



APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

## ÍNDICE

1.- OBJETO DEL PROYECTO.....	1
2.- INTRODUCCIÓN.....	1
3.- ESTUDIO DE NECESIDADES, LIMITACIONES Y CONDICIONANTES.....	3
3.1 Descripción de la aplicación.....	3
3.1.1 Principal.....	3
3.1.2 Especificaciones.....	11
3.1.3 Cálculo de PID.....	13
3.1.3.1 Acción proporcional.....	14
3.1.3.2 Acción integral.....	14
3.1.3.3 Acción derivada.....	15
3.1.3.4 PID ISA.....	15
3.1.3.5 Criterios para la obtención del regulador PID.....	17
3.2 Planteamiento de necesidades.....	19
4.- PLANTEAMIENTO DE SOLUCIONES ALTERNATIVAS.....	20
4.1 Obtención del modelo matemático.....	20



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

4.1.1 Modelado Teórico.....20

4.1.2 Identificación experimental.....21

    4.1.2.1 Identificación ante respuesta escalón.....21

    4.1.2.2 Identificación por mínimos cuadrados.....22

4.2 Elección del modelo.....22

4.3 Tipo de Control.....25

    4.3.1 Continuo.....26

    4.3.2 Discreto.....26

5.- JUSTIFICACIÓN DE LA SOLUCIÓN ADOPTADA.....28

    5.1 Control analógico.....28

    5.2 Control digital.....33

6.- BIBLIOGRAFÍA.....39



## APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

### 1.- OBJETO DEL PROYECTO

El objetivo de este proyecto es el desarrollo de una aplicación informática, implementada en el entorno de Matlab para el cálculo y obtención de reguladores PID continuos y discretos.

Esta aplicación pretende facilitar la complejidad que puede presentar la obtención de un regulador PID, para el control de un sistema que responda a un comportamiento concreto esperado.

Esta aplicación nos permitirá obtener la función de transferencia del controlador (Proporcional - Integral - Derivativo), para su posterior simulación, y comparación entre el sistema compensado y el sistema sin compensar.

Mediante la interfaz gráfica de Matlab, se pueden obtener gráficas y simulaciones para el análisis del comportamiento dinámico de los sistemas, ajustar a un comportamiento deseado, para así después poder realizar el montaje del circuito según los parámetros obtenidos.

### 2.- INTRODUCCIÓN

El controlador PID se ha desarrollado a lo largo de un período de tiempo que se extiende al menos 250 años. Es útil tener alguna perspectiva de este desarrollo en orden a comprender muchos de los temas. La tecnología utilizada para implementarlos, naturalmente, ha cambiado significativamente a lo largo de los años.

Los primeros controladores fueron dispositivos mecánicos (reguladores centrífugos) empleados para controlar molinos de viento y máquinas de vapor. La medida de la velocidad angular se combinaba con la actuación de las válvulas.

Comenzando en los años 50, pronto estuvieron disponibles las versiones electrónicas de los controladores PID. Las acciones de control se obtuvieron mediante un simple computador analógico basado en amplificadores operacionales. La transmisión de señal se normalizó también en el rango de 4–20 mA. Representar el cero mediante una corriente distinta de cero era útil para realizar diagnósticos.

Prácticamente todos los controladores PID fabricados hoy día se basan en microprocesadores. El controlador PID se ha convertido en un banco de pruebas para muchas nuevas ideas en control.

Un cambio importante ocurrió en conexión con el desarrollo del control de procesos industriales. Las funciones de medida, control, y actuación fueron entonces separadas y se construyeron dispositivos especiales que efectuaban las acciones de control.



## APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

El controlador PID es una implementación simple de la idea de realimentación. La realimentación puede reducir los efectos de las perturbaciones, puede hacer que un sistema sea insensible a las variaciones del proceso y puede lograr que un sistema siga fielmente a sus señales de entrada.

El controlador PID corrige el error entre un valor medido y el valor que se quiere obtener calculándolo y luego sacando una acción correctora. El algoritmo de cálculo del control PID se da en tres parámetros distintos: el proporcional, integral y derivativo.

Tiene la capacidad de eliminar errores en estado estacionario mediante la acción integral, y puede anticipar el futuro con la acción derivativa. Los controladores PID, o incluso los controladores PI, son suficientes para muchos problemas de control, particularmente cuando las dinámicas del proceso son benignas y los requisitos de comportamiento no son complejos.

Puede ser llamado también PI, PD o P en la ausencia de las acciones de control respectivas. Los controladores PI son particularmente comunes, ya que la acción derivativa es muy sensible al ruido, y la ausencia del proceso integral puede evitar que se alcance al valor deseado debido a la acción de control.

Los controladores PID se encuentran en un gran número en todas las industrias y se presentan de muchas formas diferentes. El controlador PID es un elemento clave de los sistemas para control de motores, y un componente importante de los sistemas distribuidos para control de procesos.

Se encuentran en sistemas tan diversos como los reproductores de CD y DVD, y el control de velocidad de los coches. En control de procesos, más del 95% de los lazos de control son del tipo PID, en la mayoría de los casos, son realmente control PI.

La importancia de conocer los controladores PID es que son muy empleados en la industria, y todavía no se hace un uso correcto de ellos. Los problemas en la medida se deben a utilizar un filtro anti-aliasing mal diseñado o incluso a no emplearlo, a una excesiva acción de filtrado en sensores inteligentes, a la existencia de mucho ruido, o a una mala calibración. Es posible conseguir mejoras sustanciales, pero se requiere un conocimiento específico del proceso así como del control PID.



## APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

### 3.- ESTUDIO DE LAS NECESIDADES, LIMITACIONES Y CONDICIONANTES.

#### 3.1 Descripción de la aplicación

La aplicación está dividida en tres bloques: Principal, Especificaciones y Calcular PID. Cada uno de ellos nos permite realizar una serie de funcionalidades concreta, y todas ellas están relacionadas por medio de variables de programación comunes.

##### 3.1.1 Principal

Este es el apartado dedicado al análisis, introducción de información y conclusión para los resultados obtenidos. Se podrán realizar las simulaciones y obtener gráficas que reflejen el comportamiento de los sistemas.

**Planta:** Una planta puede ser una parte de un equipo, tal vez un conjunto de los elementos de una máquina que funcionan juntos, y cuyo objetivo es efectuar una operación particular.

**Proceso:** Se define proceso como una operación o un desarrollo natural progresivamente continuo, marcado por una serie de cambios graduales que se suceden unos a otros de una forma relativamente fija y que conducen a un resultado o propósito determinado; o una operación artificial o voluntaria que se hace de forma progresiva y que consta de una serie de acciones o movimientos controlados, sistemáticamente dirigidos hacia un resultado determinado.

Para describir la conducta de los procesos se suelen utilizar modelos matemáticos. Los modelos proporcionan una forma unificada para tratar con sistemas de tipos muy diferentes, que hacen posible introducir una serie de conceptos útiles. Los modelos son esenciales para la simulación y el diseño del control.



## APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Para controlar un sistema es necesario conocer la dinámica del proceso. Para ello es conveniente conocer los conceptos de respuesta escalón, respuesta impulsional y función de transferencia.

**Modelos estáticos:** Se describe la conducta estacionaria del proceso mediante una curva que muestra el valor en estado estacionario de la variable del proceso y (salida) para diferentes valores de la variable manipulada  $u$  (entrada). Puede emplearse para determinar el rango de señales de control requeridas para cambiar la salida del proceso en el rango deseado, para dimensionar los actuadores y para determinar la resolución del sensor.

Se puede obtener el modelo estático experimentalmente manteniendo la entrada en un valor constante y midiendo la salida en estado estacionario. Esto da un punto de la característica del proceso. El experimento se debe repetir para cubrir el rango completo de entradas.

**Modelos dinámicos:** Debe proporcionar la relación entre la entrada y la salida durante el transitorio. Cabe distinguir entre sistemas estables y sistemas inestables. La respuesta en escalón de un sistema estable va a un valor constante. Un sistema inestable no alcanzará un estado estacionario después de un cambio en escalón.

**Modelo físico:** Una forma tradicional de obtener el modelo de un proceso es usar las leyes físicas básicas tales como balances de masas, momentos y energía. Tales descripciones típicamente conducen a un modelo matemático en términos de ecuaciones diferenciales.



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

**Función de transferencia:** La función de transferencia  $G(s)$  es la transformada de Laplace de la respuesta impulsional  $g(t)$ . La transformada de Laplace de la salida es simplemente la transformada de Laplace de la entrada multiplicada por la función de transferencia del sistema. Ésta es una de las razones principales para utilizar la transformada de Laplace cuando se analizan sistemas lineales. El análisis de sistemas lineales se reduce a pura álgebra. Un detalle agradable es que procesos, controladores y señales se describen de la misma forma.

