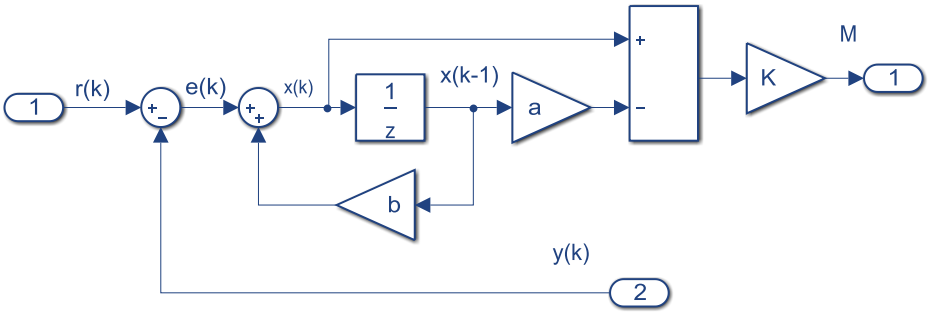


Figura 6: Controlador de Adelanto / Atraso en VE

Digitalización de controladores continuos

Muchas veces se diseña el control en tiempo continuo y luego se digitaliza el mismo, para ello es necesario que el periodo de muestreo sea pequeño, es decir la frecuencia alta para poder despreciar el efecto del Z0H

Controlador PID

Para obtener la función de transferencia discreta del controlador hay que discretizar su ecuación integro diferencial, para lo cual pueden usarse varios métodos numéricos.

Acción derivativa

Generalmente la derivada se aproxima mediante la diferencia regresiva entre dos puntos, es decir por la secante como se observa en la figura 7

$$\left.\frac{de(t)}{dt}\right|_{t=KT} \simeq \frac{[e(kT)-e((k-1)T)]}{T}$$

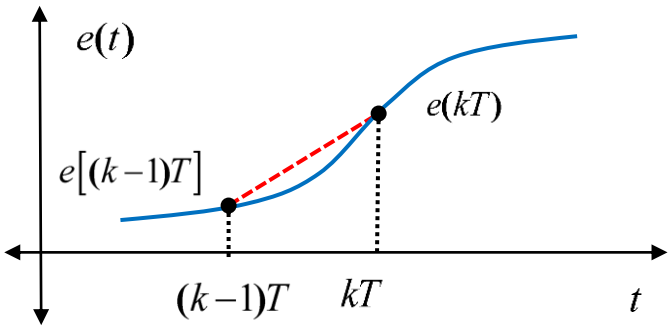
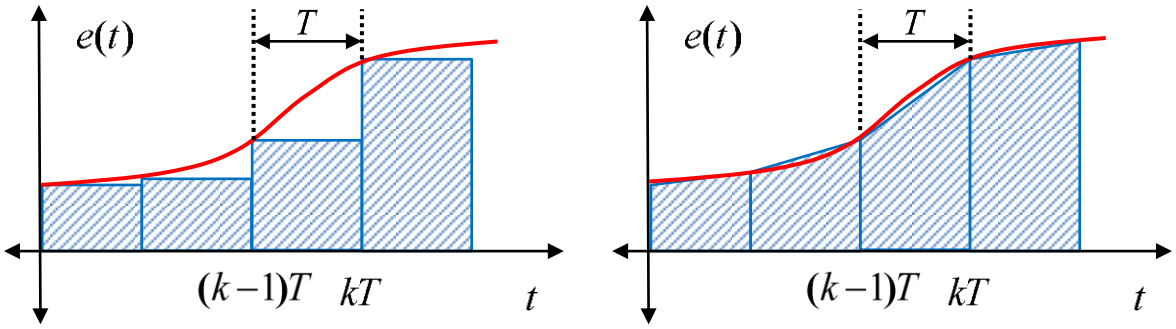


Figura 7: Derivada por diferencia regresiva

Acción Integral

Por método Euler

Si se aproxima la integral por medio del método rectangular de Euler como en la figura 8 (a)



(a) (b)

Figura 8: Aproximación integral (a) Rectangular o Euler (b) Trapezoidal o Tustin

Se obtiene:

$$m(kT) = K\left\{e(kT) + \frac{T}{T_i}\sum_{n=0}^{k} e(nT) + \frac{T_d}{T}[e(kT) - e((k-1)T)]\right\}$$

$$Z\{\sum_{n=0}^{k} x(n)\} = \frac{1}{1-z^{-1}}X(z);$$

Al transformar por Z

$$Z\left\{\sum_{n=1}^{k} e(nT)\right\} = \frac{1}{1-z^{-1}}E(z)$$

Entonces la función de transferencia del PID digital es:

$$M(z) = K[1 + \frac{T}{T_i}\frac{1}{1-z^{-1}} + \frac{T_d}{T}(1-z^{-1})].E(z)$$

Esta última ecuación puede escribirse como:

$$M(z) = K[1 + \frac{T}{T_i}\frac{1}{1-z^{-1}} + \frac{T_d}{T}(1-z^{-1})].E(z)$$
$$= K[1 + \frac{T}{T_i}\frac{1}{1-z^{-1}} + \frac{T_d}{T}(1-z^{-1})].E(z)$$

