# Informe de Laboratorio
# 2. Sesión Práctica

*Modelización y análisis de controladores en coma fija*


**Christopher Carmona**
**Aitor Salazar**

*Diseño de Controladores Digitales*,

*Diciembre 2023*

# Índice

# 1. Laboratorio 1.1

## 1.1. Ejercicio 1

El objetivo del primer ejercicio es entender el efecto del desfase que se da por el retardo de computación (también conocido como *signal skew*) y por el retardo de realimentación. Para ello hemos tomado la planta y el controlador continuo que nos han proporcionado. Cumplen las siguientes funciones de transferencia:

$$G_s = \frac{52{,}1}{1{,}21s^2 + s} \tag{1}$$

$$C_s = \frac{0{,}525s^2 + 5{,}022s + 4{,}4}{0{,}005s^2 + s} \tag{2}$$

Donde $G_s$ es la planta del motor y $C_s$ es el controlador PID continuo que nos han propuesto.

A continuación hemos muestreado las señales para tres frecuencias diferentes. Así, vamos a poder ver cómo de notorios son los efectos del desfase en función de la frecuencia de muestreo.

Para la primera frecuencia de muestreo hemos usado la función **bode** de MatLab y usando esta herramienta hemos podido tomar el valor del ancho de banda (BW) del sistema el cual es $BW = 34{,}1359\,\text{rad/s}$. Con este dato y la recomendación práctica de tomar una frecuencia de muestreo aproximadamente unas 10 veces mayor que el ancho de banda para así evitar los efectos de los posibles alias producidos por el bajo muestro, obtendremos el primer valor de frecuencia de muestreo, $f_{s1} = \frac{10BW}{2\pi} = 54{,}3273\,\text{Hz}$ lo cual nos da un periodo de muestreo de $T_{s1} = 0{,}0184\text{s}$.

Para la segunda frecuencia de muestreo tomaremos un valor frecuencia de muestreo sobremuestreado, es decir tomaremos un valor 100 veces el ancho de banda del sistema. Tras hacer los cálculos pertinentes obtenemos $f_{s2} = 543{,}273\,\text{Hz}$ y el periodo de muestreo $Ts2 = 0{,}00184\,\text{s}$.

Finalmente, el último periodo de muestreo será uno "ultracorto", el cual será 1000 veces mayor al ancho de banda del sistema, lo que nos da: $f_{s3} = 5432{,}73\,\text{Hz}$ y $Ts3 = 0{,}000184\text{s}$.

Los desfases de propagación los hemos tomado de forma relativa a la frecuencia de muestreo. Estos son $\tau_1$, $\tau_2$ y $\tau_3$ con tiempos 0,1, 0,5 y 1,5 veces el periodo de muestreo respectivamente. Estos desfases de propagación se muestran en la Tabla 1.

| $\mathbf{T_s}$ | $\tau_\mathbf{1}$ | $\tau_\mathbf{2}$ | $\tau_\mathbf{3}$ |
|---|---|---|---|
| 1,84e−2 | 1,84e−3 | 9,2e−3 | 2,76e−2 |
| 1,84e−3 | 1,84e−4 | 9,2e−4 | 2,76e−3 |
| 1,84e−4 | 1,84e−5 | 9,2e−5 | 2,76e−4 |

**Tabla 1:** *Periodos de muestreo y tiempos de desfase para retardos de computación y realimentación. Unidades: s.*

**Análisis del sistema continuo en lazo cerrado.** En lo que respecta al sistema continuo observamos un margen de fase de 124.51 °y un margen de ganancia infinito, ambos

positivos, lo cual indica que el sistema es estable en continuo. Analizando la ubicación de polos observamos que hay una quasi-cancelanción en el cuarto polo y el segundo cero de la Tabla 2. Además, tenemos una pareja de polos conjugados próximos al eje imaginario y otro en la recta real negativa. También vemos en el diagrama de bode que el sistema es de fase mínima lo cual también indica que ninguno de los polos es inestable, esto último ya era evidente con la ubicación de polos.

| **Polos** | **Ceros** |
|---|---|
| $-175{,}5239$ | $-8{,}5901$ |
| $-12{,}1639 + 8{,}5723\text{i}$ | $-0{,}9757$ |
| $-12{,}1639 - 8{,}5723\text{i}$ | - |
| $-0{,}9748$ | - |

**Tabla 2:** *Raíces del sistema en lazo cerrado.*

**Análisis del sistema muestreado.** Al hacer la discretización del sistema usando el método de Tustin, obtenemos los controladores que cumplen las siguientes funciones de transferencia en Z para las diferentes frecuencias de muestreo:

$$C_{z1} = \frac{40{,}24z^2 - 73{,}87z + 33{,}73}{z^2 - 0{,}7041z - 0{,}2959} \tag{3}$$

$$C_{z2} = \frac{89{,}46z^2 - 177{,}4z + 87{,}9}{z^2 - 1{,}689z + 0{,}6892} \tag{4}$$

$$C_{z3} = \frac{103{,}2z^2 - 206{,}2z + 103}{z^2 - 1{,}964z + 0{,}9639} \tag{5}$$

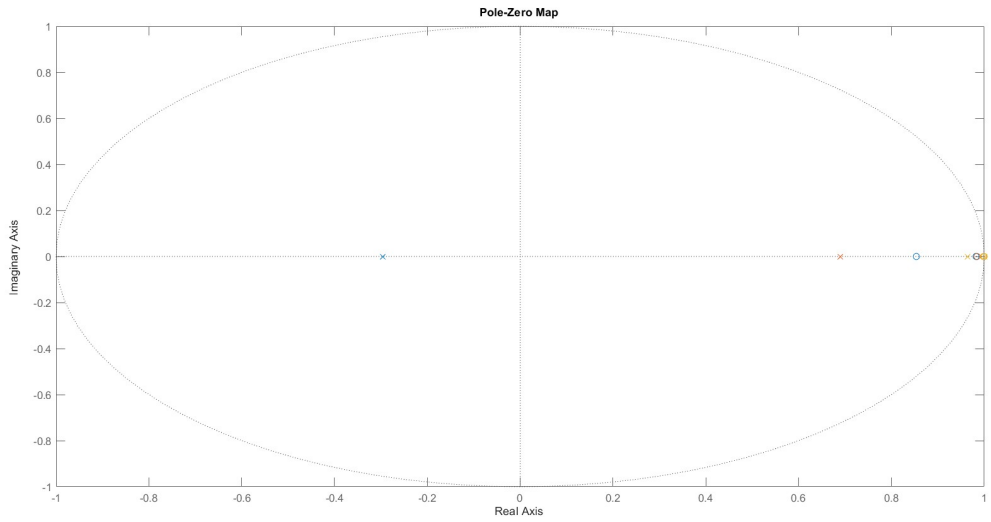



**Figura 1:** *Ubicación de polos de los controladores discretos con las 3 frecuencias de muestreo. Azul: $T_{s1}$, rojo: $T_{s2}$, amarillo: $T_{s3}$.*

Como muestra la gráfica de colocación de raíces de la Figura 1, el sistema es estable ya que tiene todos sus polos y ceros dentro del círculo unidad, pero a medida que la frecuencia

de muestreo aumenta vemos que los polos se aproximan al 1 real positivo y hace que su comportamiento se haga más críticamente estable. Esto puede tener implicaciones en el margen de ganancia y de fase del sistema.

Por otro lado la respuesta en frecuencia se puede ver en la Figura 2:

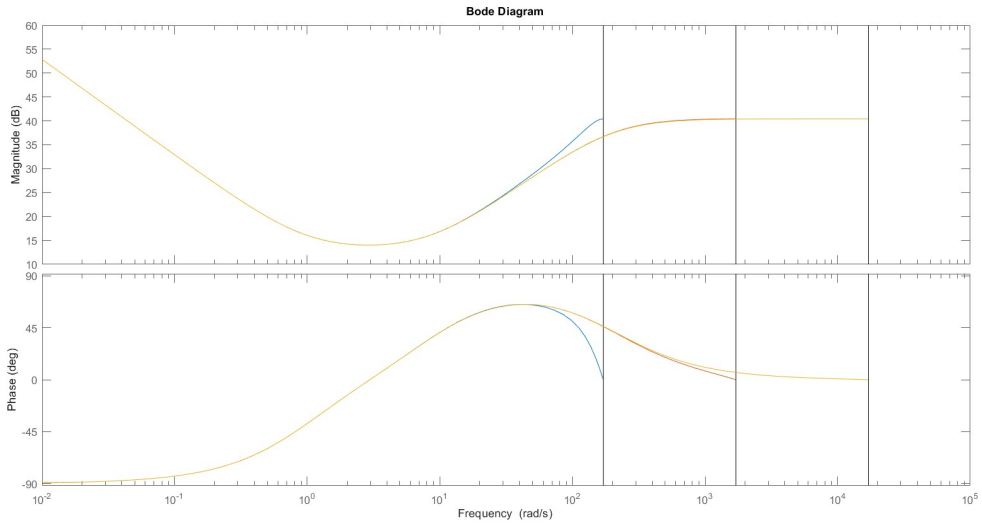



**Figura 2:** *Diagramas de bode de los 3 controladores discretos. Azul: $T_{s1}$, rojo: $T_{s2}$, amarillo: $T_{s3}$.*

Todos los controladores funcionan a modo de filtro paso bajo y añaden un desfase a las frecuencias más altas.

**Análisis de señales de salida con el controlador $\mathbf{T_{s1}}$.** Tras simular con el diagrama de Simulink de la Figura 3 que se nos ha proporcionado en la documentación, este es el resultado que obtenemos:

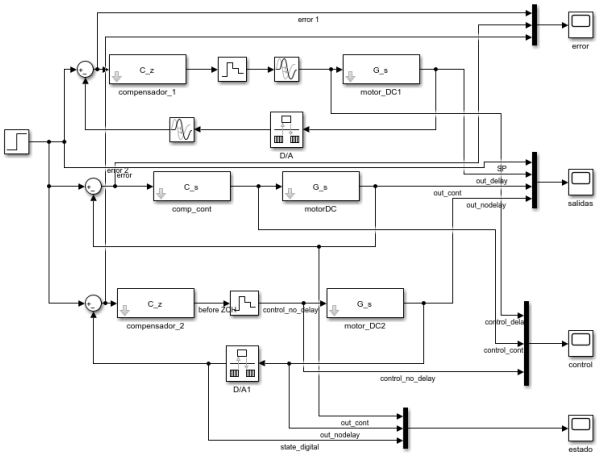
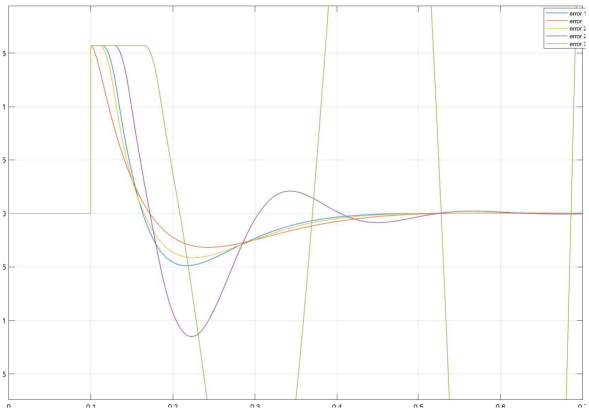
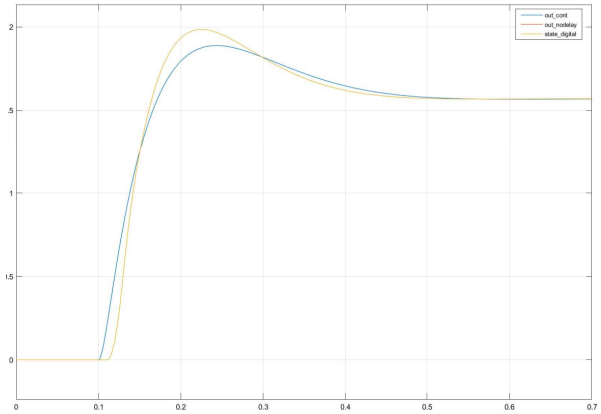