La función de transferencia de un sistema descrito mediante una ecuación diferencial lineal e invariante en el tiempo se define como el cociente entre la transformada de Laplace de la salida y la transformada de Laplace de la entrada, suponiendo que todas las condiciones iniciales sean 0.

$$G(s) = \frac{\mathcal{L}[Y]}{\mathcal{L}[U]} = \frac{Y(s)}{U(s)}$$

**Diagrama de bloques:** Un diagrama de bloques de un sistema es una representación gráfica de las funciones que lleva a cabo cada componente y el flujo de señales. Tales diagramas muestran las relaciones existentes entre los diversos componentes. A diferencia de una representación matemática puramente abstracta, un diagrama de bloques tiene la ventaja de indicar de forma más realista el flujo de las señales del sistema real.

En el diagrama de bloques todas las variables del sistema se enlazan unas con otras mediante bloques funcionales. Estos bloques son un símbolo para representar la operación matemática que sobre la señal de entrada hace el bloque para producir la salida. Las funciones de transferencia de los componentes por lo general se introducen en los bloques correspondientes, que se conectan mediante flechas para indicar la dirección del flujo de señales.

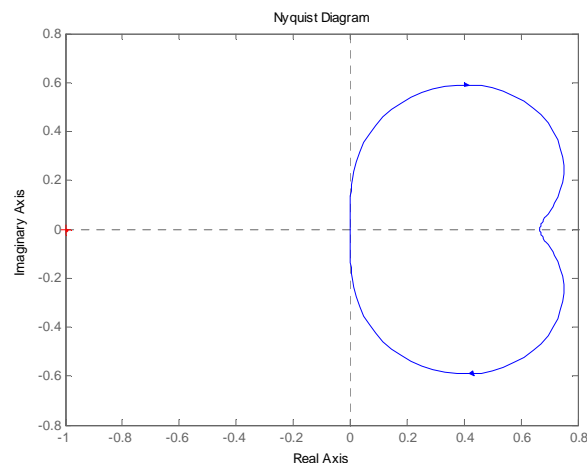


**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**



**Respuesta en frecuencia:** Otra forma de caracterizar la dinámica de un sistema lineal e invariante es investigar la respuesta del sistema a señales de entradas sinusoidales, una idea que se retrotrae al matemático francés Fourier. Es menos intuitiva que la respuesta transitoria pero da otras perspectivas.

**Diagrama de Nyquist:** El número complejo  $G(j\omega)$  se puede representar por un vector de longitud  $a(\omega)$  que forma un ángulo  $\theta(\omega)$  con el eje real. Cuando la frecuencia va de 0 a  $\infty$ , el vector describe una curva en el plano, que se llama curva de frecuencia o diagrama de Nyquist. El diagrama de Nyquist da una descripción completa del sistema.



**Figura 1.1. Diagrama de Nyquist**

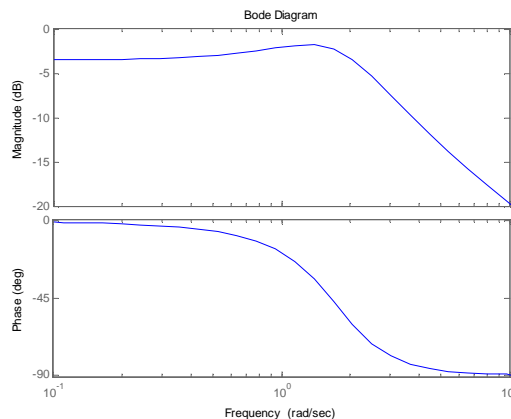




### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

**Diagrama de Bode:** Es otra representación gráfica de la función de transferencia. Consiste en dos curvas, la curva de ganancia y la curva de fase, como se muestra en la figura 1.2. La curva de ganancia muestra la razón de amplitudes  $a(\omega)=|G(j\omega)|$  en función de la frecuencia  $\omega$ . La curva de fase muestra la fase  $\theta(\omega)=\arg G(j\omega)$  en función de la frecuencia  $\omega$ . La ganancia y la frecuencia (rad/seg) se expresan en escalas logarítmicas.

La representación común de la magnitud logarítmica es  $G(j\omega)$  es  $20 \cdot \log |G(j\omega)|$  donde la base del logaritmo es 10. La unidad empleada en esta magnitud es el decibelio (dB). La principal ventaja que representa es que la multiplicación de magnitudes se convierte en sumas.



**Figura 1.2. Diagrama de Bode**

La característica básica de la respuesta transitoria de un sistema en lazo cerrado se relaciona estrechamente con la localización de los polos en bucle cerrado. Si el sistema tiene una ganancia de lazo variable, la localización de los polos en lazo cerrado depende del valor de la ganancia elegida. Es importante que el diseñador conozca cómo se mueven los polos en lazo cerrado conforme varía la ganancia del lazo.



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

**Diagrama del lugar de las raíces:** Lugar geométrico de los puntos del plano complejo que son solución de la ecuación característica para valores de  $K \in [0, +\infty]$ .

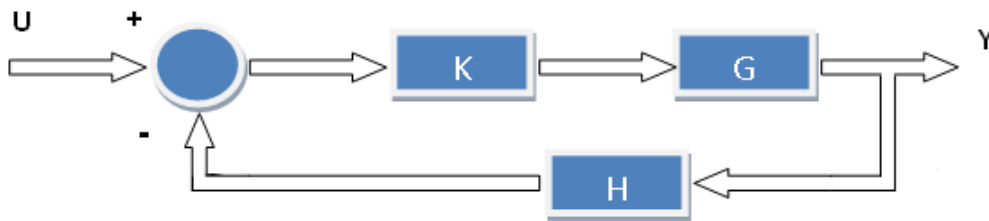


Figura 1.3. Diagrama de bloques de un sistema

- Ecuación característica

$$1 + K \cdot G_{BA} = 0$$

$$1 + K \cdot \frac{N_{BA}}{D_{BA}} = 0$$

$$D_{BA} + K \cdot N_{BA} = 0$$

$$\frac{D_{BA}}{K} + N_{BA} = 0$$

- Puntos iniciales del lugar de las raíces:  $K=0$

$$D_{BA} = D_G \cdot D_H = 0$$

Los polos del bucle cerrado para  $K=0$  son igual a los polos del bucle abierto.

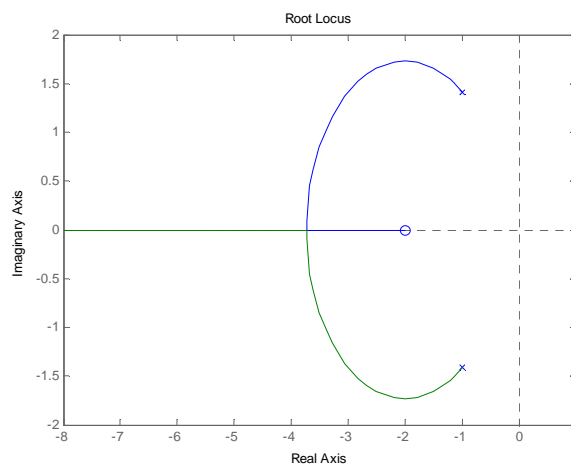
- Puntos finales del lugar de las raíces:  $K=+\infty$

$$N_{BA} = N_G \cdot N_H = 0$$



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Los polos en bucle cerrado para  $K=+\infty$  son igual a los ceros del bucle abierto. En la figura 1.4 se muestra un ejemplo de representación del lugar de las raíces de un sistema dado.



**Figura 1.4. Lugar de las raíces**

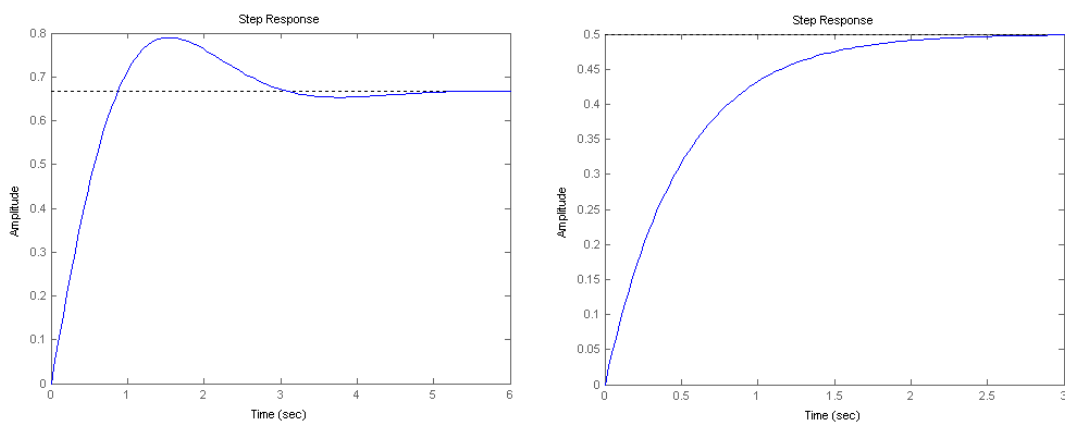
**Estabilidad:** La noción de estabilidad es intuitivamente muy sencilla, dice como se comporta un sistema después de una perturbación. Para analizar la estabilidad de un sistema debemos estudiar su ecuación característica, o lo que es lo mismo, igualar a 0 el denominador de su función de transferencia y obtener sus raíces.