Donde

Ganancia proporcional

$K_P = K$

Ganancia integral

$$K_I = \frac{KT}{T_i}$$

Ganancia derivativa

$$K_D = \frac{KT_d}{T}$$

Entonces:

$$M(z) = [K_P + \frac{K_I}{1-z^{-1}} + K_D(1-z^{-1})].E(z)$$

Aproximación Integral Trapezoidal

Si se usa la regla trapezoidal para la integral como en la figura 8 (b)

$$\int_0^{kT} e(t)dt \simeq T\sum_{n=0}^{k}\frac{e[(n-1)T] + e(nT)}{2}$$

Resulta para la parte integral por las propiedades de la transformada Z

$$Z\{\sum_{n=1}^{k}\frac{e((n-1)T) + e(nT)}{2}\} = \frac{1+z^{-1}}{2(1-z^{-1})}E(z)$$

Haciendo la división se puede poner

$$\frac{z^{-1}+1}{-z^{-1}+1} = -1 + \frac{2}{1-z^{-1}}$$

Y la parte integral queda como

$$M_I(z) = -\frac{T}{2T_I}E(z) + \frac{T}{T_I}\frac{1}{1-z^{-1}}E(z)$$

Aparece una componente proporcional de ganancia negativa que se suma a la acción proporcional quedando esta

$$K_P = K - \frac{KT}{2T_i} = K - \frac{K_I}{2}$$

Adelanto y Atraso

La función de transferencia en tiempo continuo es para ambos

$$\frac{M(s)}{E(s)} = K_A \frac{(s-c)}{(s-p)}$$
$$c, p < 0$$

Se puede reescribir como

$$(s-p)M(s) = K_A(s-c)E(s)$$

En el tiempo

$$\frac{dm(t)}{dt} - p \cdot m(t) = K_A\left(\frac{de(t)}{dt} - c \cdot e(t)\right)$$

Aplicando la primera diferencia para aproximar la derivada (Euler)

$$M(z)\frac{(1-z^{-1})}{T} - pM(z) = K_A\frac{(1-z^{-1})}{T}E(z) - K_A \cdot c \cdot E(z)$$

Multiplicando por T y z ambos miembros

$$M(z)(z-1) - pTzM(z) = K_A(z-1)E(z) - K_A \cdot cTz \cdot E(z)$$

Y operando

$$M(z)\big(z(1-pT)-1\big) = K_A\big(z(1-cT)-1\big)E(z)$$

Haciendo

$$M(z)(1-pT)\left(z - \frac{1}{1-pT}\right) = K_A(1-cT)\left(z - \frac{1}{1-cT}\right)E(z)$$

$$\frac{M(z)}{E(z)} = K_A\frac{1-cT}{1-pT}\frac{z - \frac{1}{1-cT}}{z - \frac{1}{1-pT}}$$

Igualando con la expresión general

$$\frac{M(z)}{E(z)} = K\frac{z-a}{z-b}$$

$$a = \frac{1}{1-cT} \quad b = \frac{1}{1-pT}$$

Es decir la ubicación del cero a y del polo b dependen de T al igual que la ganancia resultante

Y la ganancia es

$$K = K_A\frac{1-cT}{1-pT} = K_A\frac{b}{a}$$

Es estable al ser p negativo ya que

$$|1-pT| > 1 \Rightarrow |b| < 1$$

Y de fase mínima ya que al ser c negativo

$$|1-cT|>1 \Rightarrow |a|<1 \Rightarrow$$

Para que el algoritmo funcione similar al diseño de tiempo continuo, es decir que el efecto del Z0H sea despreciable debe darse que la tasa de muestreo sea alta comparada con las constantes de tiempo del polo y del cero

$$T << \frac{1}{p}, T << \frac{1}{c}$$

Pero si el tiempo de muestreo es demasiado pequeño

$$cT \to 0 \Rightarrow a \to 1,\ pT \to 0 \Rightarrow b \to 1$$

Tanto el polo como el cero tienden a juntarse en el 1

También se puede hacer mediante la transformación bilineal que mapea el semiplano izquierdo en el círculo unitario

$$z = \frac{1+\frac{T}{2}s}{1-\frac{T}{2}s}$$

$$s = \frac{2}{T}\frac{z-1}{z+1}$$

Y se encuentra disponible en Matlab como
[Cd,Pd,Kd]=bilinear(Cc,Pc,Kc,1/T)