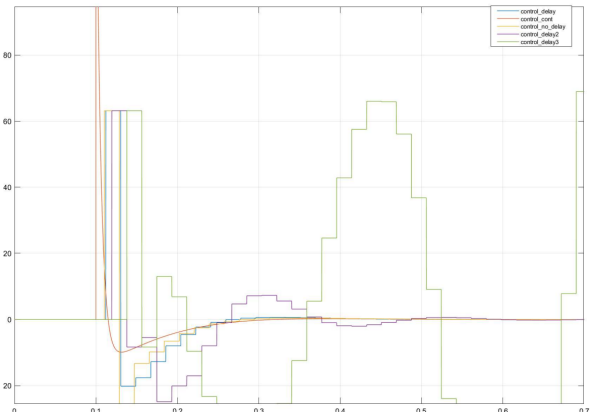

**Figura 3:** *Diagramas de Simulink del sistema en lazo cerrado continuo, discreto sin retardo y discreto con retardo.*



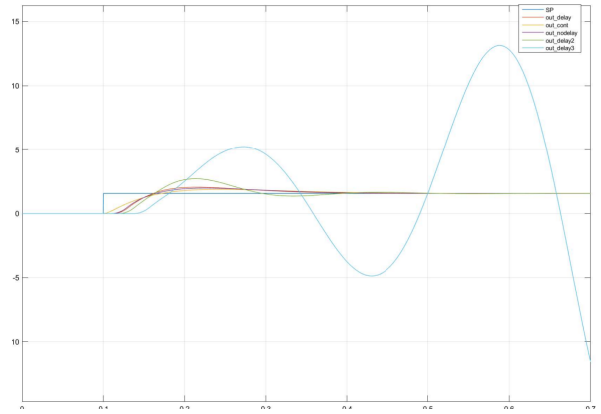
**a** *Señal error. Sistema continuo (rojo), sistema discretizado sin retardo (amarillo), sistema discretizado con retardo $\tau_1$ (azul), sistema discretizado con retardo $\tau_2$ (morado), sistema discretizado con retardo $\tau_3$ (verde).*



**b** *Salida de sistema continuo (azul), Salida del sistema discretizado previo al muestreo(rojo), salida del sistema discretizado posterior al muestreo(amarillo).*



**c** *Señal control sistema continuo (rojo), sistema discretizado sin retardo (amarillo), sistema discretizado con retardo $\tau_1$ (azul), sistema discretizado con retardo $\tau_2$ (morado), sistema discretizado con retardo $\tau_3$ (verde).*



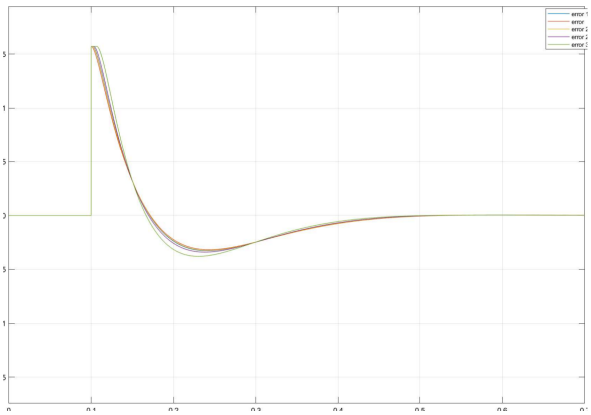
**d** *Señal salida y referencia (azul),sistema continuo (amarillo), sistema discretizado sin retardo (morado), sistema discretizado con retardo $\tau_1$ (rojo), sistema discretizado con retardo $\tau_2$ (verde), sistema discretizado con retardo $\tau_3$ (azul claro).*

**Figura 4:** *Comparativa de señales del sistema a controlar. (a) Señal de error del sistema. (b) Señal de salida del sistema que muestra el efecto del muestreo. (c) Señal de control del sistema. (d) Señal de salida del sistema que muestra efecto del retardo.*

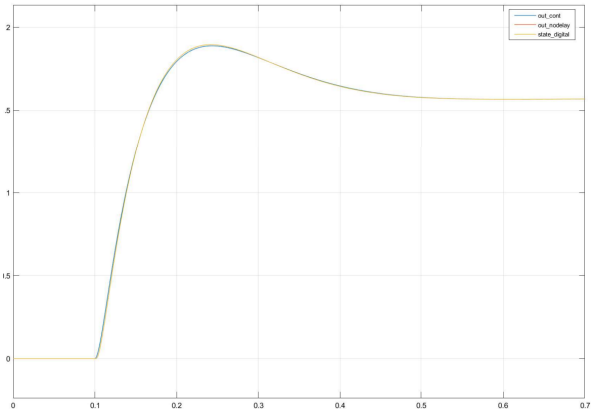
*En mi maquina se ve como usando un D/A ZOH si se ve el muestreado pero no lo que está puesto de normal, tengo duda con esto*

En las gráficas de la Figura 4 se observan los efectos que tienen el periodo de muestreo y el retardo en las diferente señales del sistema. Por un lado, las Figuras 4a y 4d muestran como a medida que aumenta el retardo de computación y el de realimentación el sistema tarda más en estabilizarse, hasta llegar al punto en que con $\tau_3$ se desestabiliza completamente, lo que termina descontrolando el sistema. Esto sucede debido a que las señal de control del sistema es tan lenta que no puede adaptarse a la dinámica del sistema. Por otro lado, también se ve el efecto del periodo de muestreo en la señal de control la cual se muestra muy escalonada.

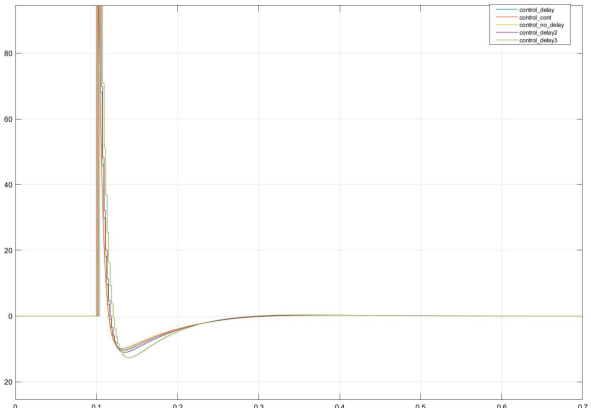
**Análisis de señales de salida con el controlador $\mathbf{T_{s2}}$.** Repitiendo la simulación con periodo de muestreo $T_{s2}$ obtenemos lo siguiente:



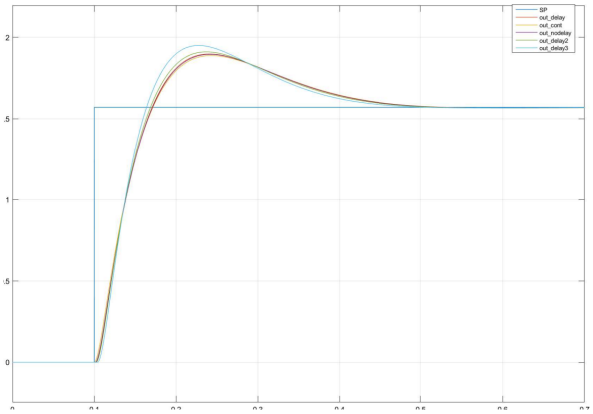
**a** *Señal error. Sistema continuo (rojo), sistema discretizado sin retardo (amarillo), sistema discretizado con retardo $\tau_1$ (azul), sistema discretizado con retardo $\tau_2$ (morado), sistema discretizado con retardo $\tau_3$ (verde).*



**b** *Salida de sistema continuo (azul), Salida del sistema discretizado previo al muestreo(rojo), salida del sistema discretizado posterior al muestreo(amarillo).*



**c** *Señal control sistema continuo (rojo), sistema discretizado sin delay (amarillo), sistema discretizado con retardo $\tau_1$ (azul), sistema discretizado con retardo $\tau_2$ (morado), sistema discretizado con retardo $\tau_3$ (verde).*



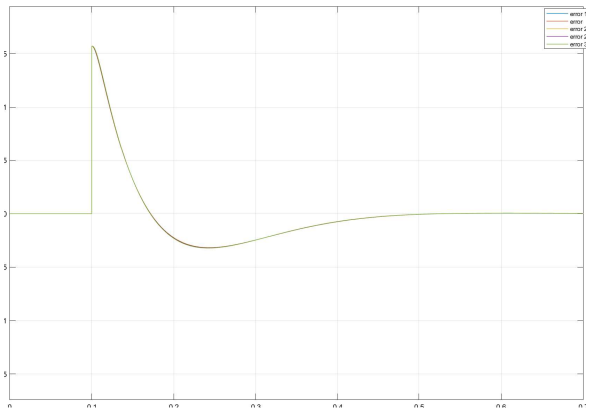
**d** *Señal salida y referencia (azul),sistema continuo (amarillo), sistema discretizado sin retardo (morado), sistema discretizado con retardo $\tau_1$ (rojo), sistema discretizado con retardo $\tau_2$ (verde), sistema discretizado con retardo $\tau_3$ (azul claro).*

**Figura 5:** *Comparativa de señales del sistema a controlar. (a) Señal de error del sistema. (b) Señal de salida del sistema que muestra el efecto del muestreo. (c) Señal de control del sistema. (d) Señal de salida del sistema que muestra efecto del retardo.*

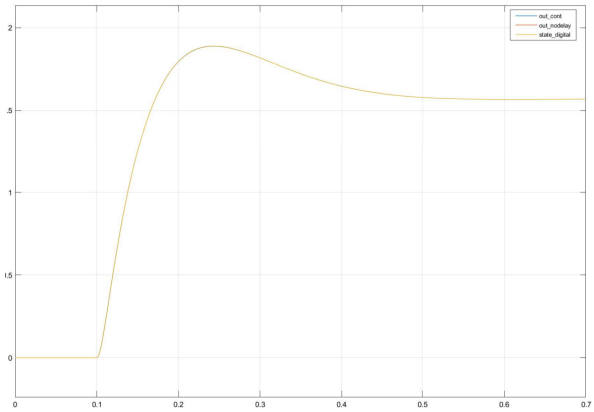
En las gráficas que se muestran en la Figura 5 se ve que evitamos el problema de desestabilización al que nos hemos enfrentado en el caso previo. Además, los errores en discreto y en continuo son mucho más próximos en cualquiera de los 3 casos de retardo comparado con el muestreo de $T_{s1}$. Por otro lado la señal de control es mucho mas suave y se aproxima mucho más al continuo aunque el retardo aumente hasta el caso de $\tau_3$. Por último, el aumento del retardo se ve reflejado como un aumento en el rebose pero aun así, sigue siendo un sistema estable.

