Universidad Nacional de Córdoba

Facultad de Ciencias Exactas, Físicas y Naturales

Sistemas de Control

Trabajo Final Integrador

Profesor: AGUERO Adrian, REYES Reinaldo

Alumno/s: SIAMPICHETTI Gino, MAFFINI Agustin

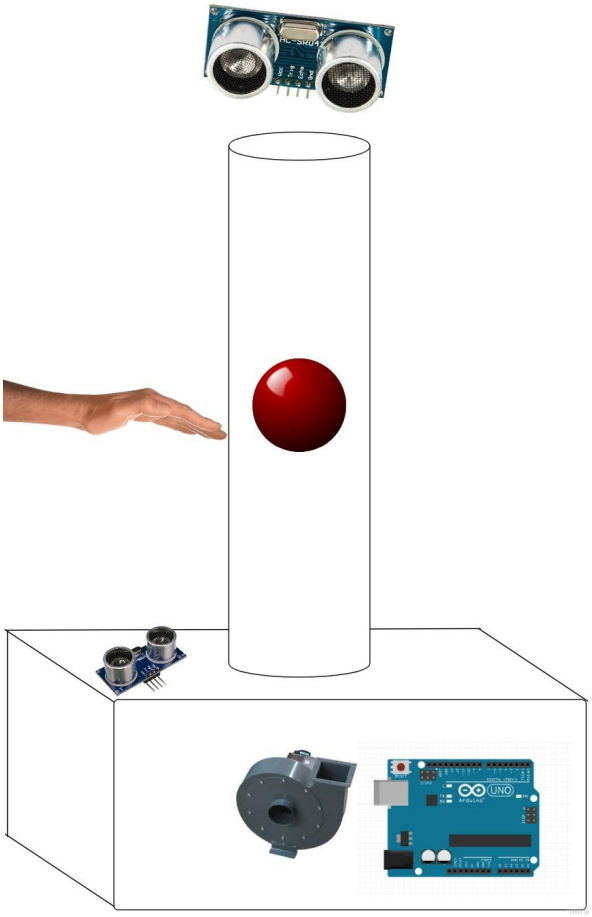
Matrícula/s: 41411540,

Carrera: Ing Computación

Índice

Introducción

Con el fin de aplicar los conocimientos adquiridos en la asignatura Sistemas de Control I, desarrollamos este trabajo práctico integrador basándonos en el control de levitación de una pelota de telgopor, la cual se desplaza verticalmente dentro de un tubo cilíndrico transparente, impulsada por un soplador de aire (motor CD con paletas planas perpendiculares al sentido de giro). La posición de la pelota es ajustada con la posición de la mano, detectada por un sensor ultrasónico, con un rango de 0-39 cm.



Desarrollo

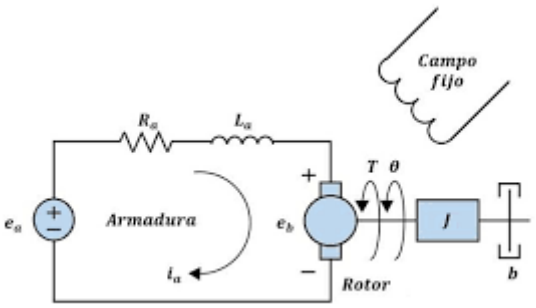
Análisis sin considerar la carga

Determinación de los elementos a utilizar

- Arduino UNO
- Motor DC (Datasheet)
- (2) Sensor ultrasonico HC-SR04 (Datasheet)
- Resistencia 220 [Ω]
- Transistor NPN 2N2222A
- Diodo 1N4007
- Esfera telgopor (pelota)
- Protoboard
- Entre otros

Modelado del motor de corriente continua

El motor DC controlado por inducido utiliza la corriente del inducido como la variable de control. El campo en el estator se puede establecer mediante una bobina de excitación y corriente o un imán permanente. En la siguiente figura se muestran los elementos más importantes:



Podemos analizar este circuito en el dominio del tiempo, planteando las siguientes ecuaciones

$$e_a(t) = R_a i_a(t) + L_a \frac{di_a(t)}{dt} + e_b(t)$$

$$e_a(t) - R_a i_a(t) - e_b(t) = L_a \frac{di_a(t)}{dt} \quad (1)$$

donde $e_b(t)$ es la fuerza contraelectromotriz.

Observando la parte mecánica,

$$T_m(t) = J\frac{d\omega(t)}{dt} + B\omega(t)$$

$$T_m(t) - B\omega(t) = J\frac{d\omega(t)}{dt} \quad (2)$$

Donde $T_m(t)$ es el torque del motor de corriente continua, B es el coeficiente de fricción equivalente al motor de CD (corriente continua) y la carga montados sobre el eje del motor, J es el momento de inercia total del rotor y de la carga con relación al eje del motor, $\omega(t)$ es la velocidad angular del motor y $\frac{d\omega(t)}{dt}$ es la aceleración angular.

Asumiendo que existe una relación proporcional K_a (constante contraelectromotriz $\frac{V}{rad\ s}$) entre el voltaje inducido en la armadura y la velocidad angular del eje del motor,

$$e_b(t) = K_a\omega(t) \quad (3)$$

Y se supone la siguiente relación electromecánica que establece que el torque mecánico es proporcional, K_m(Constante de Torque [Nm / A]), a la corriente eléctrica que circula por el motor DC.

$$T_m(t) = K_m i_a(t) \quad (4)$$

Podemos aplicar la transformada de Laplace a las ecuaciones 1, 2, 3 y 4 para obtener con mayor facilidad la función de transferencia

$$E_a(s) - R_a I_a(s) - E_b(s) = sL_a I_a(s) \quad (5)$$

$$T_m(s) - B\omega(s) = Js\omega(s) \quad (6)$$

$$E_b(s) = K_a\omega(s) \quad (7)$$

$$T_m(s) = K_m I_a(s) \quad (8)$$

Queremos obtener la relación velocidad angular con el voltaje de entrada, $\omega(s)/E_a(s)$.