Ejemplo 1
Sea el controlador de adelanto de fase en tiempo continuo

$$C_{AD}(s) = 2\frac{s+1}{s+3}$$

Se desea digitalizarlo con $T = 0.1$ para obtener su versión digital

$$C_{AD}(z) = K\frac{z-a}{z-b}$$

Usando las formulas obtenidas

$$a = \frac{1}{1-cT} = \frac{1}{1+T} = 0.9091$$

$$b = \frac{1}{1-pT} = \frac{1}{1+3T} = 0.7692$$

$$K = K_A\frac{b}{a} = 3\frac{b}{a} = 1.6923$$

Luego

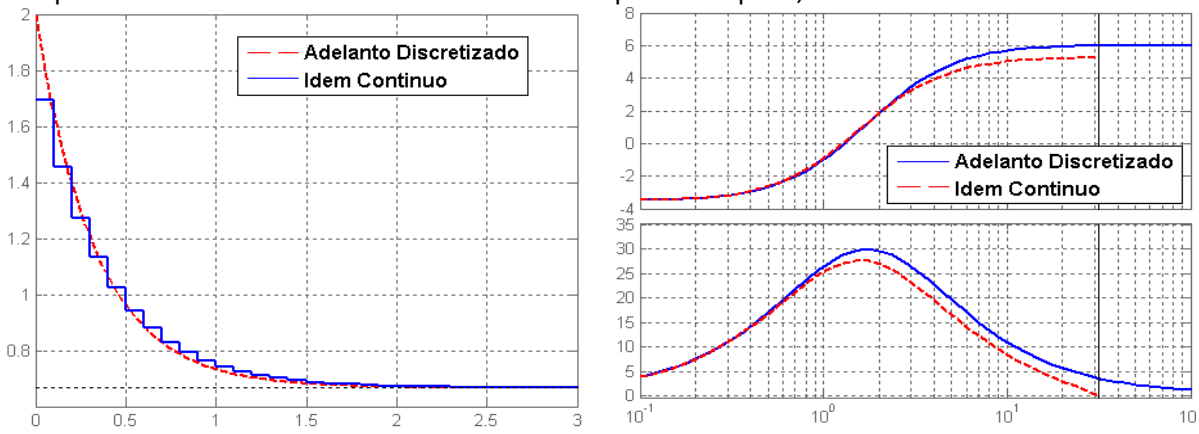
$$C_{AD}(z) = 1.6923\frac{z-0.9091}{z-0.7692}$$

Con la transformacion bilineal se tiene

$$C_{AD}(z) = 1.8261\frac{z-0.9048}{z-0.7391}$$

Que es muy similar, en la figura (a) se muestra la respuesta al escalón de ambos y en la (b) el Bode donde pude verse que en una frecuencia se produce un adelanto de fase maximo cercano a 30º el continuo y 25º el

discreto, lo que da el nombre al controlador, también se aprecia que a bajas frecuencias coinciden pero a medida que se acercan a la frecuencia de muestreo se apartan un poco, es decir la relación no es lineal



(a) (b)

Figura: Controlador de adelanto continuo y discretizado, (a) Respuesta al escalon, (b) Bode

Tabla : Codigo controlador de adelanto

c=-1,p=-3,ka=2 *T=0.1* *G=zpk([c],[p],ka)* *a=1/(1-c*T), b=1/(1-p*T), K=ka*b/a*	*Gd=zpk(a,b,K,T)* *[Cd,Pd,Kd]=bilinear(c,p,ka,T) % entrega ceros, polos y ganancia* *figure(1), step(G,Gd)* *figure(2), bode(G,Gd)*

Efecto del ZOH

Se demostró que el ZOH puede ser aproximado mediante una función racional de primer orden con un polo en

$$p_{ZOH} = -\frac{2}{T} \Rightarrow \tau_{ZOH} = \frac{T}{2}$$

Si la ubicación de ese polo que conforme se aumenta el periodo de muestreo se aleja hacia la izquierda se encuentra lejana a los polos dominantes del sistema puede no tenerse en cuenta
Caso contrario habrá que rediseñar el control analógico incluyéndolo

Ejemplo PID

Ejemplo 2: Digitalización controlador PID de cancelación

Supóngase que se diseñó por LR un controlador de cancelación para la planta

$$Gp(s) = \frac{3}{(s+1)(s+2)} = \frac{3}{s^2+3s+2}$$

Con especificaciones Sobrepaso del 5% lo que implica que

$$\xi = 0.707$$

Y tiempo de establecimiento 1.6 s*eg*. por lo que

$$\sigma = -\frac{4}{ts} = 2.5 = \omega_d$$

Y se obtuvo un controlador PID cuya FT es

$$C(s) = \frac{(s+4.2)(s+1)}{s}$$

El cual ubica los polos en

$$p_{1,2} = -2.5 \pm j2.5 = \sigma \pm j\omega_d$$

Tambien existe un cero en

$$c = -4.2$$

El cual no se puede considerar **dominado** por los polos por lo que el sobrepaso será mayor al esperado