**Análisis de señales de salida con el controlador $\mathbf{T_{s3}}$** Tras simular con el diagrama de Simulink que se nos ha proporcionado en la documentación, obtenemos el resultado

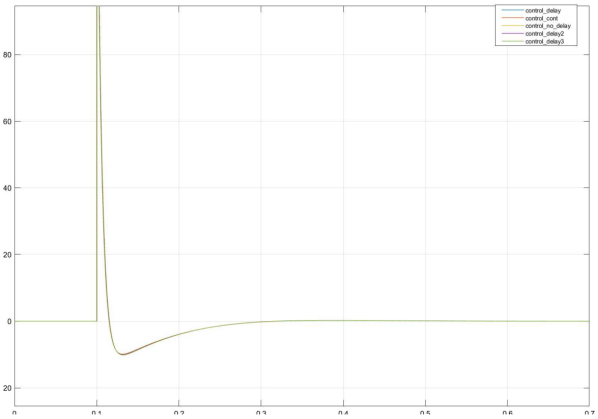
que se muestra en la Figura 6.



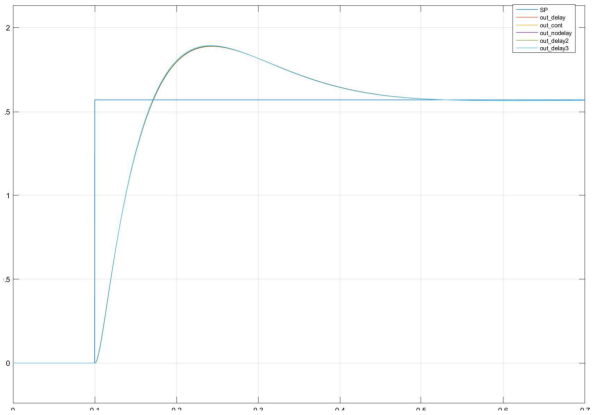
**a** *Señal error sistema continuo (rojo), sistema discretizado sin retardo (amarillo), sistema discretizado con retardo $\tau_1$ (azul), sistema discretizado con retardo $\tau_2$ (morado), sistema discretizado con retardo $\tau_3$ (verde).*



**b** *Salida de sistema continuo (azul), Salida del sistema discretizado previo al muestreo(rojo), salida del sistema discretizado posterior al muestreo(amarillo).*



**c** *Señal control sistema continuo (rojo), sistema discretizado sin delay (amarillo), sistema discretizado con retardo $\tau_1$ (azul), sistema discretizado con retardo $\tau_2$ (morado), sistema discretizado con retardo $\tau_3$ (verde).*



**d** *Señal salida y referencia (azul),sistema continuo (amarillo), sistema discretizado sin retardo (morado), sistema discretizado con retardo $\tau_1$ (rojo), sistema discretizado con retardo $\tau_2$ (verde), sistema discretizado con retardo $\tau_3$ (azul claro).*

**Figura 6:** *Comparativa de señales del sistema a controlar. (a) Señal de error del sistema. (b) Señal de salida del sistema que muestra el efecto del muestreo. (c) Señal de control del sistema. (d) Señal de salida del sistema que muestra efecto del retardo.*

En este último caso, que se muestra en la Figura 6 apenas se nota diferencia del sistema continuo a los diferentes sistemas discretos, lo que indica que el sistema digital responde muy bien a retardos de computación y realimentación con una frecuencia de muestreo muy alta. Sin embargo, no podemos olvidar que lo más normal suele ser que estemos condicionados por unos elemento de adquisición que tienen un límite de frecuencia de muestreo. Un mayor número de datos también implicará que el controlador deberá hacer muchas más operaciones, lo que supone un mayor esfuerzo computacional.

En conclusión, este análisis deja claro que cuando el retraso es grande en comparación con la frecuencia de muestreo y el número de muestras es muy pequeño, el controlador no es capaz de reaccionar adecuadamente a los cambios del sistema, lo cual dificulta la estabilización de la señal y en algunos casos extremos hasta lo desestabiliza. Por otro lado,

vemos que tener una frecuencia de muestreo muy alta puede dar resultados muy buenos y evita los problemas producidos por un retraso con $f_s$ pequeña, pero, a su vez, genera dos nuevos problemas. El primero es el aumento del coste computacional y la necesidad de una elección de instrumentos de adquisición de mayor calidad. El segundo está relacionado con el tiempo de procesado lógico. En función de la complejidad del algoritmo, no podremos pasar de ciertas frecuencias. Si sobrepasáramos este límite se producirían *overruns* lo que daría lugar a errores al no ser capaz de dar una respuesta de control a tiempo en respuesta a la entrada. Lo óptimo es buscar un compromiso entre estos casos como sería el que nos da $f_{s2}$, este es robusto ante el retardo de computación y no supone un coste tan grande.

## 1.2. Ejercicio 2

El archivo `simulink_discrete_3` contiene los bloques para la ejecución del segundo ejercicio de la tarea `Lab01_1`.

**a)** Estos son los valores del periodo de muestreo y del filtro pasobajos:

$$\omega_s = 10 \cdot BW = 341{,}36\, rad/s \tag{6}$$

$$T_{s1} = \frac{2\pi}{\omega_s} = 0{,}0184\, s \tag{7}$$

$$\omega_c = \frac{\omega_s}{2} = 170{,}68\, rad/s \tag{8}$$

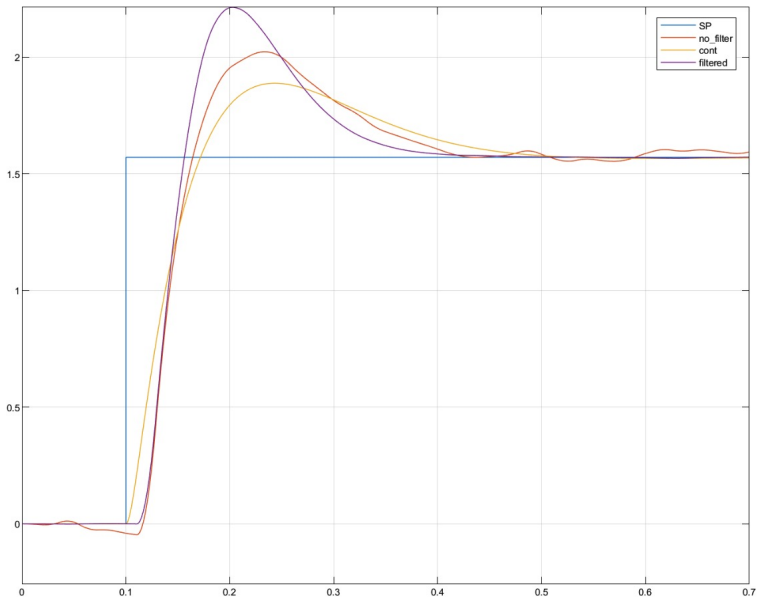



**Figura 7:** *Salida de la planta con el controlador continuo y discreto. En el caso del controlador discreto se ha añadido ruido blanco a la señal de control y en la gráfica se muestran la respuesta de la planta con filtro antialiasing y sin él.*

Con estos valores se puede ver claramente el efecto del filtro y del ruido en la adquisición de los datos en la Figura 7. Con un filtro de orden 1, en la comparativa de la salida con ruido antes y después de ser filtrada, se puede ver que la salida pasa a ser más limpia y estable pero también aumenta considerablemente el rebose. Además, como se muestra en la Figura 8, la señal de control se vuelve más estable una vez se le aplica el filtrado.

Cabe destacar que a mediada que se agranda el orden del filtro, el sobreimpulso en la salida aumenta. Sin embargo, esta frecuencia de muestreo no parece suficiente para un control preciso. Teóricamente, si se añade un filtro *antialiasing* al sistema, la frecuencia de este debería ser 10 veces mayor que el ancho de banda en lazo cerrado y la frecuencia de muestreo debería ser 10 veces mayor que la frecuencia de corte del filtro, es decir, la frecuencia de muestreo debería ser 100 veces mayor que el ancho de banda del sistema en lazo cerrado. Para el caso a) no se cumple ninguna de las dos.

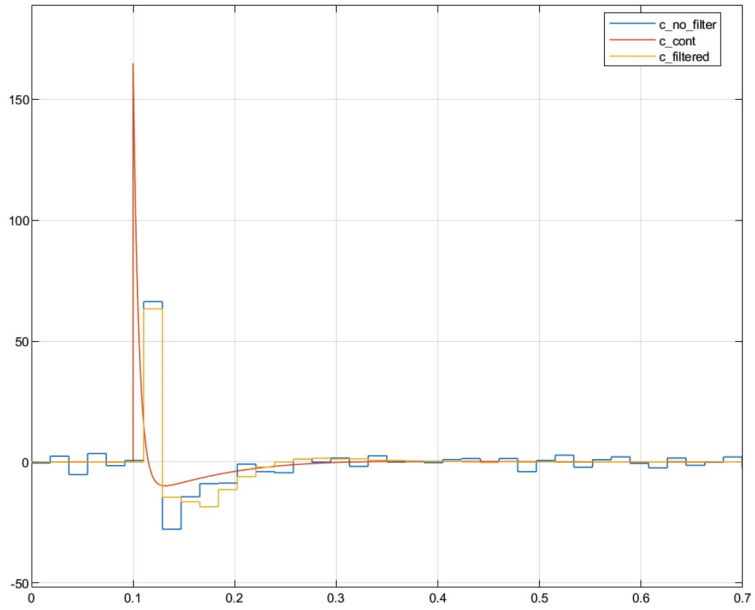


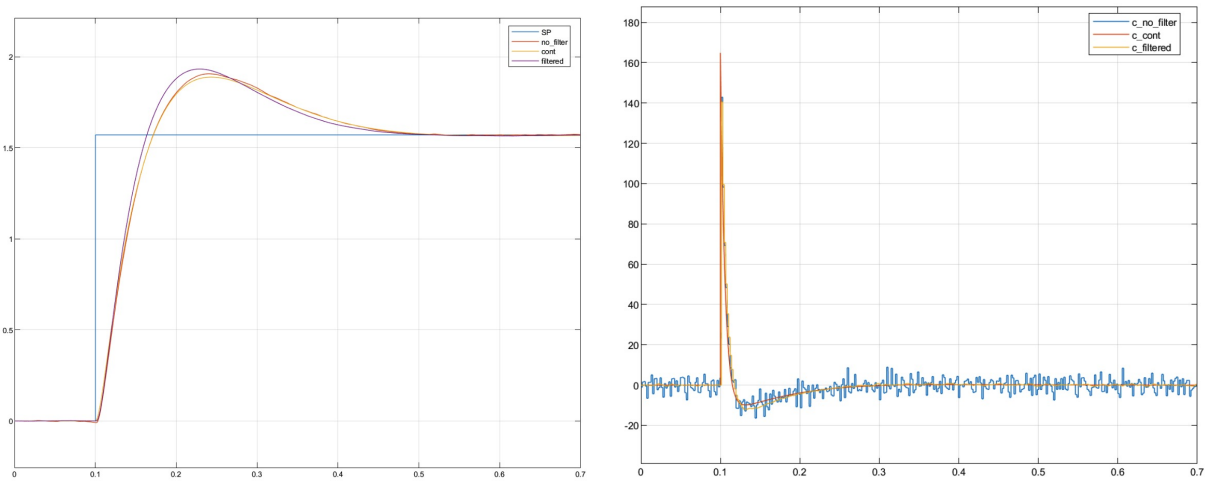

**Figura 8:** *Comparativa de señales de control para controlador continuo y discreto para periodo de muestreo $T_{s1}$. El controlador discreto se ha muestreado con periodo $T_{s1}$ y se muestra con y sin filtro antialiasing.*

Por otro lado, con el bloque **Bode Plot** se puede comprobar que efectivamente hay una pérdida de ganancia de $3dB$ en $\omega_c/2$. Además, también puede apreciarse cómo aumenta el desfase añadido a la señal a medida que aumenta el orden del filtro.