Las raíces del denominador es lo que se llama polos del sistema. En el caso de tener un sistema continuo, este se considerará estable si no tiene polos con parte real positiva. Los polos situados sobre el eje imaginario son considerados como inestables. Para el caso de tener un sistema en el plano z, se considerará estable al sistema, siempre que los polos se encuentren dentro de una circunferencia de radio la unidad, centrada en el origen. Los polos situados sobre el radio de la circunferencia serán considerados como inestables.

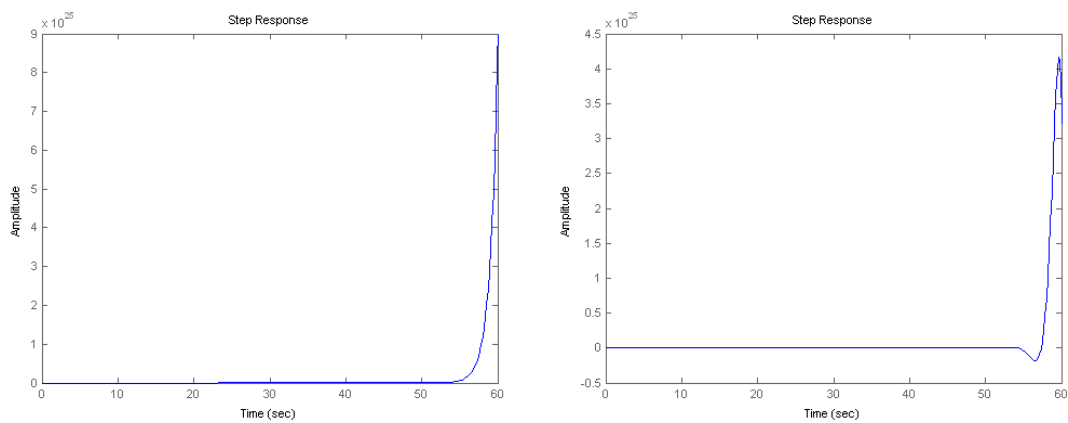


**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

En la figura 1.5 vemos varios ejemplos de respuesta estable, y en la figura 1.6 de respuesta inestable.



**Figura 1.5. Respuesta estable**



**Figura 1.6. Respuesta inestable**

La primera imagen de la figura 1.5 muestra una respuesta de un sistema subamortiguada, es decir que presenta oscilación. Esto se debe que haya polos del sistema complejos. Por otro lado, la otra imagen de la figura 1.5 representa una respuesta sobreamortiguada, o lo que es lo mismo, que no tiene oscilaciones.



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

**3.1.2 Especificaciones**

**Tiempo de establecimiento:** Tiempo que tarda la salida en alcanzar un porcentaje determinado del valor final. Puede establecerse habitualmente atendiendo a dos criterios, 98% y 95%.

$$t_{e(98\%)} = \frac{4}{\sigma} \text{ (seg)}$$

$$t_{e(95\%)} = \frac{\pi}{\sigma} \text{ (seg)}$$

Donde  $\sigma$  es el polo dominante, que es el polo que marca la dinámica del sistema, el polo más lento. Para el caso continuo será el polo más próximo al eje imaginario, y en caso discreto el más alejado del cero.

**Tiempo de pico:** Es el tiempo que tarda la salida hasta alcanzarse el primer pico de sobreoscilación.

$$t_p = \frac{\pi}{\omega_p} \text{ (seg)}$$

**Sobreoscilación:** Se define como la amplitud de la primera oscilación en porcentaje sobre el valor final de referencia.

$$\delta = e^{\frac{-\sigma \cdot \pi}{\omega_p}} \text{ (\%)}$$

**Error en régimen permanente (error estacionario):** Diferencia entre la referencia aplicada al servosistema y la realimentación de la salida, una vez se alcanza el régimen permanente.



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

El error en régimen permanente depende de la entrada de referencia:

- Error de posición: La referencia es un escalón unitario  $R(s)=1/s$
- Error de velocidad: La referencia es una rampa unitaria  $R(s)=1/s^2$
- Error de aceleración: La referencia es una parábola unitaria  $R(s)=1/s^3$

**Tipo del servosistema:** Número de integradores en la función de transferencia en bucle abierto del servosistema.

- Tipo 0: no hay integradores en el bucle
- Tipo 1: un solo integrador en el bucle
- Tipo 2: dos integradores en el bucle

Tipo \ Error	$e_p$	$e_v$	$e_a$
0	$\frac{1}{1+k_p}$	$\infty$	$\infty$
1	0	$\frac{1}{k_v}$	$\infty$
2	0	0	$\frac{1}{k_a}$

Figura 1.7. Tabla de errores



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Para el caso de reguladores continuos, el valor de las constantes se obtendrá del siguiente modo:

$$K_p = \lim_{s \rightarrow 0} K \cdot G(s) \cdot G_R(s) \cdot H(s)$$

$$K_v = \lim_{s \rightarrow 0} s \cdot K \cdot G(s) \cdot G_R(s) \cdot H(s)$$

$$K_a = \lim_{s \rightarrow 0} s^2 \cdot K \cdot G(s) \cdot G_R(s) \cdot H(s)$$

Para el caso de reguladores discretos:

$$K_p = \lim_{z \rightarrow 1} K \cdot G(z) \cdot G_R(z) \cdot H(z)$$

$$K_v = \frac{1}{T} \lim_{z \rightarrow 1} \frac{(1 - z^{-1})}{z^{-1}} K \cdot G(z) \cdot G_R(z) \cdot H(z)$$

$$K_a = \frac{2}{T^2} \lim_{z \rightarrow 1} \frac{(1 - z^{-1})^2}{z^{-1}} K \cdot G(z) \cdot G_R(z) \cdot H(z)$$

#### 3.1.3 Cálculo de PID

El algoritmo PID se puede contemplar desde muchas direcciones diferentes. Se puede ver como un dispositivo que puede operarse con unas pocas reglas heurísticas, pero que también puede estudiarse analíticamente.

$$u(t) = K \cdot (e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt})$$



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

La señal de control se representa como  $u$ , y  $e$  es el error del control. La señal de control es la suma de tres términos: el término P, que es proporcional al error, el término I, que es proporcional a la integral del error, y el término D, que es proporcional a la derivada del error. Los parámetros del controlador son la ganancia proporcional  $K$ , el tiempo integral  $T_i$ , y el tiempo derivativo  $T_d$ .

#### **3.1.3.1 Acción proporcional**

La acción aplicada es proporcional al error. Es un control de presente.

$$u(t) = K \cdot e(t)$$

Un valor grande de  $K$  implica una acción de control fuerte, lo que produce una respuesta rápida del sistema, y por tanto peligro de inestabilidad. Con una ganancia grande, el error en régimen permanente es pequeño.

Por el contrario, un valor pequeño de  $K$  implica una acción de control suave, lo que produce una respuesta lenta del sistema, y por tanto mayor estabilidad. Con una ganancia pequeña, el error en régimen permanente es grande.

#### **3.1.3.2 Acción integral**

La acción aplicada es proporcional a la integral del error. Es un control de pasado. Las acciones son pequeñas cuando hay poco error acumulado, y grandes cuando el error acumulado es elevado.

$$u(t) = K_i \cdot \int_0^t e(\tau) d\tau$$





### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Las ventajas de incluir el control integral son que aumenta el tipo del servosistema, y conserva la historia del error. Las desventajas son que aumenta el peligro de inestabilidad y ralentiza la respuesta.