Se digitalizara el PID por método de derivación por diferencia regresiva e integración rectangular (Euler) para que haya al menos $M > 10$ muestras por ciclo de la salida
El controlador hay que parametrizarlo haciendo

$$C(s) = \frac{(s+4.2)(s+1)}{s} = \frac{s^2 + 5.2s + 4.2}{s}$$

$$C(s) = K + \frac{K / T_i}{s} + KT_d s = \frac{KT_d s^2 + Ks + K / T_i}{s}$$

Igualando resultan para el continuo

$$K^C{}_P = K = 5.2$$

$$K^C{}_D = KT_d = 1$$

$$K^C{}_I = K / T_i = 4.1$$

Como la parte imaginaria de los polos es la frecuencia natural amortiguada

$$\omega_d = 2.5 rad / seg$$

Luego el periodo de dicha frecuencia será

$$T_d = \frac{2\pi}{\omega_d} = 2.51$$

Entonces el periodo de muestreo deberá ser de

$$T = \frac{2.5}{M}$$

En la tabla 1 se muestran los valores de T para distintos M

Tabla : Muestras por ciclo y periodo de muestreo para $\omega_a = 2.5 rad / seg$

M	T
10	0.25
20	0.125
40	0.0625

Para estos periodos de muestreo la constante de tiempo del ZOH es despreciable siendo

$$\tau_{ZOH} = \frac{T}{2} < 0.125$$

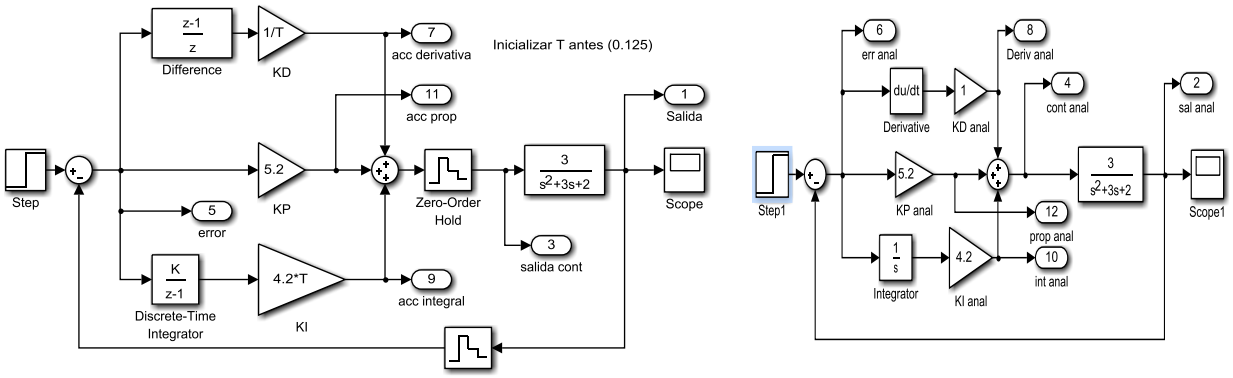
Luego empleando las formulas del método Euler

$$K = K_P \Rightarrow K_P = 5.2 \quad K_I = \frac{KT}{T_i} = 4.2T \qquad K_D = \frac{KT_d}{T} \Rightarrow K_D = \frac{1}{T}$$

Aquí puede verse que como al disminuir T aumenta la ganancia derivativa y baja la integral
Si se usa un tiempo de muestreo chico, por ejemplo para 20 muestras por ciclo

$$T = 0.125 \Rightarrow K_D = 8, K_I = 0.525$$

Hay mucho predominio de la parte derivativa sobre la integral y es similar a la proporcional



(a) (b)

Figura : Esquema de simulación paralelo (a) Digital (b) Analogico

En la figura (a) se muestran las salidas simuladas con y en la (b) la acciones de control para $T = 0.125$
Se observa que el controlador digital tiene prácticamente el mismo sobrepaso e incluso es más rápido, ello a expensas de una señal de control fuerte al principio, esto es principalmente debido a la acción derivativa que se incrementa inversamente al periodo de muestreo