**b)** Esta es la nueva frecuencia de corte del sistema:

$$\omega_c = 10BW = 341,36\, rad/s \tag{9}$$

Con esta nueva frecuencia de corte, el filtro se ajusta a lo establecido teóricamente para el correcto funcionamiento de un filtro antialiasing. Esto significa que la frecuencia de muestreo debería ajustarse acorde con la de corte, es decir, la nueva frecuencia de muestreo será como mínimo 2 veces la frecuencia de corte del filtro para cumplir el criterio de Nyquist. No obstante, para mantener un diseño más conservador hemos decidido mantener una relación de frecuencia de muestreo 10 veces mayor a la del corte del filtro. De esta manera minimizamos el efecto de ralentización del filtro, por lo tanto, la frecuencia de muestreo es 100 veces el ancho de banda ($f_s = 3413,6$ rad/s). Con esta nueva frecuencia de muestreo, la respuesta del sistema mejora considerablemente y el sobreimpulso que sufría antes se ve muy reducido tanto para la señal filtrada como sin filtrar, tal y como se muestra en la Figura 9a.

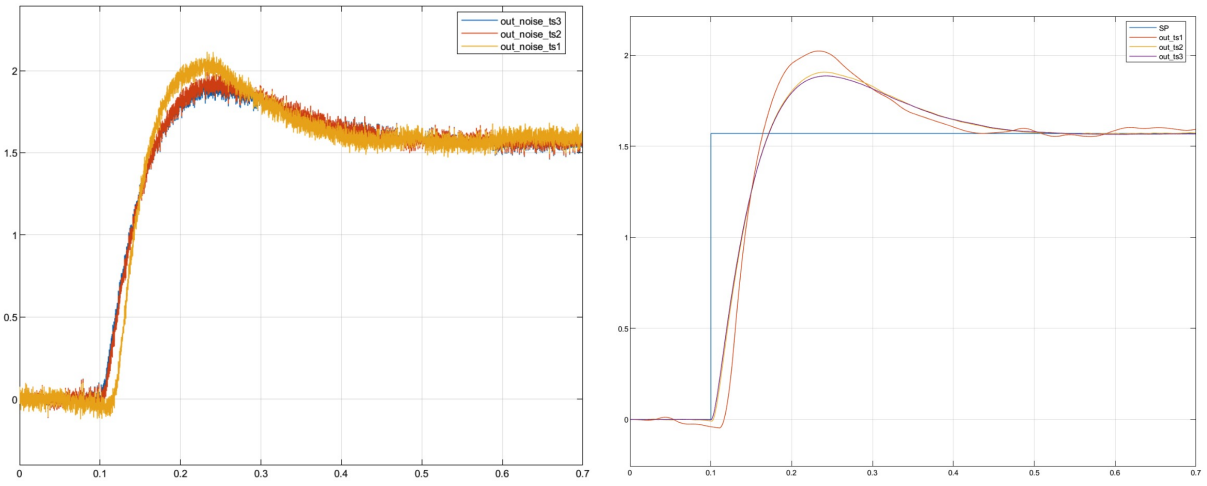



**a** **b**

**Figura 9:** *(a) Salida de la planta con el controlador continuo y discreto para un periodo de muestreo* 10 *veces menor. Se muestra la señal filtrada y sin filtrar. (b) Señal de control para sistema continuo y discreto con periodo de muestreo* $T_{s2}$*.*

La mejora también se ve reflejada en la señal de control que se ve en la Figura 9b, la cual se hace más estable y alcanza un máximo mucho más cercano al que alcanza el controlador continuo. Sin embargo, su respuesta sigue siendo ligeramente más lenta que la del controlador continuo, lo cual explicaría el sutil desfase que hay entre la salida continua y discreta.

**c)** En el apartado anterior se ha ajustado el periodo de muestreo $T_{s1}$ hasta hacerlo igual a $T_{s2}$ para poder ajustarse correctamente a los efectos del filtro. Si se retira el filtro anti-aliasing, la frecuencia de muestreo recomendada vuelve a ser 10 veces el ancho de banda del sistema. Por lo que tanto, tanto $T_{s2}$ como $T_{s3}$ serían considerados como periodos de sobremuestreo en este caso.




**a** **b**

**Figura 10:** *(a) Salida de la planta en lazo cerrado sin filtrar y con ruido blanco. Se muestra la señal filtrada y sin filtrar. (b) Salida de la planta en lazo cerrado sin filtrar pero sin ruido. La estabilidad aumenta cuando se reduce el periodo de muestreo.*

Analizando primero la salida con ruido del sistema de lazo cerrado de la Figura 10a, se ve que a medida que se reduce el periodo de muestreo, la respuesta del sistema se acelera y disminuye considerablemente el rebose de este. Puede verse esta misma tendencia en la salida de la planta de la Figura 10b. Aunque también muestra otra importante diferencia, a medida que aumenta la frecuencia de muestreo, la salida de la planta se vuelve más inmune al ruido, que da como resultado una señal más limpia y estable.

# 2. Laboratorio 1.2

La sesión de laboratorio `Lab01_2` se centra en la programación directa de controladores. El objetivo es simplificar y reestructurar la programación de los controladores para tener así respuestas más rápidas y más similares a la respuesta que se obtiene en continua. Este laboratorio consta de dos ejercicios: en el primero se deben programar cuatro formas directas de filtros orden 2 y después se debe simular su comportamiento para verificar que funcionan como se espera que lo hagan; en el segundo ejercicio se parte del diagrama de Simulink del laboratorio anterior y se sustituyen los controladores discretos utilizados por controladores escritos con formas directas.

## 2.1. Ejercicio 1

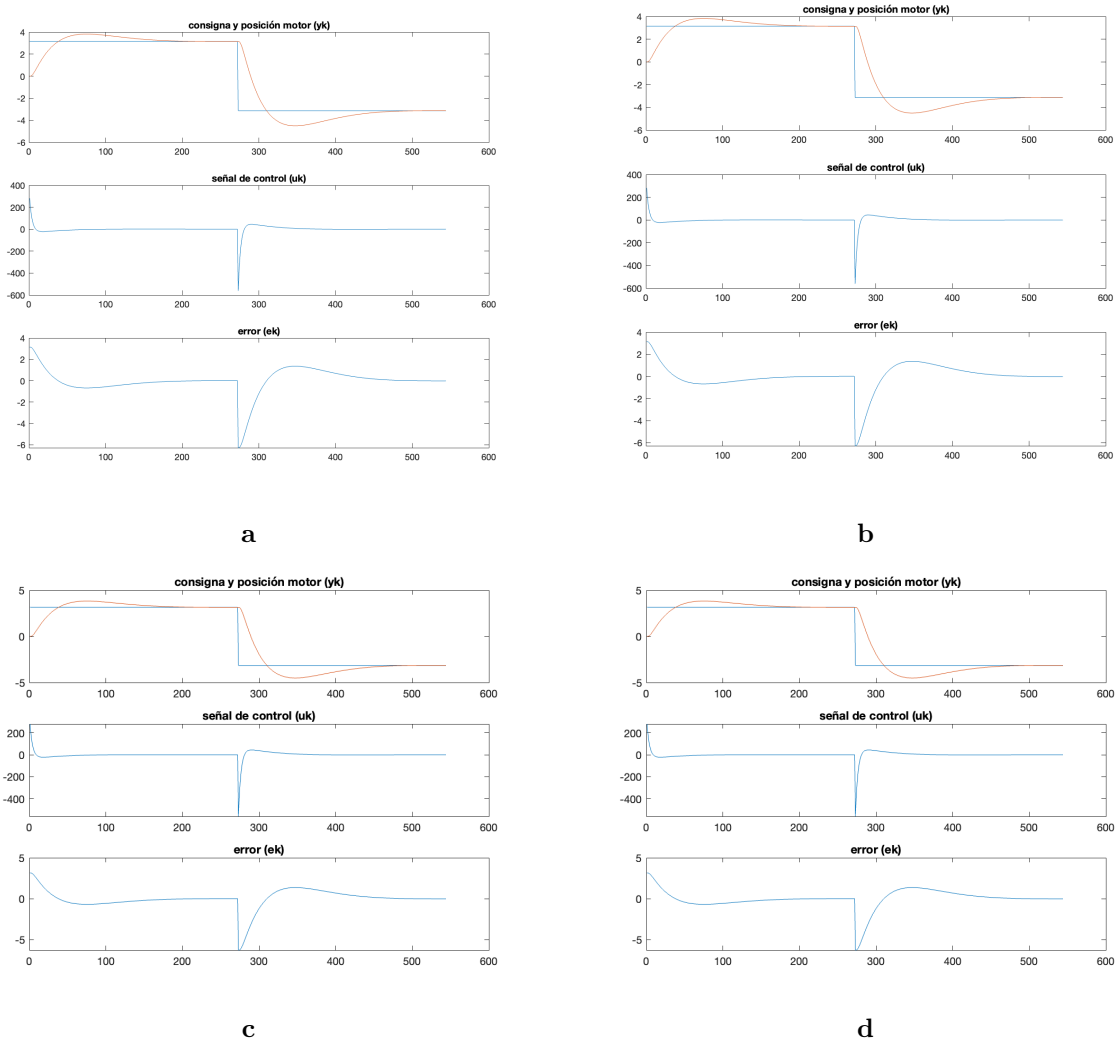



**Figura 11:** *Comparativa que muestra los resultados obtenidos con las cuatro formas directas. La salida de la planta es idéntica en los cuatro casos.*

En el script `controlador_directo` se ha completado el código para simular el controlador en forma directa 2D y se ha añadido el código que faltaba para calcular la salida de los controladores en forma directa 3D y 4D. Si se simula la respuesta de la planta con los cuatro tipos de controladores directos que se han programado utilizando el script

`control_float_test`, se consiguen las gráficas que se muestran en la Figura 11. Estas gráficas muestran como el resultado es idéntico independientemente de qué técnica de programación directa se use.