De la ecuación 8, podemos despejar la corriente

$$I_a(s) = \frac{T_m(s)}{K_m} \quad (9)$$

de la ecuación 6, despejamos el torque del motor

$$T_m(s) = \omega(s)(Js + B) \quad (10)$$

reemplazamos la ecuación 10 en la 9

$$I_a(s) = \frac{\omega(s)(Js+B)}{K_m} \quad (11)$$

reemplazamos la ecuación 7 en la 5 y agrupamos los términos en función de la corriente

$$E_a(s) - K_a\omega(s) = I_a(s)(sL_a + R_a) \quad (12)$$

reemplazamos la ecuación 11 en la 12 y agrupamos los términos en función de la velocidad angular

$$E_a(s) - K_a\omega(s) = \frac{\omega(s)(Js+B)}{K_m}(sL_a + R_a)$$

$$E_a(s) = \omega(s)[\frac{(Js+B)(sL_a+R_a)}{K_m} + K_a]$$

$$E_a(s) = \omega(s)[\frac{(Js+B)(sL_a+R_a)+K_aK_m}{K_m}]$$

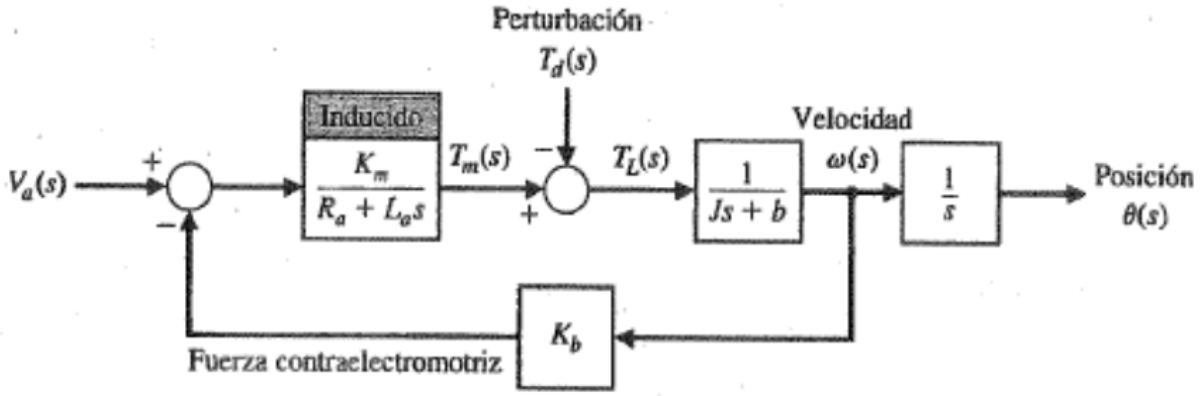
por último, despejamos y desarrollamos para obtener la relación entre el voltaje de entrada y la velocidad (salida):

$$\frac{\omega(s)}{E_a(s)} = \frac{K_m}{(Js+B)(sL_a+R_a)+K_aK_m}$$

$$\frac{\omega(s)}{E_a(s)} = \frac{K_m}{JL_as^2+(JR_a+BL_a)s+K_aK_m}$$

Diagrama de bloques del sistema

Respecto al modelado matemático, el diagrama de bloques en el dominio de Laplace es



Si bien el motor CD es de lazo abierto, la realimentación negativa de ganancia K_b se ilustra solo para el modelo matemático.

Determinación de la función de transferencia del sistema

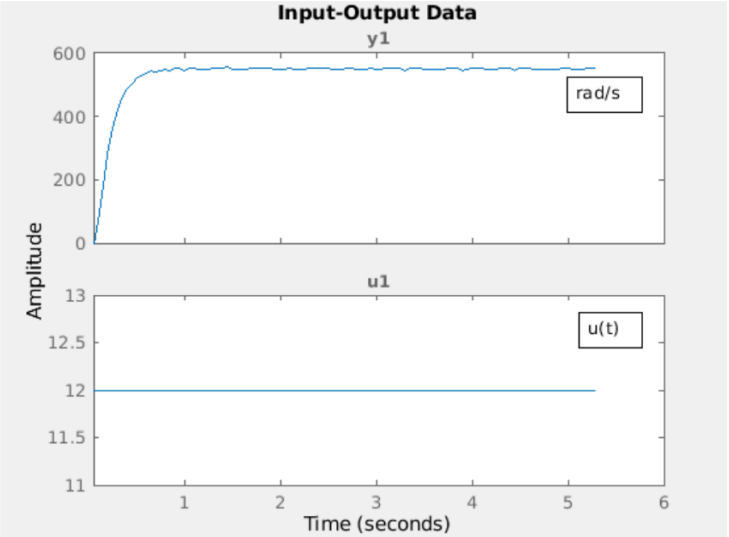
Para determinar la función de transferencia a lazo abierto del motor de corriente continua utilizado, al desconocer sus características físicas, optamos por realizar mediciones experimentales con ayuda del Arduino UNO, un módulo de sensor infrarrojo y en el eje del motor colocamos una hélice liviana de 8 palas. De esta manera, al girar la hélice, el sensor infrarrojo detecta las palas y el Arduino cuenta la cantidad en un tiempo de muestreo determinado, mediante interrupciones. El resultado es RAD/s ante la entrada escalón unitario sobre el motor CD. Ver código implementado en Arduino.

A continuación, se detalla el procedimiento introducido en Matlab para obtener la función de transferencia a partir de los resultados obtenidos.

Estimación 1

Se definieron los vectores **u** y **rads** (entrada escalón y velocidad angular, respectivamente). La herramienta *'iddata()'* nos permite encapsular los valores medidos de entrada y salida para el sistema (motor) que queremos identificar, con un tiempo de muestreo dado. (Datos: hoja 1)

```
data1 = iddata(rads, u, 0.05); % salida, entrada, muestreo
plot(data1);   % graficamos la entrada y la salida en el tiempo
```



Con la herramienta *'tfest()'* realizamos una estimación de la función de transferencia, según los polos y ceros que indiquemos. En principio, el sistema parece ser de primer orden:

```
tf10 = tfest(data1, 1, 0);
```

obteniendo una estimación muy buena del 90.9%. Sin embargo, del modelado matemático sabemos que la función de transferencia de un motor CD es de segundo orden, con 2 polos y ningún zero:

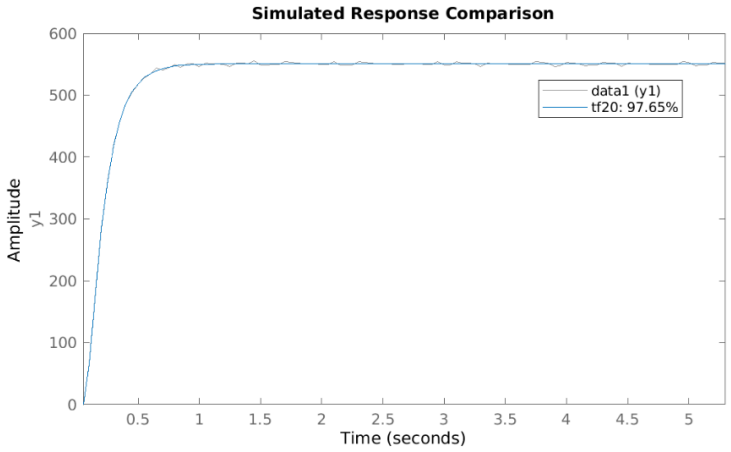
```
tf20 = tfest(data1, 2, 0);
```

En este caso, la estimación aumentó a 97.65% y la función de transferencia calculada es

$$TF_{20}(s) = \frac{8104}{s^2+31.83s+176.6}$$

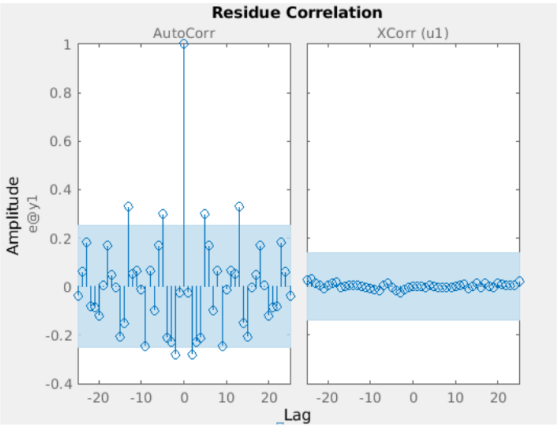
Podemos comparar los datos ingresados (*data1*) con la estimación realizada:

```
compare(data1, tf20);
```



Claramente es muy buena, prácticamente idéntica. Además, podemos hacer un análisis de residuos y veremos que hay una alta correlación:

```
resid(data1, tf20);
```

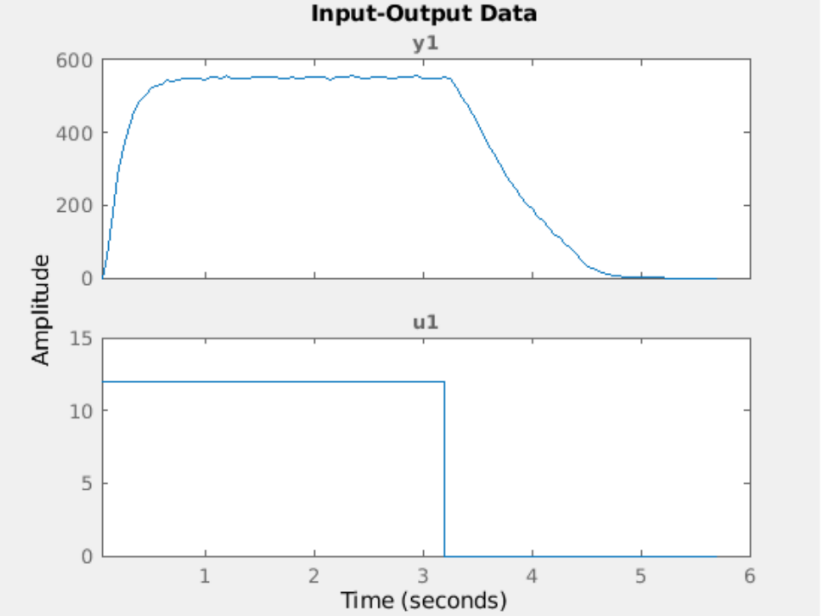


Estimación 2

Respecto a la estimación 1, queremos obtener una función de transferencia teniendo en cuenta el caso cuando el motor se desenergiza y, por lo tanto, la velocidad disminuye. Para ello, ingresamos un pulso de unos segundos. Los resultados fueron los siguientes: (datos: hoja 2)

```
u1 = [];      % escalon unitario
rads1 = [];  % valores de rad/s obtenidos
data2 = iddata(rads1, u1, 0.05);
```

```
plot(data2);
```



Buscamos la estimación que más se aproxime al 100%, variando el número de polos y ceros

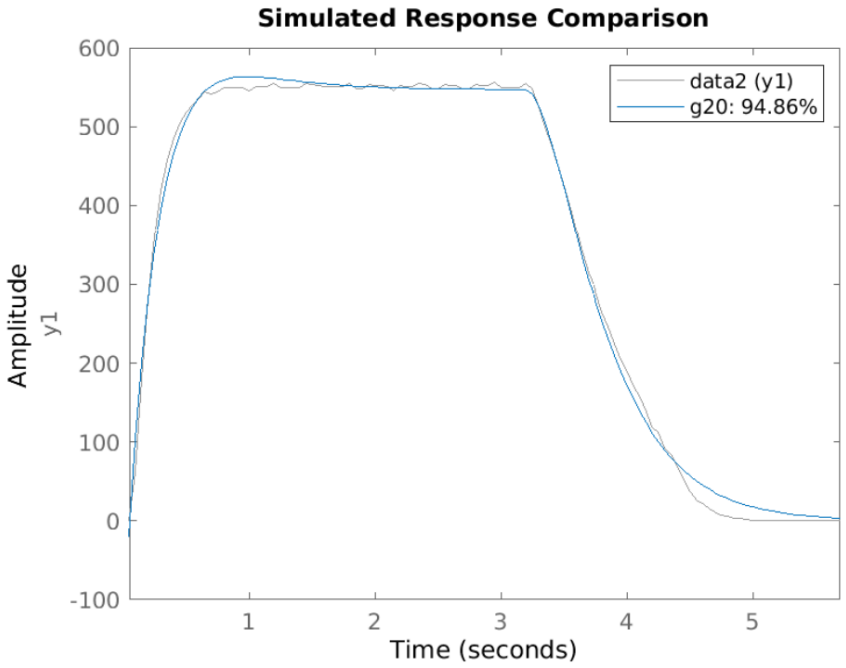
```
g10 = tfest(data2, 1, 0);    % est: 78.61%
g20 = tfest(data2, 2, 0);    % est: 94.86%
g21 = tfest(data2, 2, 1);    % est: 84.38%
g31 = tfest(data2, 3, 1);    % est: 86.96%
g32 = tfest(data2, 3, 2);    % est: 83.26%
```