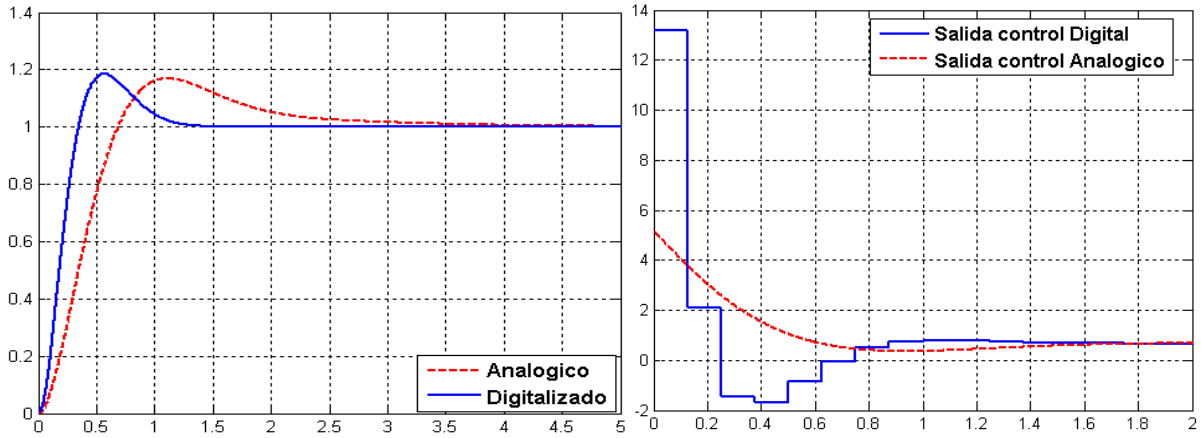


Figura_: (a) Salidas (b) Acciones de control

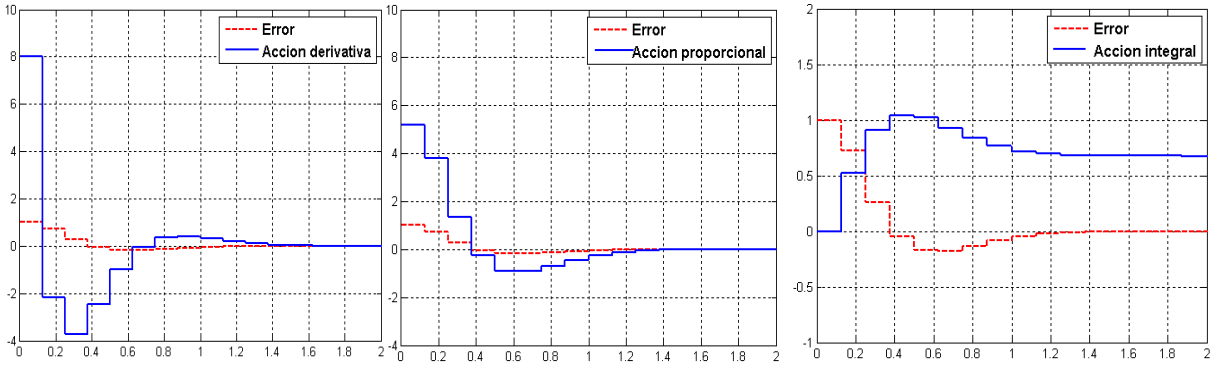
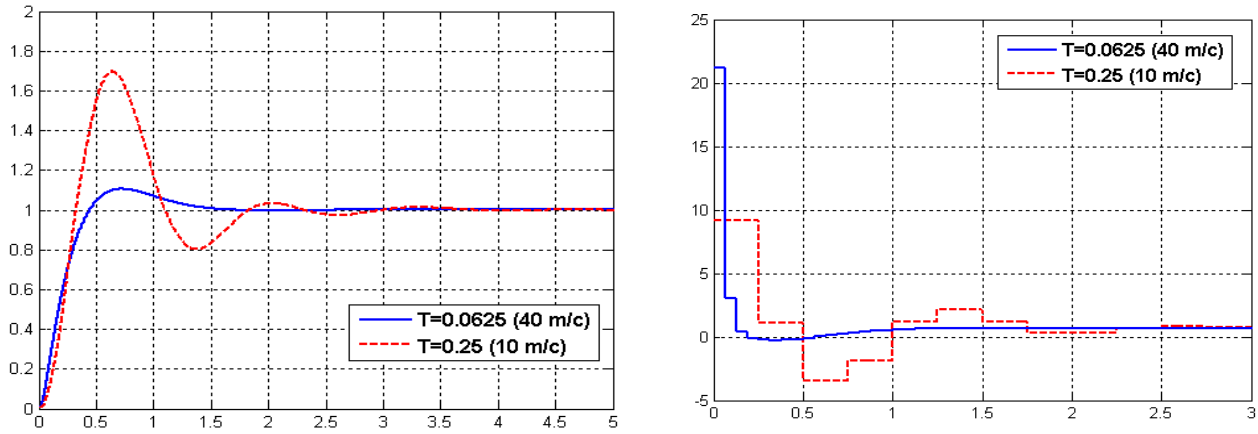