## 2.2. Ejercicio 2

Partiendo del archivo `simulink_discrete_2` de `Lab01_1`, se eliminan los bloques controladores discretos y se sustituyen por un bloque que implemente el controlador en forma directa. La primera implementación de formas directas se hará mediante un bloque de *Matlab Function* que contiene el código de la forma directa que se vaya a usar. Se hace la prueba con las formas directa 1D y 3D del diagrama que se muestra en la Figura 12.

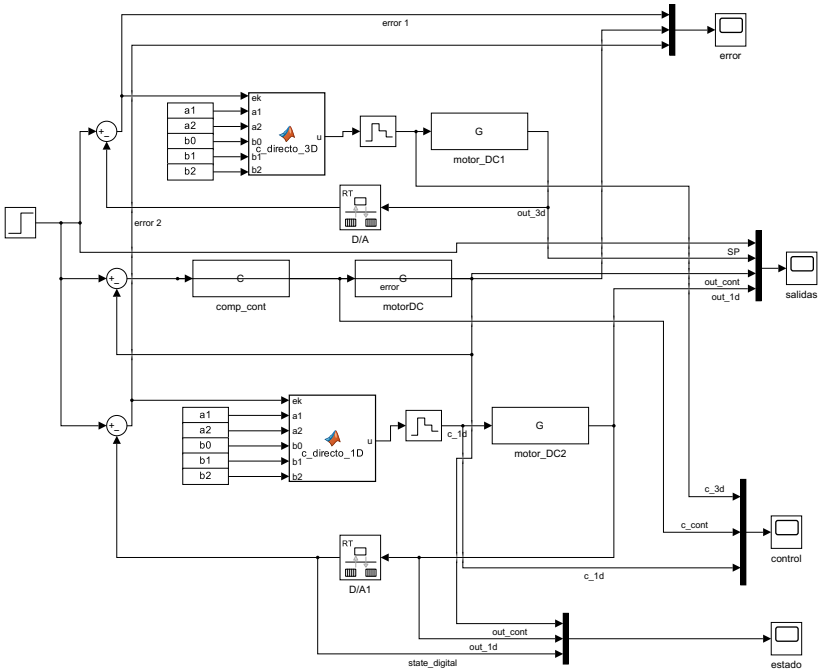



**Figura 12:** *Diagrama de Simulink que calcula la respuesta del sistema continuo y discretizado utilizando las formas directas 1D y 3D. Las formas directas se han implementado escribiéndolas en un bloque de Matlab Function.*

Utilizando el diagrama de bloques de la Figura 12 el sistema responde con la salida que se muestra en la Figura 13. Como puede verse, la respuesta del sistema es prácticamente idéntica en los 3 casos, como podría esperarse al implementar el mismo controlador. Aun así, se percibe un pequeño retardo del sistema discreto respecto al continuo y el rebose es también ligeramente menos pronunciado. Hemos comprobado que aumentando la frecuencia de muestreo este *gap* disminuye así que creemos que es debido a la frecuencia de muestreo. La Figura 13b muestra la comparativa entre la salida en continua y la salida con los controladores discretizados con las formas directas 1D Y 3D. Ampliando la imagen, se puede ver la diferencia es mínima. La salida de los controladores discretizados es tan similar que nos son diferenciables porque se muestra una encima de la otra.

La segunda implementación se hace interconectando directamente bloques de Simulink que formen la estructura de la forma directa. Para poder usar el diagrama de bloques de Simulink es necesario ejecutar primero el script `control_float_test`, que ha sido modificado para que cargue los parámetros del controlador discretizado directamente al

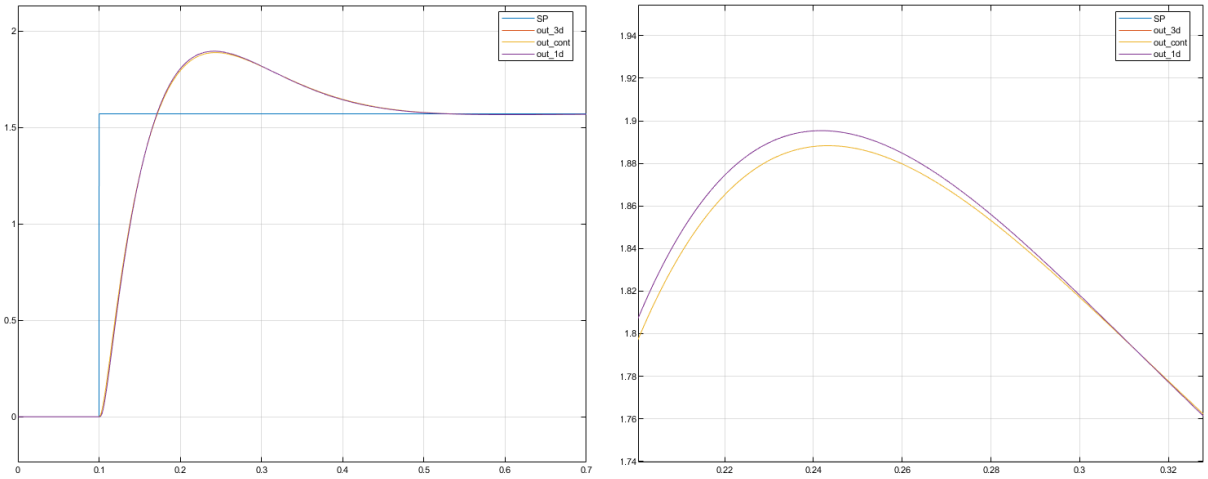



**Figura 13:** *Comparativa de la salida de la planta con controlador continuo y discretizado con forma directa 1D y 3D, implementado en bloque Matlab Function. (a) Salida de la planta* zoom out*. (b) Salida de la planta* zoom in*.*

*workspace*. Esta modificación se ha hecho para que los bloques de Matlab Function de los controladores en forma directa puedan tomar los parámetros de la workspace utilizando su máscara. La Figura 14 muestra el diagrama que se ha implementado en esta segunda mitad.

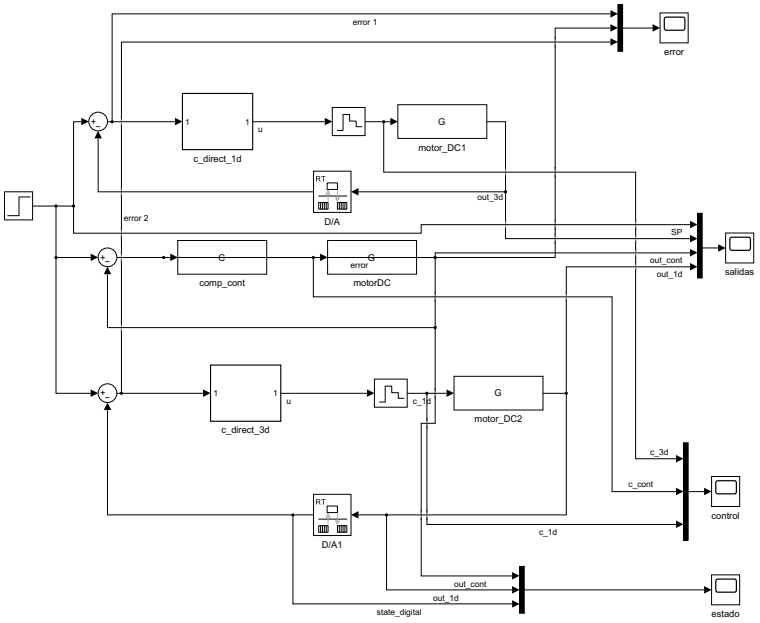
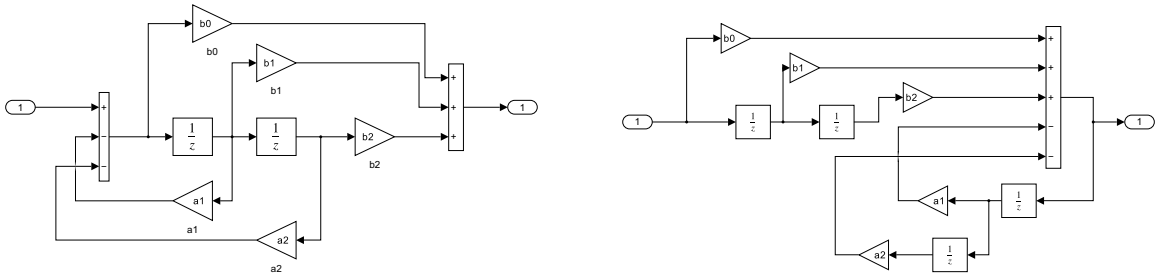



**Figura 14:** *Diagrama de Simulink que calcula la respuesta del sistema continuo y discretizado utilizando las formas directas 1D y 3D. Las formas directas se han implementado en un subsistema de bloques de Simulink.*

Dentro de los bloques `c_direct_1d` y `c_direct_3d` se encuentran los diagramas que se muestran en la Figura 15. Estos diagramas de bloques se han construido haciendo uso de los bloques de ganancia, retraso unitario y suma. Las formas directas se han implementado

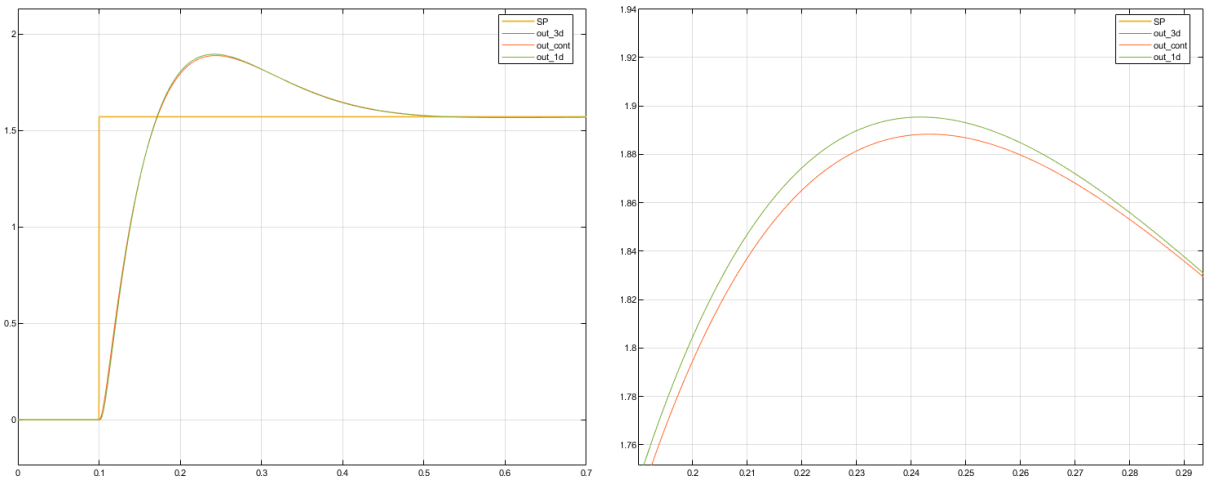
particularizando las definiciones canónicas para el caso de $n = 2$.