Nos quedamos con la estimación g20, que es muy buena y además tiene la misma cantidad de polos y ceros que $TF_{20}(s)$ (calculada en la estimación 1). Entonces, $G_{20}(s)$ es,

$$G_{20}(s) = \frac{418.2}{s^2+6.184s+9.174}$$

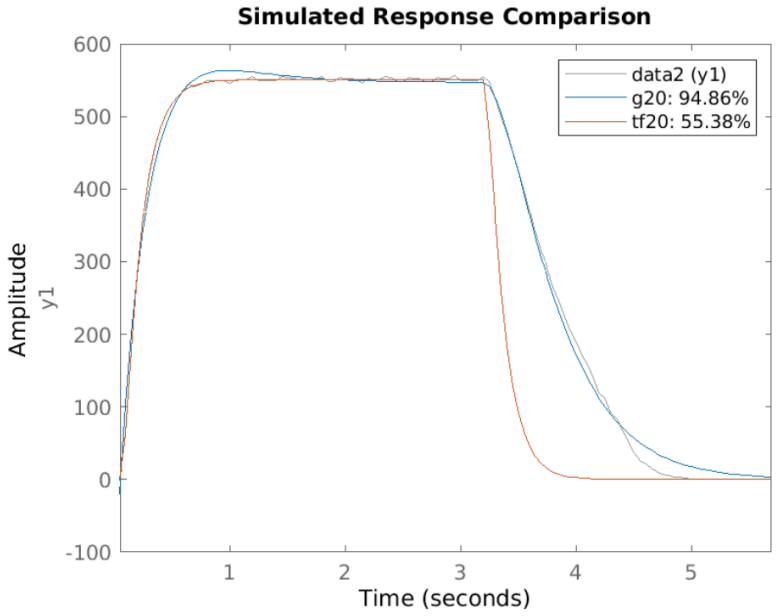
Aquí se encuentra el script de Matlab.

Si la comparamos con los datos ingresados, el resultado sigue siendo muy bueno



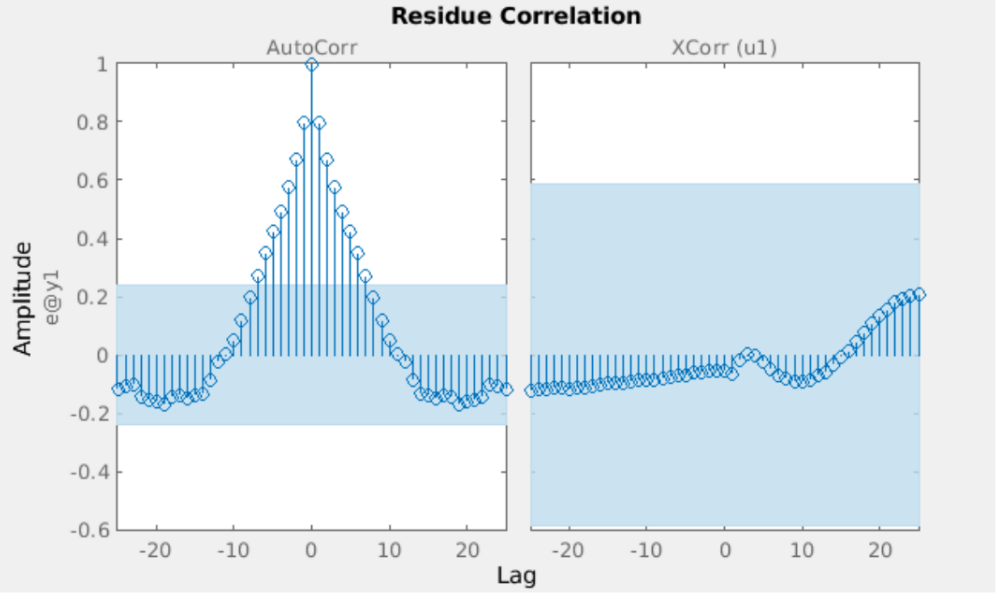
Si bien g20 tiene una estimación menor que tf20, tenemos una representación más amplia y real de la respuesta del motor. Esto se refleja al comparar ambas estimaciones con los datos experimentales:

```
compare(data2, g20, tf20);
```



Nuevamente hacemos un análisis de residuos y, si bien la correlación disminuyó respecto al primer caso, sigue siendo buena.

```
resid(data2, g20);
```



De esta manera, concluimos que la función de transferencia a lazo abierto del motor CD es $G_{20}(s)$:

$$G_{20}(s) = \frac{418.2}{s^2+6.184s+9.174}$$

Comparando esta expresión con una típica de segundo orden,

$$F(s) = \frac{K_m}{s^2+2\zeta\omega_n s+\omega_n^2}$$

$$K_m = 418.2 \quad ; \quad 2\zeta\omega_n = 6.184 \quad ; \quad \omega_n^2 = 9.174 \quad \rightarrow \quad \omega_n = 3.023 \quad ; \quad \zeta = 1.021$$

Con este resultado realizaremos los siguientes análisis y diseños.