Figura_: (a) Salidas (b) Acciones de control con $T = 0.125$

Influencia del Periodo de muestreo

La gran acción derivativa se puede atenuar subiendo el tiempo de muestreo, por ejemplo $T = 0.25$ pero no necesariamente mejora la calidad pues también baja la acción integral y las muestras por ciclo (10), también se pude hacer $T = 0.0625$ lo que significan 40 muestras por ciclo
En la figura (a) se muestran las salidas simuladas con ambos periodos donde se nota que hay un mayor sobrepaso y mayor frecuencia de oscilación amortiguada para $T = 0.25$ mientras que si se baja a

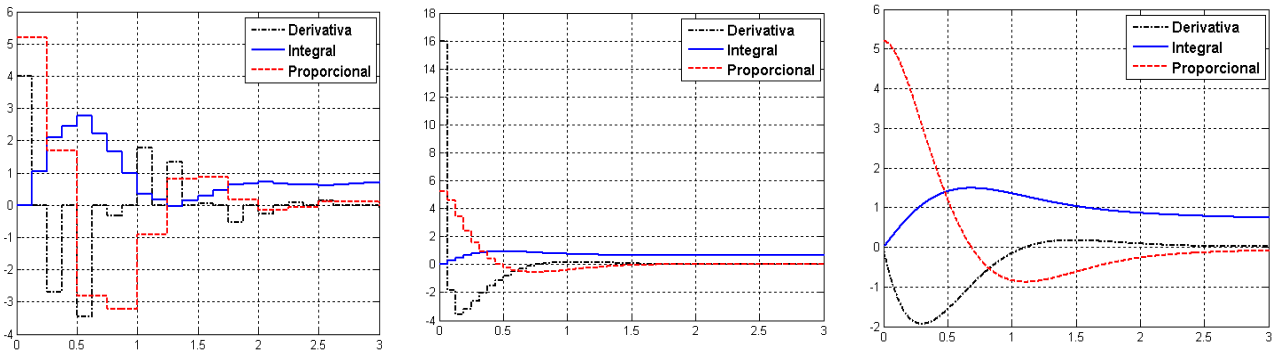
$T = 0.0625$ mejora sustancialmente el desempeño, en la (b) Las acciones de control que explican el efecto, es decir para $T = 0.0625$ hay una gran acción derivativa (patada derivativa) pero solo al inicio, dando una respuesta menos oscilatoria ya que el pico derivativo inicial es absorbido por la característica pasa bajos del proceso



(a) (b)

Figura : Variación con el periodo de muestreo, (a) Salidas (b) Acciones control

En la figura (a) y (b) se muestran las acciones para ambos tiempos de muestreo y en la (c) la analógica, observándose que en el caso de bajo tiempo de muestreo las acciones son en cierta forma parecidas a las de tiempo continuo



(a) (b) (c)

Figura : Acciones de control (a) $T = 0.25$ (b) $T = 0.0625$, (c) Analogicas

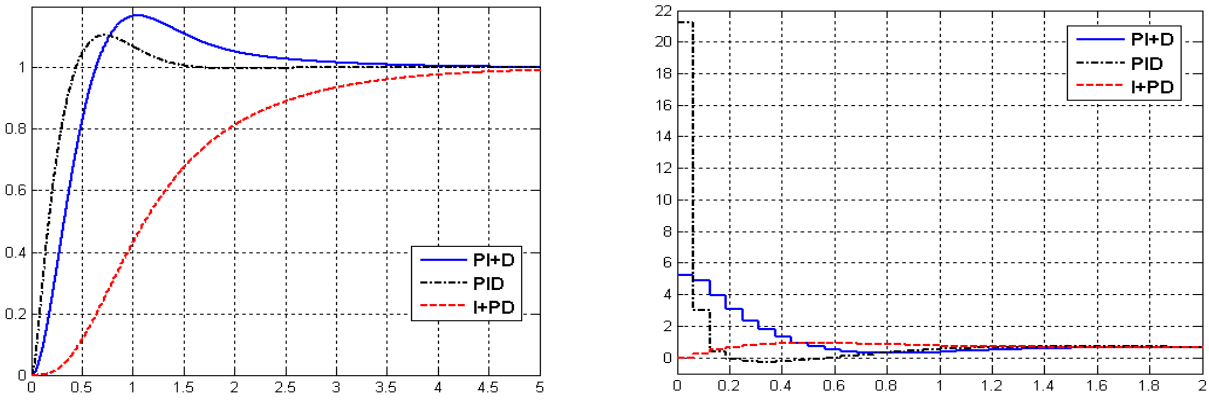
La elección de periodo es entonces una relación de compromiso

- Si se desea bajo sobrepaso se escoge un periodo $T = 0.0625$ (o cercano a el) que corresponde a $m = 40 \frac{muestras}{ciclo}$
- Si la amplitud inicial de la señal de control no es admisible o bien el conversor no soporta dicha tasa se escoge un periodo $T = 0.125$ (o cercano a el) que corresponde a $m = 20 \frac{muestras}{ciclo}$
- Si se desea un balance se puede elegir un periodo intermedio