**a** **b**

**Figura 15:** *(a) Diagrama de bloques de controlador directo 1D. (b) Diagrama de bloques de controlador directo 3D.*

Utilizando el sistema de la Figura 14 se obtiene la respuesta en lazo cerrado de la Figura 16. Como se ve en la imagen, la salida de la planta es aparentemente idéntica a la que se muestra en la Figura 13. Lo que indica que la salida de la planta que se obtiene utilizando la implementación con Matlab Function comparada con la que se obtiene implementando sobre bloques de Simulink no tiene ninguna diferencia apreciable.




**a** **b**

**Figura 16:** *Comparativa de la salida de la planta con controlador continuo y discretizado con forma directa 1D y 3D, implementado directamente sobre bloques de Simulink. (a) Salida de la planta* zoom out*. (b) Salida de la planta* zoom in*.*

# 3. Laboratorio 1.3

En esta tercera practica trataremos de comprobar las ventajas que ofrece el uso de operadores $\delta$ en la discretización de controladores.

## 3.1. Ejercicio 1

En este primer ejercicio de la practica escribiremos una función de MatLab la cual haga una transformación de función de transferencia continua a $\delta$. La función **c2delta** debe tener como entradas sys, Ts y method los cuales indican el sistema a transformar, el periodo $T_s$ de discretización y el método transformación respectivamente.

Para conseguir los coeficientes de los parámetros de forma analítica hemos calculado los parámetros que obtendríamos una vez transformada la función de transferencia en función de los parámetros de la función inicial, es decir, si la **TF(s)** está parametrizada por $(a_0, ..., a_n, b_0, ..., b_n)$ y la **TF($\delta$)** está parametrizada por $(a_1, ..., a_n, b_0, .., b_m)$ reescribiremos los parámetros de **TF($\delta$)** de la siguiente manera:

$$TF(\delta, ad_1(a_0, ..., b_n), ..., ad_n(a_0, ..., b_n), bd_0(a_0..., b_n), ..., b_m(a_0, ..., b_n))$$

Los coeficientes que hemos conseguido para el método simple para un sistema de orden 2 son:

$$bd_2 = \frac{4b_2T^2}{4 + 2a_1T + a_2T^2}, \qquad bd_1 = \frac{4b_1T + 4b_2T^2}{4 + 2a_1T + a_2T^2}, \qquad bd_0 = \frac{4b_0 + 2b_1T + b_2T^2}{4 + 2a_1T + a_2T^2}$$
$$ad_2 = \frac{4a_2T^2}{4 + 2a_1T + a_2T^2}, \qquad ad_1 = \frac{4a_1T + 4a_2T^2}{4 + 2a_1T + a_2T^2}$$

Los coeficientes que hemos conseguido para el método clásico para un sistema de orden 2 son:

$$bd_2 = \frac{16b_0}{a_2T^4 + 2a_1t^3 + 4T^2}, \quad bd_1 = \frac{16Tb_0 + 4b_1T^2}{a_2T^4 + 2a_1T^3 + 4T^2}, \quad bd_0 = \frac{4T^2b_0 + 2b_1T^3 + b_2T^4}{a_2T^4 + 2a_1T^3 + 4T^2},$$
$$ad_2 = \frac{16}{a_2T^4 + 2a_1T^3 + 4T^2}, \quad ad_1 = \frac{16T + 4a_1T^2}{a_2T^4 + 2a_1T^3 + 4T^2}.$$

De manera que la función de transferencia en delta se escribirá como se indica en la siguiente ecuación:

$$C(\delta) = \frac{bd_2\delta^{-2} + bd_1\delta^{-1} + bd_0}{ad_2\delta^{-2} + ad_1\delta^{-1} + 1} \tag{10}$$

De la misma manera debemos crear otra función equivalente que transforme en este caso de Z a $\delta$. Repitiendo el mismo proceso que para la función anterior obtenemos las siguientes relaciones entre parámetros:

$$b_{2d} = b_0 + b_1 + b_2, \qquad b_{1d} = 2b_0 + b_1, \qquad b_{0d} = b_0,$$
$$a_{2d} = 1 + a_1 + a_2, \qquad a_{1d} = 2 + a_1$$

y para el caso clásico vemos que es parecido al caso simple pero en vez de tener una relación de $z$ a $\delta$ de la siguiente forma $z = 1 + \delta$ tenemos una de la siguiente manera $z = 1 + \delta T$ con lo que obtenemos los siguientes coeficientes:

$$b_{2d} = \frac{b_0 + b_1 + b_2}{T^2}, \qquad b_{1d} = \frac{2b_0 + b_1}{T}, \qquad b_{0d} = b_0,$$
$$a_{2d} = \frac{1 + a_1 + a_2}{T^2}, \qquad a_{1d} = \frac{2 + a_1}{T}.$$

En la Figura 17 se muestran los diagramas de Simulink para modelizar el retardo unitario utilizando la transformada $\delta$ en su variante simple y en su variante clásica.

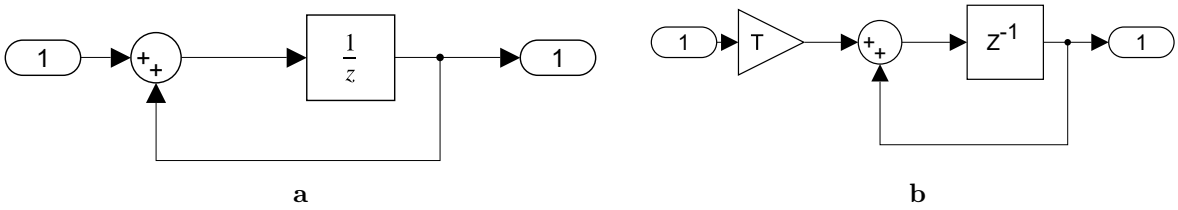



**Figura 17:** *Diagramas de Simulink de retardos unitarios de transformada $\delta$. (a) Simple. (b) Clásico.*

## 3.2. Ejercicio 2

El objetivo de este ejercicio es calcular los coeficientes para controladores discretizados con periodo de muestreo $Ts = 1$ms para posteriormente poder compararlos. Haciendo uso de las funciones creadas en el ejercicio anterior obtenemos los siguientes coeficientes reunidos en la Tabla 3:

| $\mathbf{C_z}$ | ***double* precision** | $\mathbf{C_\delta}$ | ***double* precision** |
|---|---|---|---|
| $b_{0d}$ | 95.911290909090910 | $p$ | 95.911290909090920 |
| $b_{1d}$ | -190.9086909090909 | $qd1$ | 0.913890909090909 |
| $b_{2d}$ | 94.998199999999980 | $rd_1d_2$ | 0.000800000000000 |
| $a_{1d}$ | -1.818181818181818 | $d_1$ | 0.181818181818182 |
| $a_{2d}$ | 0.818181818181818 | $d_2$ | 0.000000000000000 |

**Tabla 3:** *Coeficientes del controlador Z y del controlador $\delta$ con precisión de máquina.*

Es destacable que los coeficientes de la función $\delta$ han sido renombrados para su uso en un controlador futuro, el cual se indica en la Figura 19b. Como se puede observar en los coeficientes obtenidos para la transformación Z, los parámetros tienen parte entera mucho mayor que los de la transformada $\delta$, lo cual, de cara a procesamiento digital de controladores con lógica de coma fija puede ser perjudicial. Esto nos obliga a usar un mayor número de bits para la parte entera de los parámetros. En cambio, con el uso de transformada delta obtenemos números que mayormente son decimales, lo que implica que podemos dedicar una mayor cantidad de bits para procesar la parte decimal de los parámetros. Este mejor uso de longitud de palabra puede hacer que nos ahorremos mucha lógica.

## 3.3. Ejercicio 3

Para probar los controladores Z y $\delta$, se ha tomado el esquema de Simulink de `Lab01_2` y se ha modificado para sustituir los controladores en forma directa que se usaron en esa práctica por los dos controladores que se van a usar, dando como resultado el diagrama de la Figura 18. Por un lado, se ha implementado un controlador discretizado en Z en forma directa 1D utilizando las indicaciones del anexo de la documentación, que se muestra en la Figura 19a. Por otro lado, también se ha implementado un controlador discretizado con la transformada delta que se muestra en la Figura 19b. Ambos controladores se han construido directamente sobre bloques de Simulink; el primero utiliza los coeficientes del controlador discretizado con Tustin, mientras que el segundo utiliza los coeficientes de la forma canónica 1D modificada, además de los bloques de retraso unitario de la transformada delta.

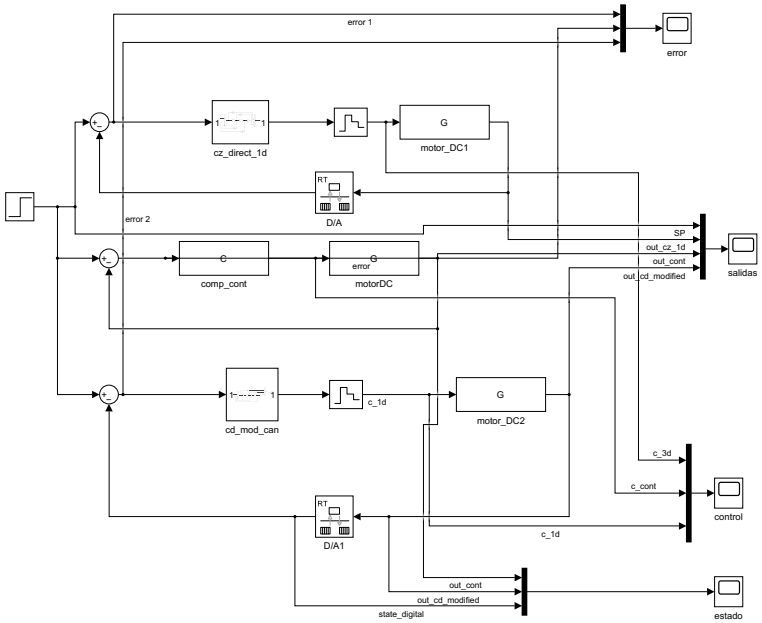
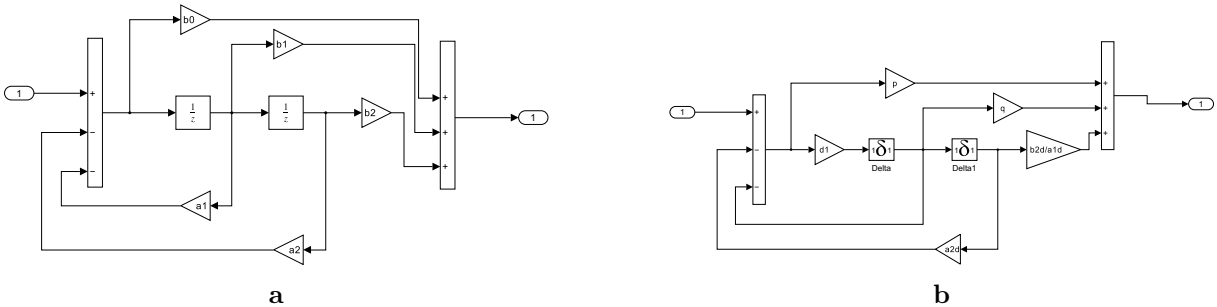



**Figura 18:** *Diagrama de bloques que se ha utilizado para hacer las simulaciones con el controlador discretizado en Z con Tustin y con el controlador discretizado en $\delta$.*