#### 3.1.3.3 Acción derivada

La acción aplicada es proporcional a la derivada del error. Es una acción de futuro. Las acciones son pequeñas cuando el error cambia lentamente y grandes cuando lo hace rápidamente.

$$u(t) = K_d \frac{de(t)}{dt}$$

Las ventajas de incluir acción derivada son que considera la tendencia del error, y aporta amortiguamiento adicional. Las desventajas son que amplifica los ruidos de alta frecuencia y aplica acciones elevadas ante cambios bruscos en la referencia.

#### 3.1.3.4 PID ISA

**ISA:** The Instrumentation, Systems and Automation Society.

Sólo se tiene en cuenta el formato ISA en representación continua. El regulador PID quedaría representado de la siguiente forma:

$$PID = k_r \frac{(s + a)}{s} (s + b)$$

Donde a es el cero del regulador que hace referencia a la acción integral, y b el cero de la acción derivativa. El regulador PID en formato estándar o ISA sería:

$$PID_{ISA} = k_p \left( 1 + \frac{1}{T_i s} + T_d s \right)$$



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

Donde  $k_p$  es la constante de ganancia proporcional,  $T_i$  es el tiempo de integración y  $T_d$  es el tiempo derivativo. Dichas constantes se obtienen del siguiente modo para los diferentes tipos de reguladores:

- **Regulador P**

$$P(s) = k_r$$

- **Regulador PI**

$$PI = k_r \frac{(s + a)}{s}$$

$$PI_{ISA} = k_p \left(1 + \frac{1}{T_i s}\right)$$

$$k_p = k_r$$

$$T_i = \frac{1}{a}$$

- **Regulador PD**

$$PD = k_r (s + b)$$

$$PD_{ISA} = k_p (1 + T_d s)$$

$$k_p = k_r \cdot b$$

$$T_d = \frac{1}{b}$$

- **Regulador PID**

$$PID = k_r \frac{(s + a)}{s} (s + b)$$

$$PID_{ISA} = k_p \left(1 + \frac{1}{T_i s} + T_d s\right)$$

$$k_p = k_r \cdot (a + b)$$



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

$$T_i = \frac{a + b}{a - b}$$

$$T_d = \frac{1}{a + b}$$

El equivalente en discreto sería:

$$PID = k_p \frac{(z + a)(z + b)}{(z - 1)z}$$

**3.1.3.5 Criterios para la obtención del regulador PID**

- **Criterio del argumento**

El criterio del argumento consiste en obtener la posición que el cero o ceros que forman el regulador PID deben tener. Esta posición se obtiene a partir del ángulo que debe formar con respecto a un punto de diseño seleccionado por el diseñador.

El punto de diseño normalmente se escoge atendiendo al comportamiento que se desea obtener del sistema.

$$s = -\sigma \pm j\omega_p$$

Donde  $\sigma$  es el polo dominante del sistema, y se obtiene a partir del tiempo de establecimiento, y  $\omega_p$  es la frecuencia y parte imaginaria del punto, y se obtiene a partir de la sobreoscilación.



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

La ecuación de la que obtenemos el ángulo que forma el cero del regulador con el punto de diseño, es la siguiente:

$$180^{\circ} = \sum_{i=1}^n \alpha_i - \sum_{j=1}^m \beta_j$$

Siendo  $\alpha_i$  los ángulos que forman los polos del sistema con el punto de diseño, y  $\beta_j$  los ángulos que forman los ceros del sistema con el punto de diseño.

A partir del ángulo calculado, se obtiene la ubicación del cero/s aplicando trigonometría.

#### - **Criterio del módulo**

El criterio del módulo se aplica para obtener el valor de la ganancia  $K_r$  que permite cumplir las especificaciones. Para obtener el valor de la ganancia se deben de conocer las distancias de los polos y ceros del sistema al punto de diseño seleccionado.

$$K_{LDR} = \frac{\prod d_{pi}}{\prod d_{zi}}$$

$$K_r = \frac{K_{LDR}}{K_G \cdot K_H}$$

#### - **Cancelación**

Consiste en cancelar el/los cero/s del sistema. Para poder aplicar este método los polos a cancelar deben ser polos estables en bucle abierto. Tampoco se cancelan polos complejos.



## APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

### - Criterio de 1/10

Este método supone una aproximación que se considera como válida, aunque no es un método exacto. Consiste en situar el cero en un décima parte del valor del polo dominante del sistema ( $\sigma$ ).

### 3.2 Planteamiento de necesidades

En el presente proyecto se pretende realizar el diseño de un regulador PID para un proceso concreto y unas especificaciones dadas. La aplicación pretende contemplar todos los conceptos teóricos descritos anteriormente.

El usuario de la aplicación introducirá la función de transferencia del proceso que se desea controlar y las especificaciones que espera conseguir para el conjunto del sistema.

Se necesitará un ordenador con la versión de Matlab 2010 b del 16 Agosto de 2010 instalada, para poder utilizar la interfaz gráfica y poder realizar las simulaciones del comportamiento de los sistemas.

Aunque no es objeto de este proyecto, pero para poder diseñar un regulador PID, antes es necesario conocer la ecuación matemática a la que responde el proceso a estudiar.

El comportamiento del proceso se puede obtener mediante las leyes físicas, o experimentalmente. En el caso de obtenerse de forma experimental, también serían necesarios aparatos de laboratorio, como fuente de alimentación, generador de señales y osciloscopio.



## APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

### **4.- PLANTEAMIENTO DE SOLUCIONES ALTERNATIVAS.**

#### **4.1 Obtención del modelo matemático**

Cuando se quiere realizar el control de cualquier tipo de sistemas, lo primero que se debe realizar es un estudio teórico sobre el mismo, para ello se debe calcular un modelo matemático que sea fiel al comportamiento del sistema en la realidad (función de transferencia), se entiende por función de transferencia la relación que existe entre la salida del sistema y la entrada al mismo. Para ello tenemos varias posibilidades una de ellas es el modelado teórico del sistema a controlar y otra es la identificación experimental, que puede ser de dos tipos. Identificación experimental mediante respuesta escalón o por mínimos cuadrados.

##### **4.1.1 Modelado Teórico**

El modelado teórico consiste, en obtener un modelo matemático que se comporte de igual forma que el sistema lo haría en la realidad. Este método se calcula de forma teórica, por ecuaciones de la Física Clásica o por otros métodos.

El primer paso que se debe hacer en el modelado teórico de cualquier sistema es la obtención de las ecuaciones diferenciales que definen dicho sistema. Se entiende por ecuación en diferencias lineales o diferenciales lineales a aquellas ecuaciones que constan de una suma de términos del tipo constante multiplicada por función o por una derivada de esta.

Una vez halladas se obtienen alrededor del punto de funcionamiento elegido con anterioridad.



### **APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

El segundo paso consiste, si existen ecuaciones no lineales, linealizarlas, tanto las lineales como las no lineales. Son ecuaciones lineales que cumplen con la definición anteriormente citada. Una vez están linealizadas, se sustituye el punto de funcionamiento en dichas ecuaciones.

El tercer y último paso consiste en obtener la Transformada Inversa de Laplace (TIL), que nos permite pasar de ecuaciones en diferencias a ecuaciones en dicho dominio. Una vez se aplicada la T.I.L. se calcula la función de transferencia del sistema a modelar.

#### **4.1.2 Identificación experimental.**

Para este caso, la obtención de la función de transferencia se realiza de forma experimental.

En este método, para el cálculo de la función de transferencia se conoce la entrada (se aplica la que se desee) y se observa como evoluciona el sistema ante la entrada aplicada, de esta manera se puede calcular la función de transferencia.

Para ello se dispone de varios métodos.

##### **4.1.2.1 Identificación ante respuesta escalón.**

Este tipo de identificación consiste en aplicar varias entradas de tipo escalón al proceso y estudiar su respuesta, para poder calcular la función de transferencia.

Para que la identificación sea correcta, todas las pruebas tienen que partir de las mismas condiciones, para ello primero el sistema debe alcanzar el punto de funcionamiento. Una vez alcanzado, se realiza la prueba y se observa la respuesta del sistema ante esa entrada.



## APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Para una correcta identificación es conveniente que se realicen varias pruebas, para obtener varias funciones de transferencia, para que esta sea lo más fiel al proceso real.

### 4.1.2.2 Identificación por mínimos cuadrados

Según Gauss el principio de mínimos cuadrados es tal que los parámetros desconocidos de un modelo se deben elegir de forma que “la suma de los cuadrados de las diferencias entre los valores observados realmente y los calculados multiplicada por números que midan el grado de precisión sea mínimo”.

### 4.2 Elección del modelo

Se elige un modelo sencillo para ser empleado a modo de ejemplo. El modelo elegido es una red formada por tres resistencias de valor  $100\text{K}\Omega$  y tres condensadores de  $100\text{nF}$  cada uno.