Análisis del error de régimen permanente

Observando la expresión $G_{20}(s)$, vemos que es un sistema de tipo 0, por lo tanto

- Entrada escalón u(t):

$$k_p = \lim_{s\to 0} G_{20}(s) = \frac{418.2}{0+0+9.174} = 45.585 \quad \rightarrow \quad ess = \frac{1}{1+kp} = 0.021$$

- Entrada rampa r(t)

$$k_v = \lim_{s\to 0} sG_{20}(s) = 0\frac{418.2}{0+0+9.174} = 0 \quad \rightarrow \quad ess = \frac{1}{kv} = \infty$$

- Entrada parábola p(t):

$$k_a = \lim_{s\to 0} s^2G_{20}(s) = 0^2\frac{418.2}{0+0+9.174} = 0 \quad \rightarrow \quad ess = \frac{1}{ka} = \infty$$

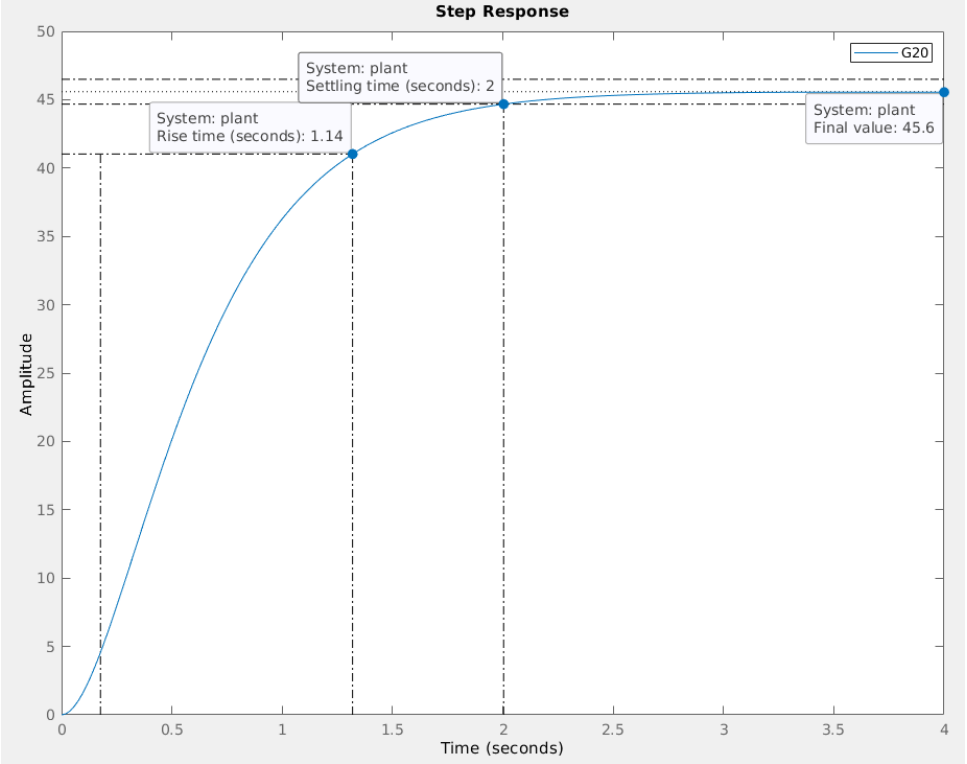
Respuesta transitoria

Tenemos una función de transferencia de segundo orden, por lo tanto analizamos los parámetros ω_n y ζ:

$$\omega_n = 3.023 \quad ; \quad \zeta = 1.021$$

- $\zeta \approx 1$, entonces decimos que es un sistema de amortiguamiento crítico.

Para analizar más detalladamente su respuesta temporal, graficamos la respuesta al escalón de la función de transferencia $G_{20}(s)$ del motor:



Del gráfico observamos que el tiempo que tarda el sistema en ir desde una amplitud del 10% al 90% es de 1.14 segundos (rise time), el tiempo que tarda en establecerse es de 2 segundos (settling time, tolerancia 5%) y el valor final es de 45.6 (final value). El sistema no presenta sobrepasamiento, por lo cual, no tenemos un tiempo pico.

Análisis de Estabilidad

Para analizar si el sistema es estable, utilizaremos el método de Routh. El criterio se refiere a la función de transferencia en lazo cerrado del sistema con realimentación unitaria, por lo cual desarrollamos el polinomio en el denominador de $G_{20}(s)$, es decir,

$$G_{CL}(s) = \frac{G_{20}(s)}{1+G_{20}(s)} = \frac{418.2}{s^2+6.184\,s+427.37}$$

la ecuación característica se encuentra en el denominador

$$p(s) = s^2 + 6.184s + 427.37$$

Analizamos si en la primera columna de la siguiente tabla hay cambio de signo

	a_0	a_2
s^2	1.0	427.37
	a_1	a_3
s^1	6.184	0
	b_0	b_1
s^0	427.37	0

como no lo hay, entonces decimos que el sistema es estable.

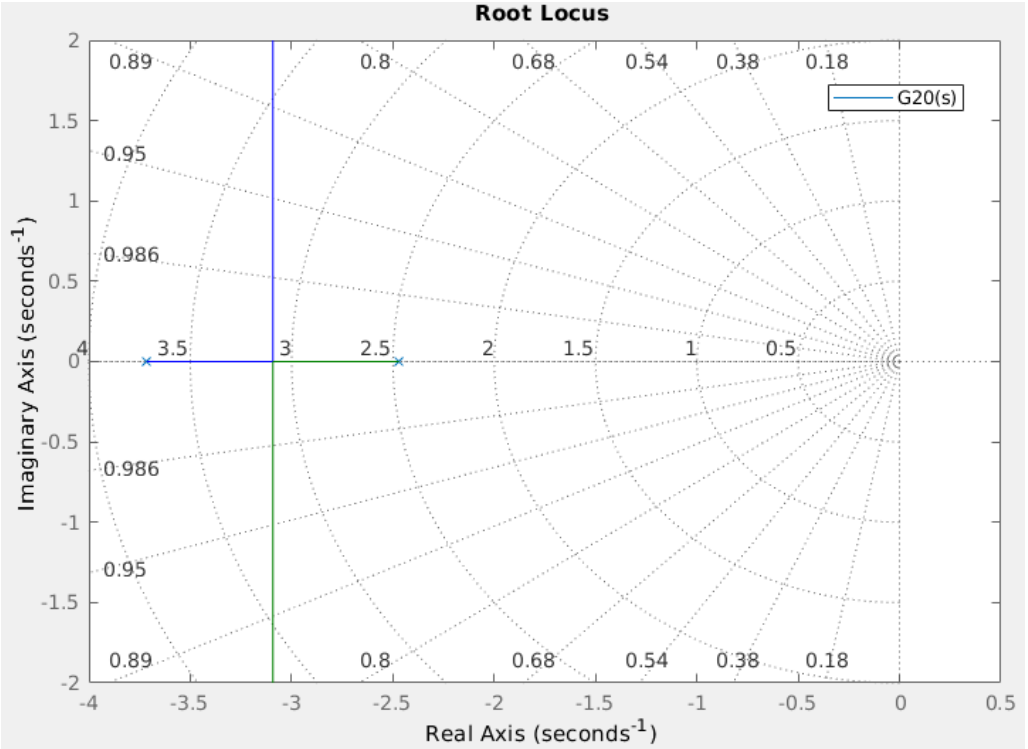
Además, podemos determinar si el sistema es estable a partir de los polos de $G_{20}(s)$: $p_1 = -3.71$, $p_2 = -2.47$. Ambos polos se encuentran a la izquierda del eje imaginario en el plano *S.* Por lo tanto, concluimos que efectivamente es sistema es estable.