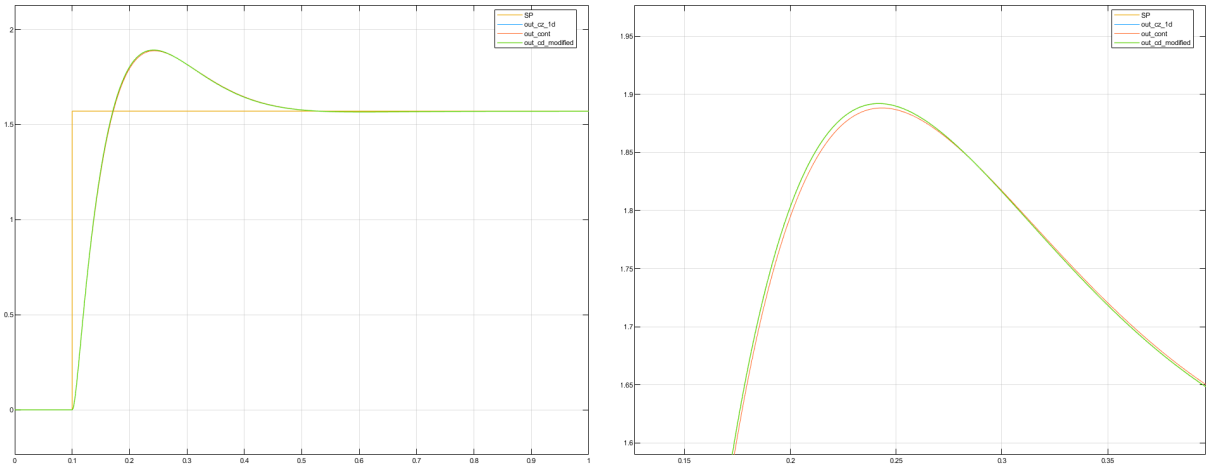

**a** **b**

**Figura 19:** *(a) Diagrama de Simulink del controlador Z en forma directa 1D. (b) Diagrama de Simulink del controlador $\delta$ construido de la forma canónica modificada.*

Debido al valor nulo del coeficiente $d_2$ de la transformada delta, para no cambiar el controlador de segundo orden a implementar hemos tenido que hacer una modificación

en la distribución del diagrama, el cual se indica en la Figura 19b. De esta forma, la modificación se aplica solo en el lado izquierdo del diagrama mientras que el lado derecho mantiene una forma más parecida a la 1D original. El resultado obtenido, manteniendo la disposición del anexo, devuelve una respuesta diferente como se muestra en la Figura 20.

Haciendo uso de sendos controladores se ha calculado la salida de la planta en lazo cerrado con periodo de muestreo igual a $T_s = 1\,\text{ms}$. El resultado obtenido se muestra en las gráficas de la Figura 20. Como se puede ver, tanto el controlador discretizado en Z dispuesto en un bloque de función de transferencia como con la forma directa 1D y el controlador en $\delta$ dan un resultado idéntico al controlador continuo.




**a** **b**

**Figura 20:** *Salida de la planta en lazo cerrado para controlador continuo, controlador discretizado en Z con forma directa 1D y controlador discretizado en $\delta$ con forma canónica modificada adaptada. (a) Salida de la planta* zoom Out*. (b) Salida de la planta* zoom In*.*

## 3.4. Ejercicio 4

Para obtener los polos de los compensadores con la precisión de máquina usamos la función **pole**,

| **Controlador Z** | **Controlador $\delta$** |
|---|---|
| 1,000 | 0,000 |
| 0,8182 | −0,1818 |

**Tabla 4:** *Polos de los controladores Z y $\delta$.*

El controlador Z tiene un polo sobre la circunferencia de radio 1, lo que lo hace críticamente estable. De la misma manera el controlador $\delta$ tiene un polo en 0 lo que lo hace críticamente estable en el plano delta también. En ambos casos los otros polos están dentro de sus respectivas regiones de convergencia.

A continuación, se limita la precisión de las representaciones de los coeficientes de los controladores primero a 16 bits y después a 12 bits. En la Tablas 5 y 6 se muestran las representaciones resultantes de limitar la cantidad de bits que se usan para expresar los coeficientes, estas han sido leídas del terminal de simulink.

| $\mathbf{C_z}$ | ***double* precision** | **16 bits** | **12 bits** |
|---|---|---|---|
| $b_0$ | 95.911290909090910 | 95.910156250000000 | 95.937500000000000 |
| $b_1$ | -190.9086909090909 | -190.906250000000000 | -190.875000000000000 |
| $b_2$ | 94.998199999999980 | 95.000000000000000 | 95.000000000000000 |
| $a_1$ | -1.818181818181818 | -1.818176269531250 | -1.818359375000000 |
| $a_2$ | 0.818181818181818 | 0.818176269531250 | 0.818359375000000 |

**Tabla 5:** *Coeficientes del controlador Z con precisión limitada a precisión de máquina, 16 bits y 12 bits.*

| $\mathbf{C_\delta}$ | ***double* precision** | **16 bits** | **12 bits** |
|---|---|---|---|
| $p$ | 95.911290909090920 | 95.910156250000000 | 95.937500000000000 |
| $q$ | 5.026400000000001 | 5.026367187500000 | 5.027343750000000 |
| $r$ | 0.004400000000000 | 0.004394531250000 | 0.004882812500000 |
| $d_1$ | 0.181818181818182 | 0.181823730468750 | 0.181640625000000 |
| $d_2$ | 0.000000000000000 | 0.000000000000000 | 0.000000000000000 |

**Tabla 6:** *Coeficientes del controlador $\delta$ con precisión limitada a precisión de máquina, 16 bits y 12 bits.*

Construyendo las funciones de transferencia en MATLAB y representando los polos y ceros de estas nuevas funciones de transferencia con precisión limitada se observa una perturbación en la posición de estos en el diagrama polar el cual se muestra en la Figura 21.

Se observa como a medida que se limita la longitud de palabra en z el polo que está en $-0{,}991$ este termina convirtiéndose en un polo complejo conjugado, y cuanto menos es su longitud de palabra mayor es su componente imaginaria. El efecto de esta alteración se puede observar en las Figuras 21a y 21b. En cambio se observa que el controlador $\delta$ se mantiene robusto a la disminución de longitud de palabra.

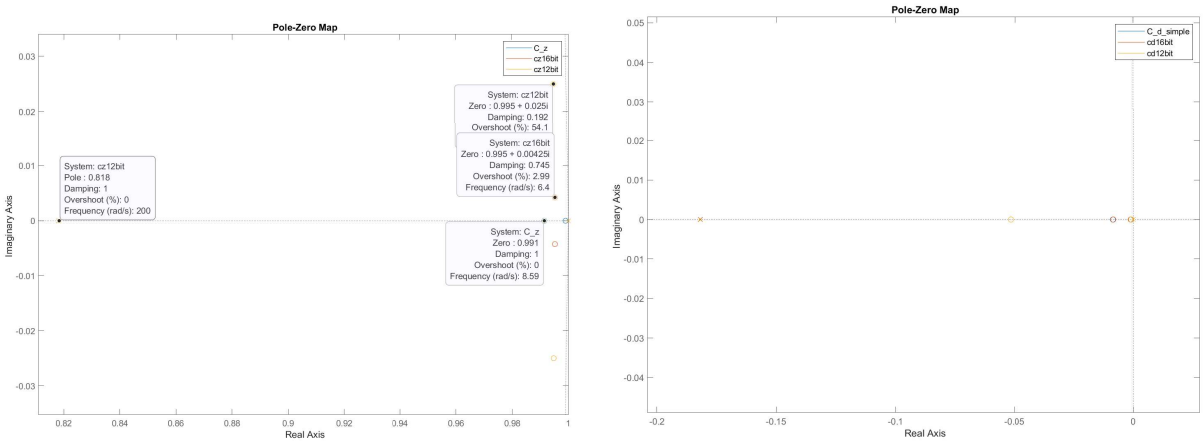
**16 bits:** El efecto de reducir la precisión hasta los 16 bits se hace notorio principalmente en el controlador discretizado en Z, el controlador en $\delta$ no se ve muy afectado. Como se puede ver en los coeficientes de la Tabla 5, esto puede ser debido al rango que toman los coeficientes que tiene cada uno. Mientras que la mayoría de los coeficientes del controlador $\delta$ tienen una parte entera cercana a 0, los coeficientes del controlador Z son relativamente mucho mayores y necesitan más bits para ser expresados adecuadamente, lo que supone una pérdida de bits para expresar la parte fraccional que se hace especialmente notable cuanto mayor sea la parte entera del coeficiente a expresar. A pesar de todo, ambas soluciones son válidas y ofrecen un buen resultado.

| Precisión | Double | 16 bits | 12 bits |
|---|---|---|---|
| Polos | 1 | 1 | 1 |
| | 0.818181818181818 | 0.818176269531250 | 0.818359375000000 |
| Ceros | 0.999024818346034 | 0.995234798191667 + 0.004245123356274 11i | 0.994788273615635 + 0.0249860702420364i |
| | 0.991446679843296 | 0.995234798191667 - 0.004245123356274 11i | 0.994788273615635 - 0.0249860702420364i |

**Tabla 7:** *Valores de posición polares de las polos y ceros de las funciones de transferencia para los controladores Z.*

| Precisión | Double | 16 bits | 12 bits |
|---|---|---|---|
| Polos | 0 | 0 | 0 |
| | -0.181818181818182 | -0.181818181818182 | -0.181823730468750 |
| Ceros | -0.00855332015668545 | -0.00842049787364861 | -0.0515027071677453 |
| | -0.000975181653983919 | -0.00109788504064292 | -0.000889393809453351 |

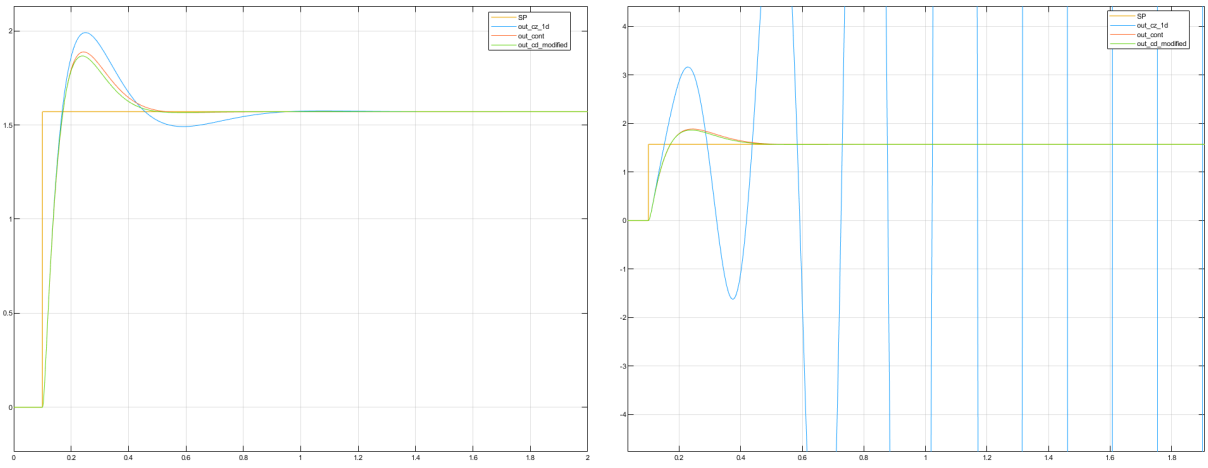
**Tabla 8:** *Valores de posición polares de las polos y ceros de las funciones de transferencia para los controladores δ.*




**a** **b**

**Figura 21:** *Ubicación de polos de las funciones de transferencia de los controladores Z y δ con representaciones de coma flotante y coma fija a 16 y 12 bits.(a) Representaciones de z. (b) Representaciones de δ.*

**12 bits:** Al reducir la precisión hasta los 12 bits, la pérdida de información en el controlador Z es tal que la solución se desestabiliza y se hace oscilatoria. Esto no ocurre con el controlador $\delta$ el cual, como ya se ha explicado, mantiene un alto grado de precisión en sus coeficientes por tener una parte entera mínima. Dado que los coeficientes de $C_z$ tienen una parte entera mayor se ven más afectados y el más afectado de todos es sin duda $b_1$. Por lo tanto, el controlador Z es inservible en 12 bits, mientras que el controlador $\delta$ sigue siendo funcional.