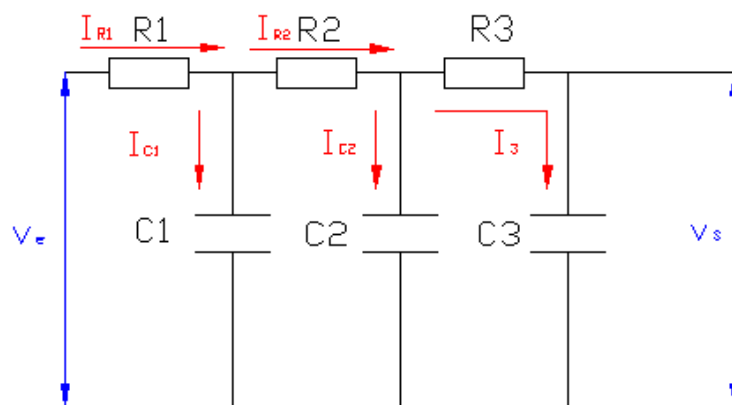


Figura 1.8. Esquema eléctrico de red RC3

En primer lugar se deben obtener las ecuaciones diferenciales que representan el comportamiento del sistema.





## PROYECTO FINAL DE CARRERA



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

La ecuación de salida del sistema será:

$$V_s(t) = V_{C3}(t)$$

La ecuación de entrada del sistema será:

$$V_e(t) = V_{R1}(t) + V_{C1}(t)$$

Las ecuaciones de la tensión en cada una de las resistencias serán:

$$V_{R1}(t) = R \cdot I_1(t)$$

$$V_{R2}(t) = R \cdot I_2(t)$$

$$V_{R3}(t) = R \cdot I_3(t)$$

Las ecuaciones de las corrientes de malla serán:

$$I_1(t) = I_{c1}(t) + I_2(t)$$

$$I_2(t) = I_{c2}(t) + I_3(t)$$

Las ecuaciones de la tensión en extremos de los condensadores serán:

$$V_{c1}(t) = \frac{1}{C} \int_0^t I_{c1}(\tau) d\tau$$

$$V_{c2}(t) = \frac{1}{C} \int_0^t I_{c2}(\tau) d\tau$$

$$V_{c3}(t) = \frac{1}{C} \int_0^t I_3(\tau) d\tau$$

Además, para terminar de definir el sistema serán necesarias dos ecuaciones más que relacionen diferentes variables del conjunto:

$$V_{c1}(t) = V_{R2}(t) + V_{c2}(t)$$

$$V_{c2}(t) = V_{R3}(t) + V_{c3}(t)$$



## PROYECTO FINAL DE CARRERA



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Puesto que todas las ecuaciones que definen al sistema son lineales, no será necesario linealizarlas para un punto de funcionamiento concreto. Así pues, se podrá aplicar la transformada de Laplace directamente a las ecuaciones, obteniéndose:

La ecuación de salida del sistema quedará como:

$$V_s(s) = V_{C3}(s)$$

La ecuación de entrada del sistema:

$$V_e(s) = V_{R1}(s) + V_{C1}(s)$$

Las ecuaciones de la tensión en cada una de las resistencias:

$$V_{R1}(s) = R \cdot I_1(s)$$

$$V_{R2}(s) = R \cdot I_2(s)$$

$$V_{R3}(s) = R \cdot I_3(s)$$

Las ecuaciones de las corrientes de malla:

$$I_1(s) = I_{c1}(s) + I_2(s)$$

$$I_2(s) = I_{c2}(s) + I_3(s)$$

Las ecuaciones de la tensión en extremos de los condensadores:

$$V_{c1}(s) = \frac{1}{C \cdot s} I_{c1}(s)$$

$$V_{c2}(s) = \frac{1}{C \cdot s} I_{c2}(s)$$

$$V_{c3}(s) = \frac{1}{C \cdot s} I_3(s)$$



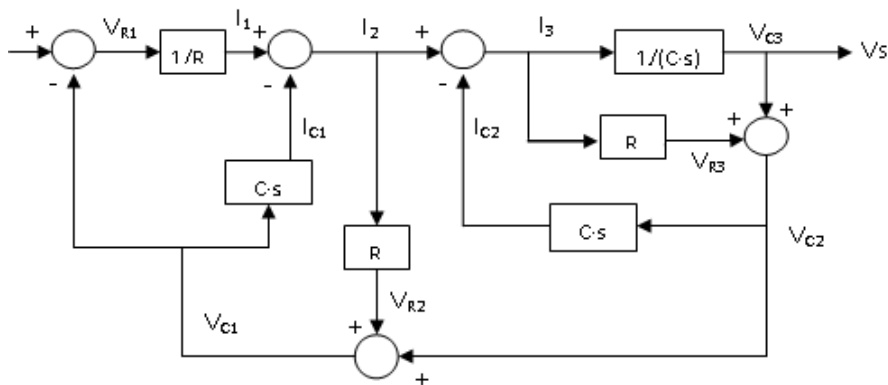
**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

Las ecuaciones de la tensión en extremos de los condensadores 1 y 2:

$$V_{c1}(s) = V_{R2}(s) + V_{c2}(s)$$

$$V_{c2}(s) = V_{R3}(s) + V_{c3}(s)$$

El siguiente paso será formar el diagrama de bloques del sistema:



**Figura 1.9. Diagrama de bloques de la red RC3**

Resolviendo el diagrama de bloques, obtendremos la función de transferencia del sistema:

$$\frac{V_s(s)}{V_e(s)} = \frac{1}{R^3 C^3 s^3 + 5R^2 C^2 s^2 + 6RCs + 1}$$

Sustituyendo por los valores de las resistencias y de los condensadores:

$$\frac{V_s(s)}{V_e(s)} = \frac{1}{10^{-6}s^3 + 0.0005s^2 + 0.06s + 1}$$

**4.3 Tipo de Control**

La aplicación nos permite obtener reguladores en continuo y en discreto. Cabe diferenciar entre ambos.



## APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

### 4.3.1 Continuo

Hace referencia al tratamiento de señales analógicas. El cálculo del regulador se hará en el dominio de Laplace.

Una señal analógica indica una señal que se puede representar matemáticamente por un conjunto de valores continuos. Contrariamente a una señal digital, que utiliza una serie de cantidades discretas para representar la señal.

Los valores analógicos representan típicamente un voltaje, una corriente eléctrica, o una carga eléctrica en torno a los componentes de los dispositivos electrónicos. Un error o ruido que afecte estas magnitudes físicas se traducirá en el error correspondiente en las señales representadas por dichas magnitudes físicas.

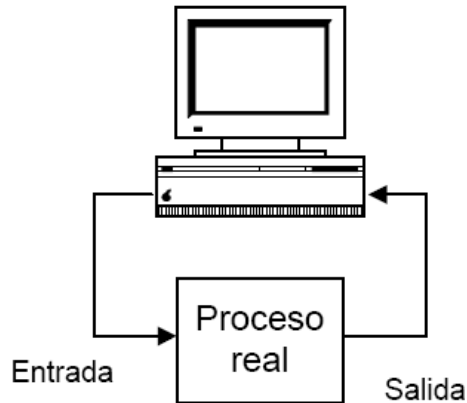
Su función es la hacer que el proceso se comporte de una manera determinada. Su implementación se realiza de la misma forma que los filtros analógicos, es decir con amplificadores operacionales, condensador, resistencias y demás elementos.

### 4.3.2 Discreto

El control por computador es hoy en día una herramienta común en la industria actual. Por tanto es importante entender los aspectos teóricos involucrados en los sistemas controlados por computador.



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

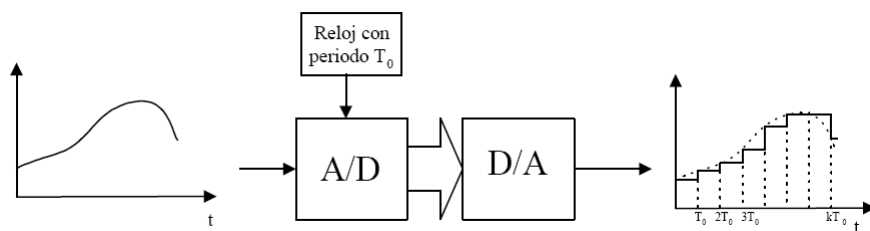


**Figura 1.10. Sistema de control por computador**

El gran avance de la tecnología digital, la computación y las tecnologías de información nos hace afirmar que en la actualidad todos los sistemas de control modernos están soportados por plataformas computacionales.