Lugar de Raíces

Analizamos la función de transferencia $G_{20}(s)$ para graficar el lugar de raíces (LdR). En principio, sabemos que tiene 2 polos, por lo cual el LdR tiene 2 ramas que comienzan en dichos polos y terminan en $\pm\ \infty$, ambas con un ángulo de salida de 90°. El centroide se ubica en

$$\beta = \frac{\sum_i^m P_i + \sum_j^n Z_j}{m-n} = \frac{-3.71-2.47-0}{2-0} = -\ 3.09$$

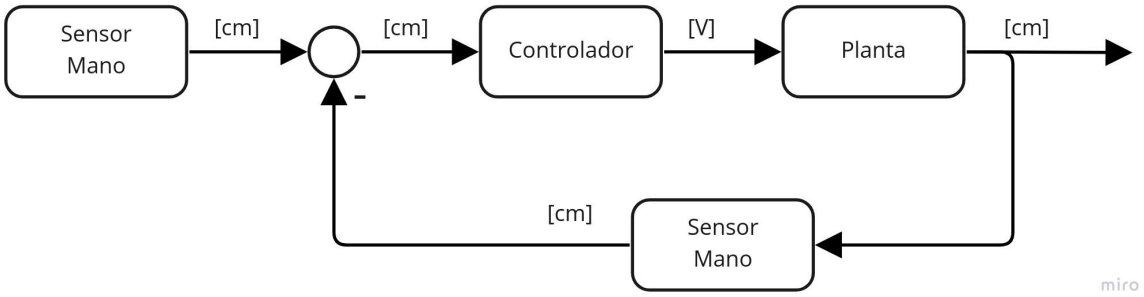
donde m y n son la cantidad de polos y ceros, respectivamente. Graficamos el LdR en Matlab:



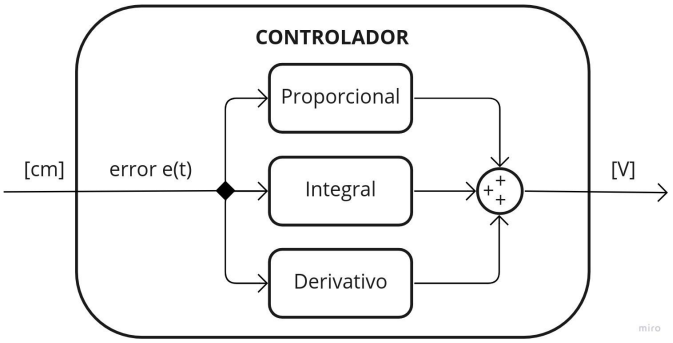
Análisis considerando la carga

Diagrama de bloques del sistema completo

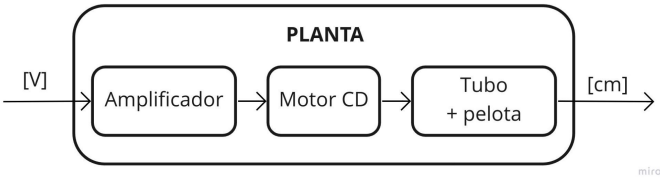
El diagrama de bloques del sistema en general se muestra a continuación. La entrada es el sensor de la mano, que ingresa una referencia en centímetros. El sumador hace la diferencia entre la entrada y la realimentación (unitaria, en este caso). Luego se encuentra el controlador para la planta. La salida es la posición de la pelota, en centímetros.

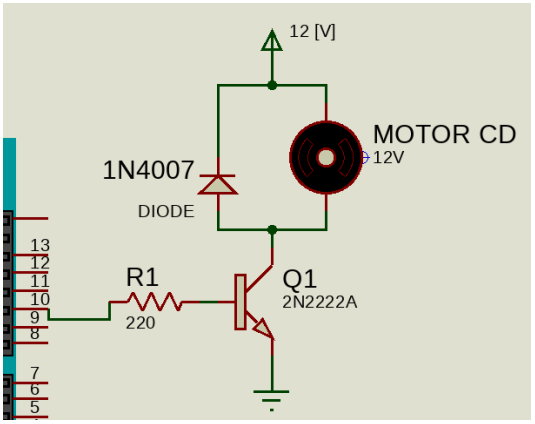


En principio, el controlador es un PID (Proporcional-Integrador-Derivador).



La planta se compone de un amplificador para controlar el motor DC, el cual eleva la pelota en el tubo.