Reordenamiento de las acciones

Si se desea evitar una acción derivativa muy alta se puede hacer tal cual se vio anteriormente un reordenamiento de las acciones de control

Se pueden usar las variante PI+D y I+PD resultando las salidas y la acciones de control mostradas en la figura para 40 muestras por ciclo, o sea $T = 0.0625$



(a) (b)

Figura : Comparativa PID versus I+PD y PI+D, (a) Salida, (b) Acciones de contrrol

- Si se usa la forma PI+D hay un mayor sobrepaso y menor tiempo de respuesta pero la acccion de control es mas suave
- Si se usa la forma I+PD la señal de control es demasiado leve deteriorando demasiado el tiempo de respuesta

Forma PI+PD

Para bajar el Sobrepasamiento se puede eliminar uno de los ceros del controlador para que no aparezca a lazo cerrado, en este caso el que no cancela ningún polo pasándolo a la realimentación y esto a su vez morigera la patada derivativa

El PID continuo ya visto

$$C(s) = \frac{(s+4.2)(s+1)}{s}$$

Se puede poner en su forma PI+PD como

$$C(s) = 4.2\frac{(s+1)}{s}\frac{(s+4.2)}{4.2}$$

Siendo entonces el controlador PI el que contiene el cero que cancela el polo

$$C_{PI}(s) = 4.2\frac{(s+1)}{s} = 4.2\left(1+\frac{1}{s}\right) = K\left(1+\frac{1}{sT_I}\right)$$

$$K = 4.2, T_I = 1$$

Y el PD en la rama realimentada es

$$C_{PD}(s) = \frac{(s+4.2)}{4.2} = (0.2381s+1) = K_1(T_D s+1)$$

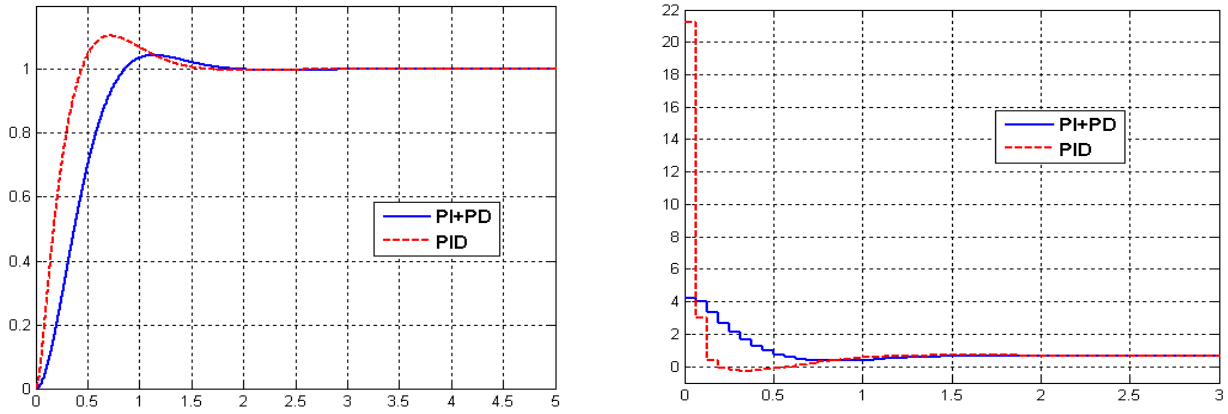
$$K_1 = 1, T_D = 0.238$$

Al digitalizar por Euler con periodo T

$$C_{PI}(z) = 4.2 + 4.2T\frac{z}{z-1}$$

$$C_{PD}(z) = 1 + \frac{0.238}{T}\frac{z-1}{z}$$

En la figura se muestran las salidas y las acciones de control comparadas para 40 muestras por ciclo, o sea $T = 0.0625$



(a) (b)

Figura : Comparativa PID versus PI+PD, (a) Salida, (b) Acciones de control

- Se ha bajado el sobrepaso y la patada derivativa e expensas de un tiempo de establecimiento levemente mayor

Observaciones sobre el muestreo

En general es conveniente muestrear a una tasa alta pero no demasiada pues

- Los cálculos deben ser efectuados en una fracción despreciable del tiempo de muestreo, lo que para sistemas rápidos exige procesadores de altas prestaciones y encarece los costos
- En el caso de que se entregue la salida un instante de muestreo posterior al sensado, ello implica un retardo que puede ser desestabilizante
- A veces los conversores o los sensores imponen limitaciones técnicas sobre el muestreo, y si se les exige velocidad y resolución pueden ser costosos
- En general debe ser pequeño para que la aproximación funcione, pero si es demasiado chico baja la acción integral y aumenta la derivativa produciendo efectos indeseables

A modo de conclusión podría decirse que hay que muestrear lo suficientemente rápido pero no demasiado, debe ser acorde a la dinámica del proceso y/o a las especificaciones de respuesta y también se deben tener en cuenta las limitaciones tecnológicas y obtener una buena relación costo vs beneficio.