**a** **b**

**Figura 22:** *Salida de la planta en lazo cerrado con las representaciones de los coeficientes de los controladores limitadas. (a) Representaciones de 16 bits. (b) Representaciones de 12 bits.*

## 3.5. Ejercicio 5

En esta tarea se deben comparar los márgenes de fase y de ganancia de los sistemas para ver cómo les afecta la limitación en la representación de los coeficientes. Por ello, en la Tabla 9 se muestran los valores de los márgenes del sistema en lazo abierto tanto para el caso continuo como el discreto con las representaciones de los coeficientes limitadas a 16 y a 12 bits.

| | Discretizado en Z | | | Discretizado en $\delta$ | | |
|---|---|---|---|---|---|---|
| | Double | 16bits | 12bits | Double | 16bits | 12bits |
| Margen de Ganancia (dB) | 38.5 | -15.8 | 9.78 | -3.63 | -3.63 | 9.78 |
| $\omega_{cruce}$ Ganancia (rad/s) | 599 | 5.91 | 26.1 | 48.2 | 48.2 | 26.1 |
| Margen de Fase (deg.) | 62.4 | 60.4 | -58.5 | -0.413 | -0.414 | -58.5 |
| $\omega_{cruce}$ Fase (rad/s) | 23.9 | 22.7 | 19.6 | 59.4 | 59.4 | 19.6 |

**Tabla 9:** *Márgenes de ganancia y fase del sistema en lazo abierto con controlador continuo y discreto. En el caso de los controladores discretos se muestran los datos para la discretización en Z y en $\delta$ con las representaciones de de los coeficientes limitadas.*

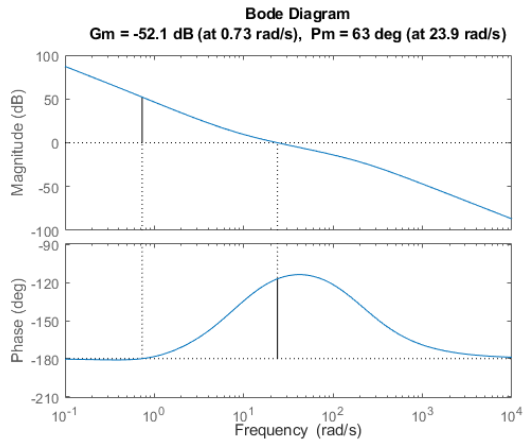
| | **Unidad** | **Continuo** | **Discretizado en Z** | | **Discretizado en $\delta$** | |
|---|---|---|---|---|---|---|
| | | | **16 bits** | **12 bits** | **16 bits** | **12 bits** |
| Margen de Ganancia | dB | -52.10 | -15.80 | 9.78 | -3.63 | -3.63 |
| $\omega_{cruce}$ Ganancia | rad/s | 0.7303 | 5.9082 | 26.1432 | 48.2042 | 48.1949 |
| Margen de Fase | deg. | 63.0401 | 60.4434 | -58.4827 | -0.4132 | -0.4141 |
| $\omega_{cruce}$ Fase | rad/s | 23.8620 | 22.7131 | 19.5665 | 59.5939 | 59.4090 |

Rehacer tabla sustituyendo continuo por discreto con precisión de máquina. Actualizar valores de delta ta,bien.

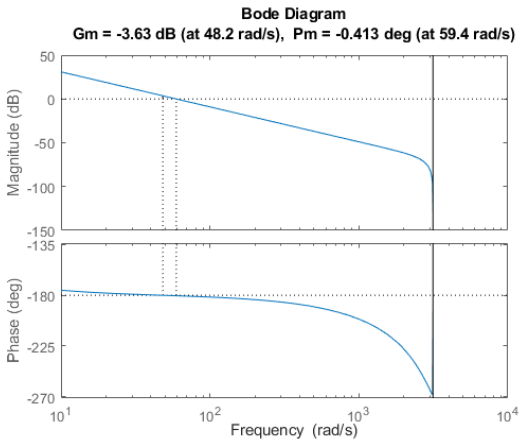
Como muestran los datos de la Tabla 9, de base, el sistema tiene un margen de fase de $-52{,}1\,dB$ y un margen de fase de 63,04°, lo que significa que el sistema es estable. Con el controlador discretizado, el sistema pierde tanto margen de ganancia como de fase sin importar qué tipo de discretización se emplee. Además, estos datos concuerdan con lo que se muestra en la Figura 22b, donde se ve que la respuesta es claramente inestable. Esto cuadra con el margen de ganancia y de fase obtenido para este controlador porque, como es bien sabido, para que el sistema sea estable se necesitan margen de ganancia negativo y margen de fase positivo.

Otro detalle importante son los márgenes para los controladores $\delta$. Ambos tienen un margen de ganancia que los sitúa en la región de la estabilidad, sin embargo, no puede decirse lo mismo de su margen de fase. Aun así, ambos controladores dan una respuesta estable y a pesar de limitar la representación de sus coeficientes a menos bits, ninguno parece haber perdido mucha precisión.

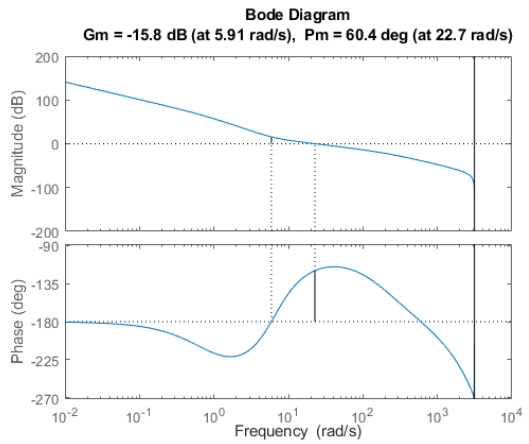
Actualizar bodes de delta y sustituir continuo por discreto con precisión de máquina.



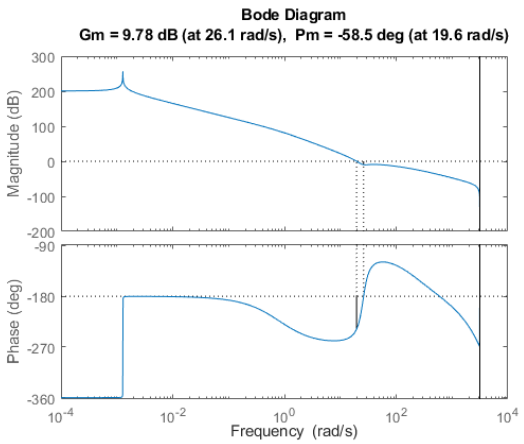

**a** *Controlador continuo.*




**b** *Controlador $\delta$ limitado a 16 bits.*




**c** *Controlador Z limitado a 16 bits.*




**d** *Controlador Z limitado a 12 bits.*

**Figura 23:** *Comparativa que muestra los diagramas de bode de la respuesta en lazo abierto con distintos controladores.*

En la Figura 23 se muestran los diagramas de bode de algunas de las respuestas del sistema en lazo abierto con distintos controladores.

## 3.6. Ejercicio 6

Como ya se ha mostrado en las Figuras 20 y 22, la respuesta en lazo cerrado de la planta se ve muy afectada por la forma de discretización del controlador y también por la cantidad de bits que se escoja para la representación de los coeficientes de los controladores.

Por un lado, los resultados de la Tabla 9 muestran que cualquier discretización del controlador supondrán una pérdida en algunas de sus características, y esto se ve reflejado en la Figura 20, donde se ve que la respuesta del controlador Z no llega a tener una respuesta tan buena como el controlador continuo.

Por otro lado, las ventajas que se obtienen al utilizar formas directas para la escritura de los controladores se pierden si no se usa una cantidad suficiente de bits que sea lo bastante representativa para expresar los coeficientes. Esto es especialmente notorio si los coeficientes a expresar tienen un parte entera que requiera tantos bits como la parte fraccional para ser representada. Los efectos de esto son claramente apreciables en la Figura 22b.

# 4. Conclusiones generales

Haciendo una vista general de lo aprendido en estas tres prácticas podemos destacar la importancia del uso de frecuencias de muestreo óptimas según las especificaciones del proyecto, con el objetivo de evitar la sensibilidad a retrasos de computación o que estos puedan llegar a ser una carga computacional excesiva para los elementos lógicos, que podrían llegar a sobrecargarse ante un gran número de operaciones. No obstante, hemos llegado a la conclusión de que un sobremuestreo aproximadamente 100 mayor al ancho de banda del sistema puede ser óptimo para una gran variedad de casos.

Respecto a los beneficios que puede generar un filtro antialiasing, previo a la adquisición de datos, podemos destacar que no es necesario poner un filtro de orden muy alto, puesto que la mayoría de los beneficios que puede aportar se pueden obtener con filtros de primer y segundo orden. Sin embargo, si que es necesario tener en cuenta la frecuencia del filtro porque dependiendo del valor que se escoja se puede evitar que entren espurios y frecuencias indeseadas dentro de la banda de transición del filtro. Posterior a eso, en la misma práctica hemos visto que existen diferentes formas de implementar los mismos controladores para poder aprovechar adecuadamente los recursos de los que disponemos, como: sumadores, multiplicadores y elementos de memoria que hacen su implementación más adaptable a las características de las plataformas en la que se vayan a usar.

Por otro lado, en la última practica hemos podido ver los beneficios que representa la transformada $\delta$ para la implementación de controladores discretos. En este sentido, hemos podido generar scripts y módulos que pueden ser muy útiles para futuros proyectos. Estos están adaptados a la capacidad de las plataformas Hardware que podremos manipular y a cómo definamos las diferentes longitudes de palabra para las variables. Es destacable que no sea necesario tener una precisión tan grande como nos ofrece un tipo de dato *Double* (64bits) para dar los parámetros de un controlador efectivo. Sin embargo, llegado un límite puede llegar a descontrolar el sistema.