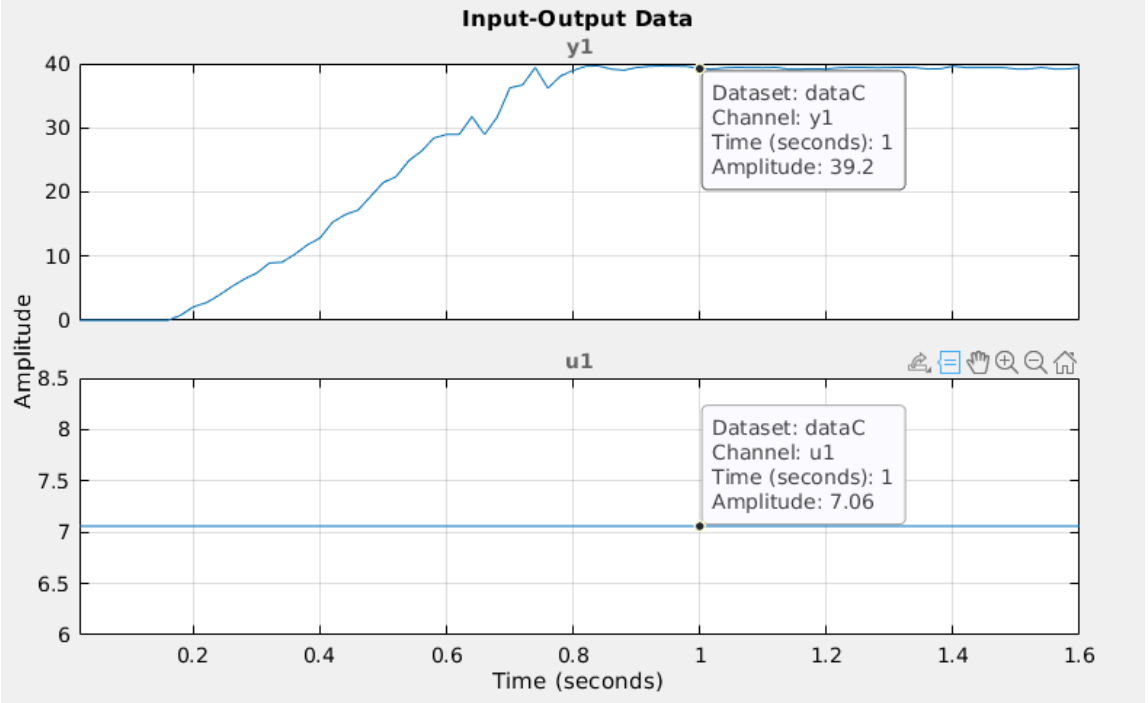


Determinación de la función de transferencia del sistema completo

Para obtener la función de transferencia del sistema completo, realizamos un procedimiento similar al presentado previamente. Es decir, aplicamos una entrada escalón a la planta y la salida será la altura a la que se eleva la pelota, obtenida con un tiempo de muestreo. Ver Anexo para más detalles.

Los datos de entrada-salida se encuentran aquí.

Nuevamente, con la ayuda de Matlab, realizamos la estimación de la función de transferencia. Para los datos ingresados:



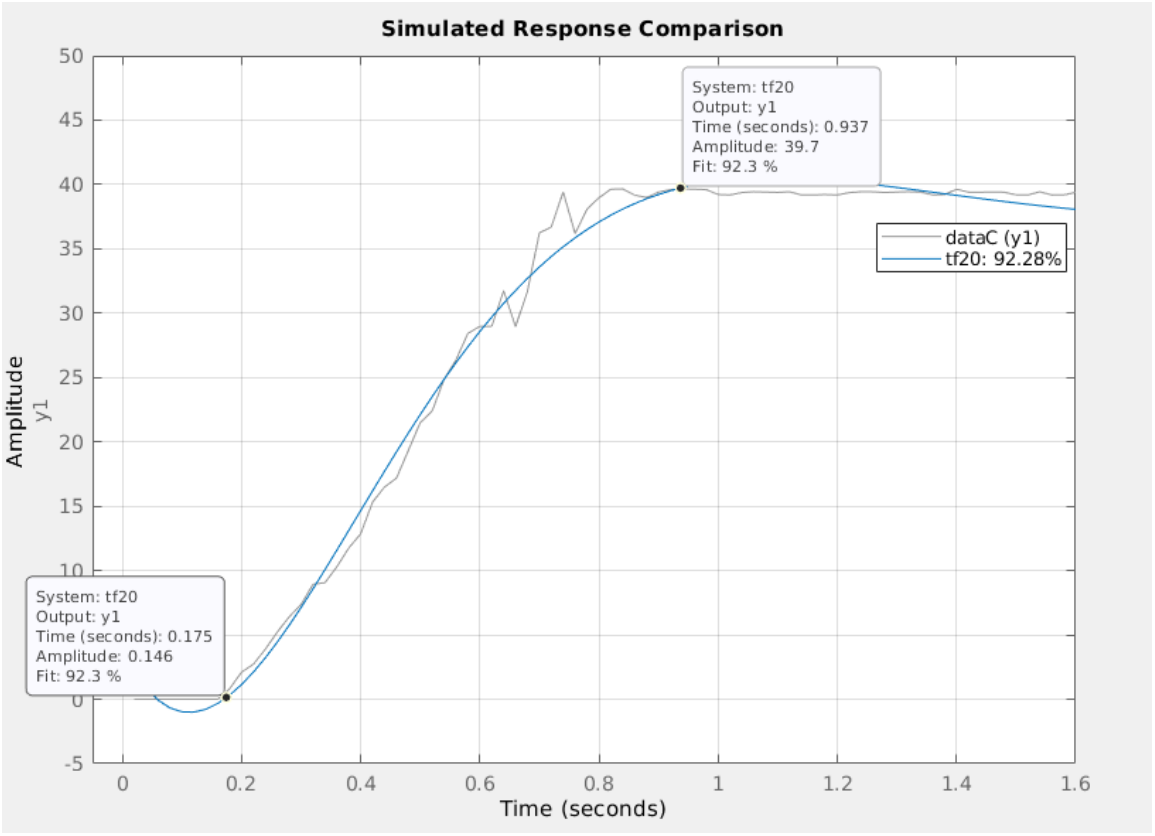
estimamos una función de transferencia con 0 ceros y 2 polos, y el resultado se ajusta a una estimación del 92.28%.

```
tf20 =

  From input "u1" to output "y1":
          89.56
  ---------------------
  s^2 + 5.145 s + 16.88

Continuous-time identified transfer function.

Parameterization:
   Number of poles: 2   Number of zeros: 0
   Number of free coefficients: 3
   Use "tfdata", "getpvec", "getcov" for parameters and their uncertainties.

Status:
Estimated using TFEST on time domain data "dataC".
Fit to estimation data: 92.28%
FPE: 1.536, MSE: 1.355
```

De esta manera, la función de transferencia de la planta con carga y a lazo abierto es

$$G_{POL}(s) = \frac{89.56}{s^2+5.145s+16.88}$$

En la siguiente figura podemos ver la correlación entre los datos ingresados y la respuesta de $G_{POL}(s)$ a la misma entrada escalón.