Con la tecnología digital podemos realizar algoritmos de control mas sofisticados, reconfigurarlos a medida que los procesos evolucionan con el tiempo. Los sistemas son más poderosos, flexibles y baratos que sus predecesores eléctricos o neumáticos.

Al hacer pasar una señal continua por un convertidor analógico- digital y luego seguido por un convertidor digital- analógico, a la salida se obtiene la señal discretizada.



**Figura 1.11. Obtención de una señal discreta**



## APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Cuando la existencia de computadores era nula o escasa, el control de procesos se realizaba de forma analógica, mediante reguladores analógicos. Estos tipos de reguladores estaban limitados ya que estaba condicionado por la acción de control (en algunos casos no se puede conseguir alcanzar dicha acción). Ha medida que la Teoría de Control avanzaba se necesitaba de reguladores mas completos y complejos, este factor junto con la fuerte expansión de los computadores y su abaratamiento, ha hecho que el control de procesos se haga habitualmente mediante computador.

### **5.- JUSTIFICACIÓN DE LA SOLUCIÓN ADOPTADA.**

#### **5.1 Control analógico**

A partir de la función de transferencia de la red RC3, diseñaremos varios reguladores PID para diferentes especificaciones, comparando los resultados obtenidos.

$$\frac{V_s(s)}{V_e(s)} = \frac{1}{10^{-6}s^3 + 0.0005s^2 + 0.06s + 1}$$

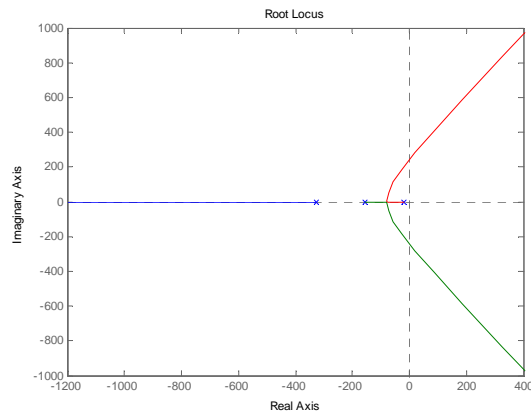
Este sistema también se puede representar en forma de polos y ceros, quedando:

$$\frac{V_s(s)}{V_e(s)} = \frac{1000000}{(s + 324.7) + (s + 155.5) + (s + 19.81)}$$

Se realiza un análisis del lugar de las raíces a partir de la función de transferencia de la red RC3, y suponiendo que se emplee un regulador de valor 1.



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**



**Figura 1.12. Lugar de las raíces de la red RC3**

Analizando la gráfica obtenida, sabemos que a partir de un cierto valor de ganancia, el sistema se volverá inestable. Obtenemos dicho valor para conocer el rango de ganancias en el cual el sistema se mantiene estable.

$$K_r \in [0, 27.623]$$

Definimos las siguientes especificaciones:

$$t_{e(98\%)} \leq 2 \text{ seg}$$

$$\delta \leq 4.33\%$$

Sabemos que para conseguir estas especificaciones,  $\sigma=2$  y  $\varphi=45^\circ$ , por tanto diseñaremos un regulador según el punto de diseño  $s=-2\pm 2j$ .

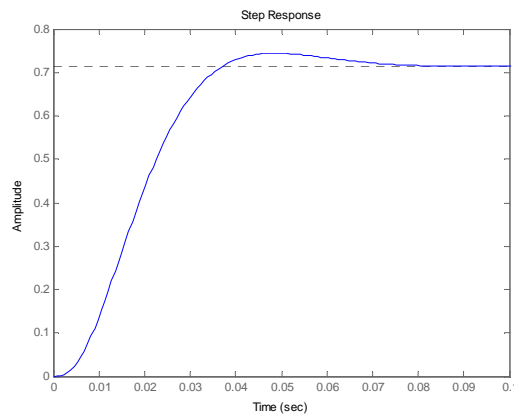


### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Para conseguir estas especificaciones, sería suficiente con un regulador de tipo P, ya que el sistema cumple las especificaciones dinámicas, y de momento no se ha planteado nada referente a las especificaciones estáticas. El regulador P necesario sería el siguiente:

$$P = 2.5066$$

Con este regulador el sistema tendría un  $t_{e(98\%)} = 0.0633 \text{ seg}$  y  $\delta = 4.12\%$ . Si atendemos a las especificaciones estáticas, como el sistema es de tipo 0, tendrá error de posición de valor 28.52%.



**Figura 1.13. Respuesta ante escalón unitario del sistema**

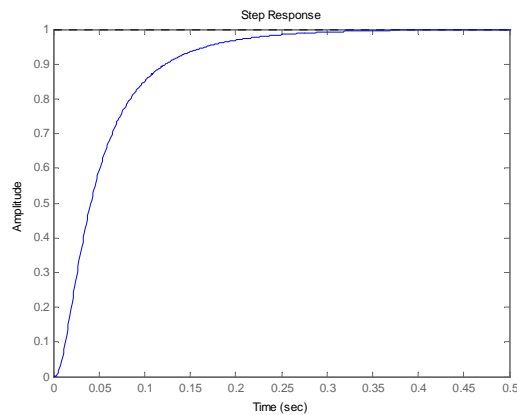
En la figura 1.13 se puede apreciar como se comporta la salida del sistema cuando a la entrada se aplica un escalón de valor la unidad. Se puede apreciar el error de posición al que se hace referencia, ya que la salida no es capaz de seguir fielmente al valor de entrada.

Para tratar de corregir este error de posición se debe de añadir parte integral al regulador. Al diseñar el regulador PI, este error de posición se reducirá al 0%.





**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**



**Figura 1.14. Respuesta ante escalón unitario del sistema con regulador PI**

En la figura 1.14 se puede apreciar que ya no existe error en la salida, el regulador PI diseñado que nos permite obtener esta salida en bucle cerrado es:

$$PI = \frac{(s + 17.17)}{s}$$

Además debemos asegurar que el sistema sigue cumpliendo las especificaciones dinámicas, que en el caso de aplicar este regulador serán un  $t_{e(98\%)} = 0.232 \text{ seg}$  y una sobreoscilación  $\delta \cong 0\%$ . Así que el regulador PI diseñado sería válido para cumplir las especificaciones dinámicas, y no tendríamos error de posición.

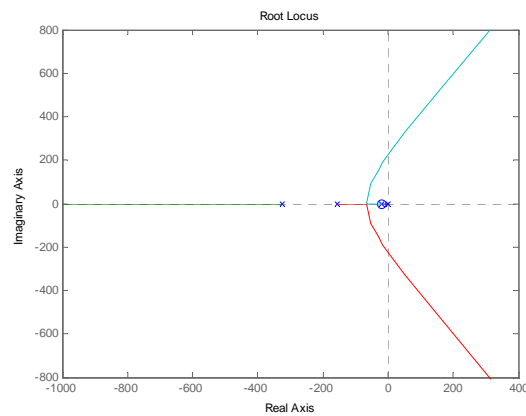
Este regulador se ha obtenido aplicando el criterio del argumento al punto de diseño seleccionado. Ahora se estudia la respuesta en el caso de aplicar cancelación para obtener el regulador PI.

$$PI = \frac{(s + 19.81)}{s}$$



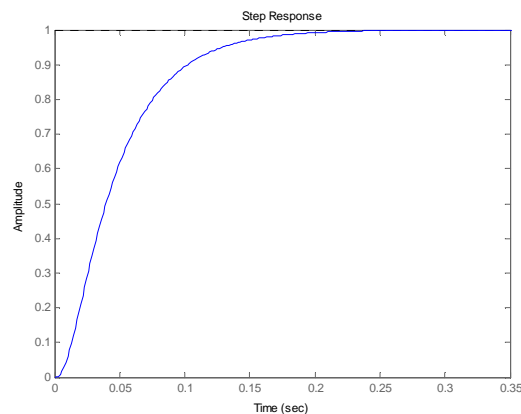
### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Es habitual cuando se aplique el método de cancelación, tratar de cancelar el polo más lento en bucle abierto, siempre que este no sea inestable. En este caso, el polo que marca la dinámica del sistema es  $s=-19.81$ . Analizamos el lugar de las raíces para el nuevo sistema con el regulador PI:



**Figura 1.15. Lugar de las raíces del sistema con regulador PI**

Ahora observamos como será la respuesta ante escalón unitario del sistema con el nuevo regulador PI.



**Figura 1.16. Respuesta ante escalón unitario del sistema con regulador PI**



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

En la figura 1.16 se puede apreciar la respuesta del sistema ante escalón unitario con el regulador calculado por el método de cancelación. De esta gráfica se deduce que el error de posición continúa siendo 0, y que sus especificaciones dinámicas son un  $t_{e(98\%)} = 0.165 \text{ seg}$  y una sobreoscilación  $\delta \cong 0\%$ .