Modos de ejecutar el código

El código se ejecuta llamando a una función o subrutina, en la tabla se lista un pseudo código sencillo

Tabla : Pseudo código simple función PID de posición

% subrutina PID	*Aprop=Kp*/Error_act % Proporcional*
Read Ref % leer del ADC (o de memoria)	*Error_acum=Error_acum+Error_act % acumulacion*
Read Sal % leer del ADC	*Aint=Ki*Error_acum % Integral*
Read Error_acum	*Ader=Kd* (Error_act-Error_ant) % Derivativa*
% solo si las constantes están en memoria	*Acc_cont=Aprop+Aint+Ader % Total*
Read Kp	*Write Acc_cont % escribir en el DAC (o memoria)*
Read Kc	*Error_ant=Error_act*
Read Ki	*Save Error_ant % guardar en memoria*
% calculo	*Save Error_acum % guardar en memoria*
Error_act=Ref-Sal % Error	*Return*

Por tiempo de espera

El código se puede ejecutar por tiempo de espera que es la manera más sencilla, pero no la mejor, un ejemplo seria el mostrado en la tabla, una desventaja es que solo corre el código basico y puede estar desincronizado con la conversión A/D

Tabla 2: ejecución por espera

```
Inicializar valores parámetros y memorias
parada=falso
Fijar reloj en T
Errores anteriores y acumulados=0
        While parada=falso
                Ejecución código algoritmo control
                Esperar hasta T
        End
 Ejecución código parada segura
```

Por Interrupción de Hardware

Una mejor manera que permite hacer otras tareas es ejecutarlo por interrupciones de hardware para asegurar su ejecución en cada instante de tiempo kT y así quedar sincronizado con la conversión A/D
Una interrupción es una señal recibida por el procesador, para indicarle que debe interrumpir el curso de ejecución actual y pasar a ejecutar código específico para tratar esta situación, la secuencia es

1. Terminar la ejecución de la instrucción máquina en curso.
2. Salvar el estado del procesador (valores de registros y flags) y el valor del contador de programa, IP, en la pila, de manera que en la CPU, al terminar el proceso de interrupción, pueda seguir ejecutando el programa a partir de la última instrucción.
3. La CPU salta a la dirección donde está almacenada la rutina de servicio de interrupción (Interrupt Service Routine, o abreviado ISR) y ejecuta esa rutina que tiene como objetivo atender al dispositivo que generó la interrupción, en este caso el conversor A/D.
4. Una vez que la rutina de la interrupción termina, el procesador restaura el estado que había guardado en la pila en el paso 2 y retorna al programa que se estaba usando anteriormente.

Hardware

Un temporizador previamente inicializado activa en cada periodo de muestreo el conversor A/D, el que cuando tiene listo el dato digital, dependiendo del tiempo de conversión genera una interrupción por hardware, normalmente cambiando el estado lógico o generando un flanco en un pin especial del procesador, lo que que obliga al mismo a detener el código que estaba ejecutando para pasar a ejecutar el código del controlador, una vez ejecutado habilita de nuevo el pin y continua con el programa.

Software

Se muestra en la tabla 3 el pseudo código, las líneas representan el momento de la interrupción la cual es asíncrona, es decir puede acontecer en cualquier momento dentro del bucle infinito, en este ejemplo seria en la instrucción i-esima
Como el algoritmo dura una pequeña fracción del tiempo de muestreo, el resto del tiempo el procesador puede dedicarse, por ejemplo, a interactuar con un sistema tipo SCADA vía algún bus de campo y también otras acciones sencillas como manejo de pantalla y teclado
Lo que siempre debe asegurarse es que el código del algoritmo de control *se ejecute completo y exactamente en cada tiempo de muestreo,* es decir tiene que tener la más alta prioridad, no puede ser interrumpida por una acción de menor nivel
Si se trata de una PC debe estar corriendo un sistema operativo de *Tiempo Real* como los que suministran Matlab o LabView, por ejemplo.

Tabla 3: Ejecución por Interrupción

```
Inicializar  parámetros, reloj de conversion y memorias
parada=falso
Valores anteriores=0
            While parada=falso
                  Instrucción 1
                  ……..
                  Instrucción i
_____ Interrupción muestreo A/D (prioridad 0)
                  Terminar instrucción i y guardar estado
                  Deshabilitar interrupciones
                     Código algoritmo control
                  Recuperar estado
                   Habilitar interrupciones
_____ Fin interrupción
                  Instrucción i+1
                  ……..
          End
 Ejecución código parada segura
```