Se puede deducir que con el regulador diseñado mediante el método de cancelación, el  $t_{e(98\%)}$  se ha reducido de 0.232 seg a 0.135 seg, mientras que la sobreoscilación sigue siendo del 0%.

De los resultados obtenidos, se deduce que el regulador  $PI = \frac{(s+19.81)}{s}$  ofrece un mejor resultado para las especificaciones deseadas, ya que nos ha reducido el tiempo de establecimiento.

En el supuesto de querer obtener otros resultados en función de unas especificaciones dadas, se puede plantear el diseño de un regulador de tipo PID.

#### 5.2 Control digital

De igual forma que para el caso continuo, a partir de la función de transferencia de la red RC3, diseñaremos varios reguladores PID para diferentes especificaciones, comparando los resultados obtenidos. El periodo de muestreo elegido para obtener el equivalente discreto es de  $T=0.01\text{seg}$ .

$$\frac{V_s(s)}{V_e(s)} = \frac{0.005571z^2 + 0.07582z + 0.0004693}{z^3 - 1.07z^2 + 0.2134z - 0.006738}$$

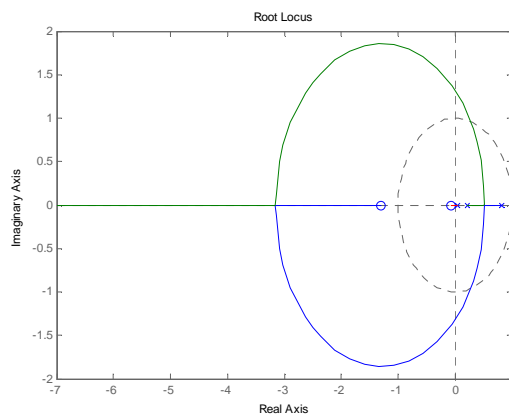


### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Este sistema también se puede representar en forma de polos y ceros, quedando:

$$\frac{V_s(s)}{V_e(s)} = \frac{0.005571(z + 1.296)(z + 0.065)}{(z - 0.8203) + (z - 0.2112) + (z - 0.03889)}$$

Se realiza un análisis del lugar de las raíces a partir de la función de transferencia de la red RC3, y suponiendo que se emplee un regulador de valor 1.



**Figura 1.17. Lugar de las raíces discreto de la red RC3**

Analizando la gráfica obtenida, sabemos que a partir de un cierto valor de ganancia, el sistema se volverá inestable. Este margen de ganancia viene establecido por la circunferencia de radio 1, a partir de ella, el sistema será inestable. Obtenemos dicho valor para conocer el rango de ganancias en el cual el sistema se mantiene estable.

$$K_r \in [0, 10.1]$$



## PROYECTO FINAL DE CARRERA



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

Definimos las siguientes especificaciones:

$$t_{e(98\%)} \leq 2 \text{ seg}$$

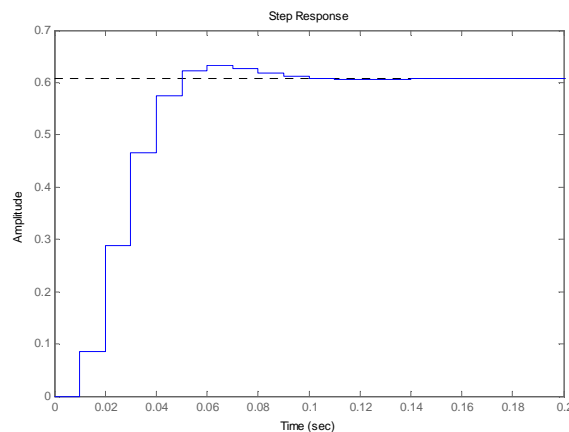
$$\delta \leq 4.33\%$$

Sabemos que para conseguir estas especificaciones,  $\sigma=2$  y  $\varphi=45^\circ$ , por tanto diseñaremos un regulador según el punto de diseño  $s=-2\pm 2j$ . A diferencia del caso continuo, ahora deberemos discretizar también las especificaciones con  $z = e^{ST}$ . Por tanto  $z=0.493\pm 0.3j$ .

Para conseguir estas especificaciones, sería suficiente con un regulador de tipo P, ya que el sistema cumple las especificaciones dinámicas, y de momento no se ha planteado nada referente a las especificaciones estáticas. El regulador P necesario sería el siguiente:

$$P = 1.55$$

Con este regulador el sistema tendría un  $t_{e(98\%)} = 0.0791 \text{ seg}$  y  $\delta = 4.12\%$ . Si atendemos a las especificaciones estáticas, como el sistema es de tipo 0, tendrá error de posición de valor 39.2%.



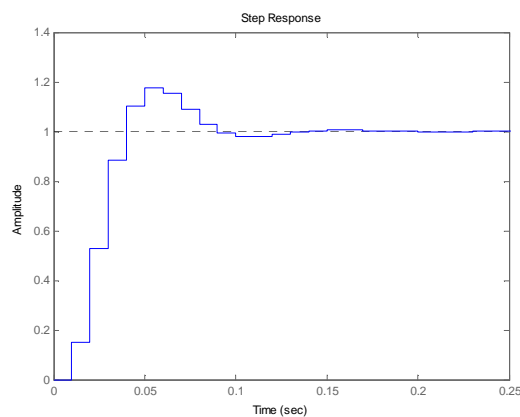
**Figura 1.18. Respuesta ante escalón unitario del sistema discreto**



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

En la figura 1.18 se puede apreciar como se comporta la salida del sistema cuando a la entrada se aplica un escalón de valor la unidad. Se puede apreciar el error de posición al que se hace referencia, ya que la salida no es capaz de seguir fielmente al valor de entrada.

Para tratar de corregir este error de posición se debe de añadir parte integral al regulador. Al diseñar el regulador PI, este error de posición se reducirá al 0%.



**Figura 1.19. Respuesta ante escalón unitario del sistema con regulador PI**

En la figura 1.19 se puede apreciar que ya no existe error en la salida, el regulador PI diseñado que nos permite obtener esta salida en bucle cerrado es:

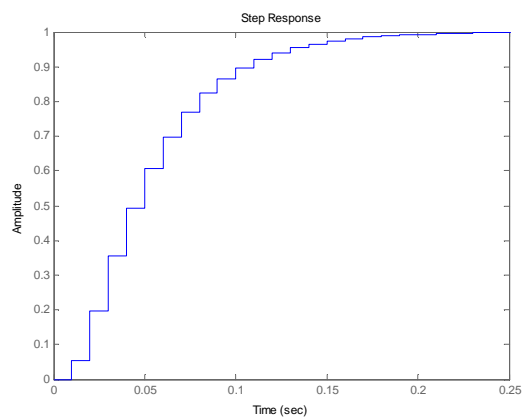
$$PI = \frac{2.7228(z - 0.8203)}{(z - 1)}$$

Además debemos asegurar que el sistema sigue cumpliendo las especificaciones dinámicas, que en el caso de aplicar este regulador serán un  $t_{e(98\%)} = 0.106 \text{ seg}$  y una sobreoscilación  $\delta \cong 17.7\%$ .



### APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS

En este caso el regulador PI diseñado no sería válido, ya que no cumple las especificaciones dinámicas, y en concreto la sobreoscilación. Debemos diseñar otro regulador PI que cumpla con las especificaciones.



**Figura 1.20. Respuesta ante escalón unitario del sistema con regulador PI**

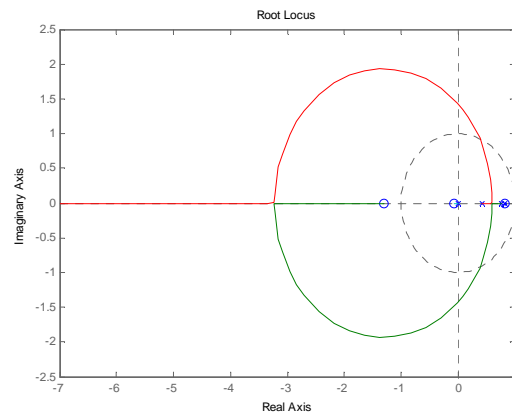
Comprobamos que el sistema cumple las especificaciones dinámicas, que en el caso de aplicar este regulador serán un  $t_{e(98\%)} = 0.16 \text{ seg}$  y una sobreoscilación  $\delta \cong 0\%$ . El nuevo regulador PI que nos permite conseguir estas especificaciones es:

$$PI = \frac{(z - 0.8203)}{(z - 1)}$$

En último lugar se obtiene el nuevo lugar de las raíces para el sistema formado con el proceso inicial y el regulador PI diseñado.



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**



**Figura 1.21. Lugar de las raíces del sistema con regulador PI**





**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

**6.- BIBLIOGRAFÍA**

- Manuel Gil Rodríguez y Díaz de Santos « Introducción rápida a MATLAB y SIMULINK para ciencia e ingeniería »
- Ana Rosa Carrera Amuriza. « Introducción a MATLAB y a la creación de interfaces gráficas».
- K. J. Aström y T. Hägglund. «Control PID avanzado». Editorial: Prentice Hall.
- K. Ogata (1980) «Ingeniería de control moderna». Editorial: Prentice Hall.
- Holly Moore. « Matlab para ingenieros ». Editorial: Prentice Hall.
- César Pérez. « Matlab y sus aplicaciones en las Ciencias y la Ingeniería ». Editorial: Prentice Hall.



PROYECTO FINAL DE CARRERA



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

**1.- INTRODUCCIÓN.**

Para la realización del presupuesto se considera que el presente proyecto ha sido realizado por un Ingeniero Técnico Industrial especialidad Electrónica Industrial.

Por tratarse de un proyecto fin de carrera no se incluye el I.V.A. en los precios considerados, así como tampoco se añade el beneficio industrial ni gastos generales al precio final. Además tampoco está previsto el pago de ninguna cantidad económica en concepto de honorario.

En la utilización de los materiales hardware (ordenador) y software, se considera su totalidad ya que se han adquirido expresamente para dicho proyecto.

**2.- PRESUPUESTO**

Para obtener el presupuesto han de definirse previamente los precios unitarios de cada unidad de obra. Pero para definir el coste de cada precio unitario ha de componerse cada uno de ellos con las cantidades de materiales y con la mano de obra que intervienen en su elaboración. Por ello, es preciso conocer una serie de cuadros de precios que llevan a determinar el precio de cada unidad de obra.

**2.1.- Cuadros de Precios**

**2.1.1.- Cuadro de Precios nº 1: Mano de Obra**

Los precios por hora cobrados por los trabajos realizados en el proyecto son:

DESCRIPCIÓN DE OPERARIOS	PRECIO (€ /hora)
• Ingeniero Técnico Industrial, especialidad Electrónica Industrial.	36.06

**TABLA 1: Cuadro de Precios nº 1: Mano de Obra**

**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

Estos precios incluyen el salario base, el porcentaje de los pluses y pagos a la Seguridad Social. Considerando que la semana presenta 5 días hábiles y trabajando 8 horas por día, resulta la siguiente lista de precios:

DESCRIPCIÓN DE OPERARIOS	PRECIO (€/semana)
<ul style="list-style-type: none"><li>Ingeniero Técnico Industrial, especialidad Electrónica Industrial.</li></ul>	1.442,4

***TABLA 2: Precio Semanal*****2.1.2.- Cuadro de Precios nº 2: Materiales**

DESCRIPCIÓN DE MATERIALES	PRECIO (€)
<ul style="list-style-type: none"><li>Ud. de ordenador portátil Packard Bell o compatible con procesador Intel Cuore Dúo 1,73GHz o superior, ratón compatible, monitor de 15,4 pulgadas, 66,5 Gbyte de disco duro y 896 Mb de memoria RAM con Windows XP Profesional instalado.</li></ul>	524,79
<ul style="list-style-type: none"><li>Ud. de programa MATLAB versión 2010b con SIMULINK v.3.0.c. de la firma Mathworks Inc.</li></ul>	1.402,99
<ul style="list-style-type: none"><li>Ud. de programa Word versión 2007 de la firma Microsoft Corporation.</li></ul>	599,00
<ul style="list-style-type: none"><li>Ud. de programa AutoCad 2008 de la firma Autodesk, Inc.</li></ul>	1178,82

***TABLA 3: Cuadro de Precios nº 2: Materiales***



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

Dentro del material utilizado, se encuentra el que no ha sido comprado específicamente para el presente proyecto. Sin embargo, su uso puede representar un coste de oportunidad o al menos el coste correspondiente al pago amortizado. La fórmula a aplicar es la siguiente:

$$Coste = Coste\_del\_material \times \frac{semanas\_de\_uso}{semanas\_de\_amortización} \quad (1)$$

Considerando un período de amortización de 2 años, resultan 104 semanas de amortización. A continuación se exponen los precios por la utilización durante una semana del siguiente material amortizable.

DESCRIPCIÓN DE MATERIALES	PRECIO (€/semana)
• Ud. de semana de utilización de programa MATLAB versión 2010b con SIMULINK v.3.0.c. de la firma Mathworks Inc.	161,88
• Ud. de semana de utilización de programa Word versión 2007 de la firma Microsoft Corporation.	34,56
• Ud. de semana de utilización de programa AutoCad versión 14 de la firma Autodesk, Inc.	11,33

**TABLA 4: Material Amortizable**

APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**2.1.3.- Cuadro de Precios nº 3: Precios Unitarios**

El presente cuadro expresa el precio de cada unidad de obra.

Nº de Orden	Descripción de la Unidad de Obra	PRECIO (€)
1	Ud. de estudio y obtención de las adecuadas estrategias de control e implementación de un sistema informático de tiempo real, utilizando los medios materiales disponibles tanto hardware como software y mano de obra.	<b>7.993,9</b>
2	Ud. de mecanografiado y elaboración de documentos, incluyendo realización de gráficos y mecanografiado de ecuaciones, utilizando los medios materiales disponibles tanto hardware como software y mano de obra.	<b>642,54</b>

**TABLA 5: Cuadro de Precios nº 3: Precios Unitarios**

**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS****2.1.4.- Cuadro de Precios nº 4: Precios Descompuestos.**

Los precios descompuestos indican los precios de las unidades de obra, indicando en cada una las cantidades de materiales y los tiempos empleados de la mano de obra

Ud. de estudio y obtención de las adecuadas estrategias de control e implementación de un sistema informático de tiempo real, utilizando los medios materiales disponibles tanto hardware como software y mano de obra.		Precio Unitario	Importe (€)	TOTAL (€)
<b>Materiales</b>	• 10 semanas de utilización de ordenador	60,55	605,5	
	• 10 semanas de utilización del programa Matlab 2010b	161,88	1618,8	
	• 4 semanas de Ingeniero Técnico Industrial especialidad Electrónica Industrial	1.442,40	5.769,60	
<b>SUMA</b>				<b>7.993,9 €</b>

**TABLA 6: Cuadro de Precios nº 4: Precios Descompuestos**



**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

Ud. de mecanografiado y elaboración de documentos, incluyendo realización de gráficos y mecanografiado de ecuaciones, utilizando los medios materiales disponibles tanto hardware como software y mano de obra.			
		<b>Precio Unitario</b>	<b>Importe (€)</b>
			<b>TOTAL (€)</b>
<b>Materiales</b>	<ul style="list-style-type: none"> <li>• 7 semanas de utilización de ordenador</li> <li>• 6 semanas de utilización del programa Word</li> <li>• 1 semana de utilización del programa AutoCad</li> </ul>	60,55	423,85
		34,56	207,36
		11,33	11,33
<b>SUMA</b>			<b>642,54 €</b>

***TABLA 7: Cuadro de Precios nº 4: Precios Descompuestos***

A modo de síntesis se adjuntan las siguientes tablas donde se especifica el número de semanas invertidas por la mano de obra y de utilización del material durante la realización del presente proyecto.

DESCRIPCIÓN DE OPERARIOS	SEMANAS
<ul style="list-style-type: none"> <li>• Ingeniero Técnico Industrial especialidad Electrónica Industrial</li> </ul>	10

***TABLA 8: Semanas invertidas***





**APLICACIÓN EN MATLAB PARA EL DISEÑO DE PIDS**

DESCRIPCIÓN DE MATERIALES	SEMANAS
• Ud. de ordenador.	17
• Ud. de programa MATLAB versión 2010b.	10
• Ud. de programa Word versión 2007.	6
• Ud. de programa AutoCad versión 2008.	1

***TABLA 9: Semanas invertidas***

**2.2.- Presupuesto de Ejecución Material**

Nº Orden de la Unidad de Obra	PRECIO (€)
1	7.993,9
2	642,54
<b>TOTAL</b>	<b>8.636,44 €</b>

***TABLA 10: Presupuesto de Ejecución Material***

El presupuesto de ejecución material asciende a la cantidad de **Ocho Mil seiscientos treinta y seis euros y cuarenta y cuatro céntimos.**

Valencia a 26 de Junio de 2011.

Fdo. Carlos Granell Torres