Compensación

Para el diseño del compensador, primero planteamos que los objetivos son: una respuesta rápida del sistema ($t_S = 3\ [s]$) con leve amortiguamiento, error en estado estable nulo ($ess = 0$) para entrada escalón y sobrepasamiento $OS = 5\%$. Un compensador PID (proporcional-integrador-derivativo) podría satisfacer estas necesidades.

Un compensador PID tiene la forma

$$G_{PID}(s) = K_P + \frac{K_i}{s} + K_D s = \frac{K_D s^2 + K_P s + K_i}{s}$$

donde K_P, K_i y K_D son las constantes proporcional, integral y derivativa, respectivamente.

Si hacemos el producto del compensador con la planta,

$$G_{OL}(s) = G_{PID}(s)\, G_{POL}(s) = \frac{89.56\,(K_D s^2 + K_P s + K_i)}{s(s^2 + 5.145\, s + 16.88)}$$

podemos notar que $G_{OL}(s)$ es un sistema es de tipo 1, por lo tanto se cumple $ess = 0$ para una entrada escalón.

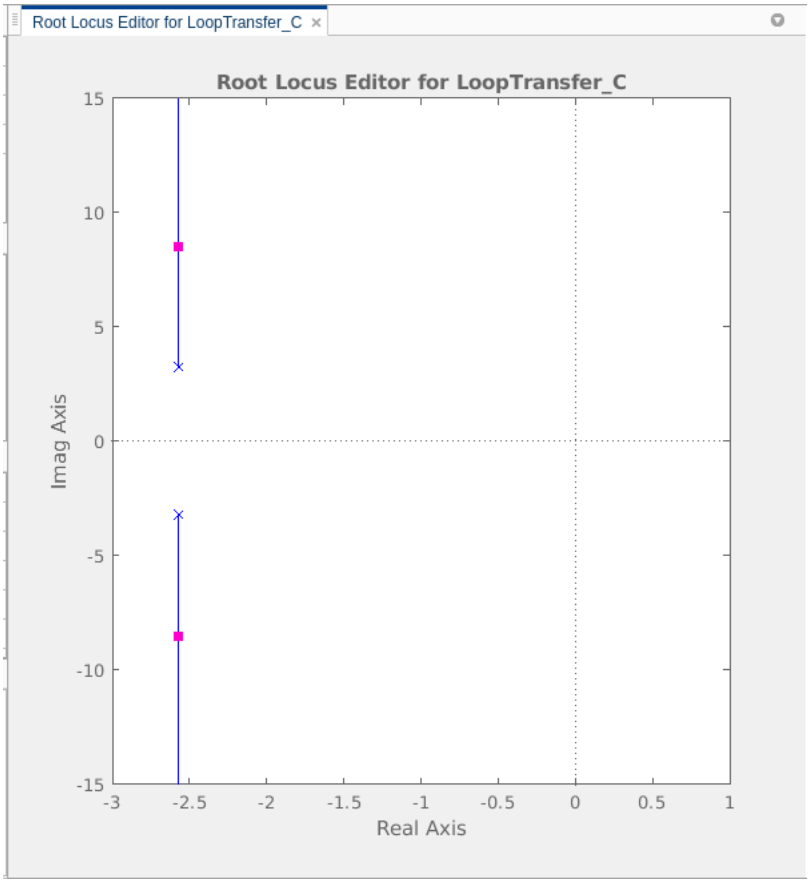
Dados los requerimientos de diseño, calculamos el polo de trabajo P_T

$$\zeta = \sqrt{\frac{ln^2(Mp)}{ln^2(Mp)+\pi^2}} = \sqrt{\frac{ln^2(0.05)}{ln^2(0.05)+\pi^2}} = 0.6901$$

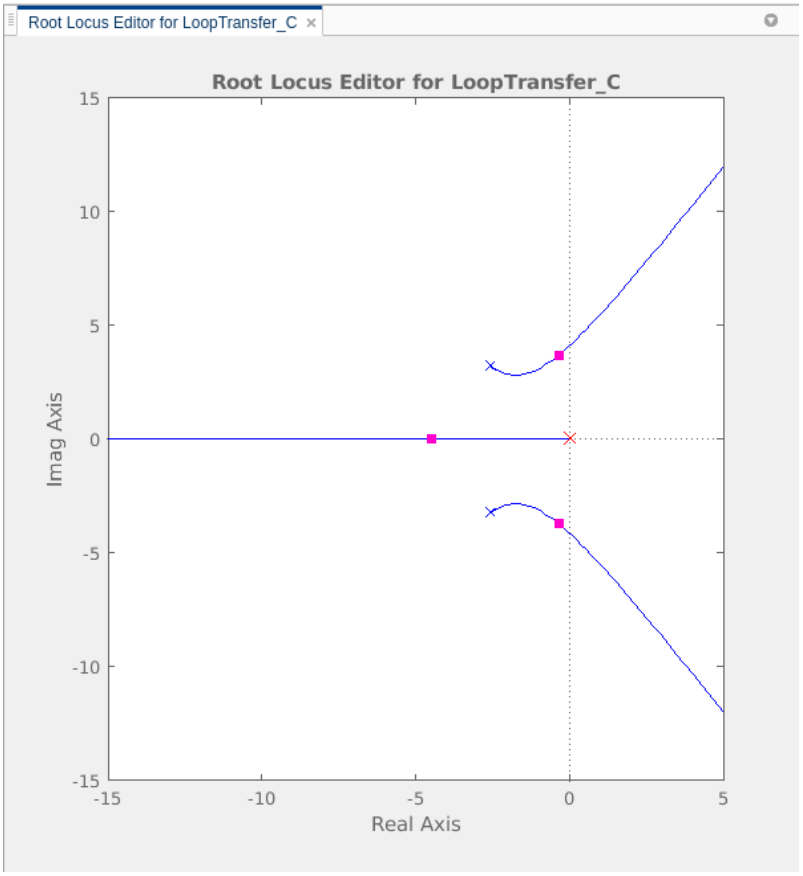
$$\omega_n = \frac{5}{t_s \zeta} = \frac{5}{3x0.69} = 2.4151$$

$$P_T = -\zeta\omega_n \pm \omega_n\sqrt{1-\zeta^2} = -1.6667 \pm j1.7478$$

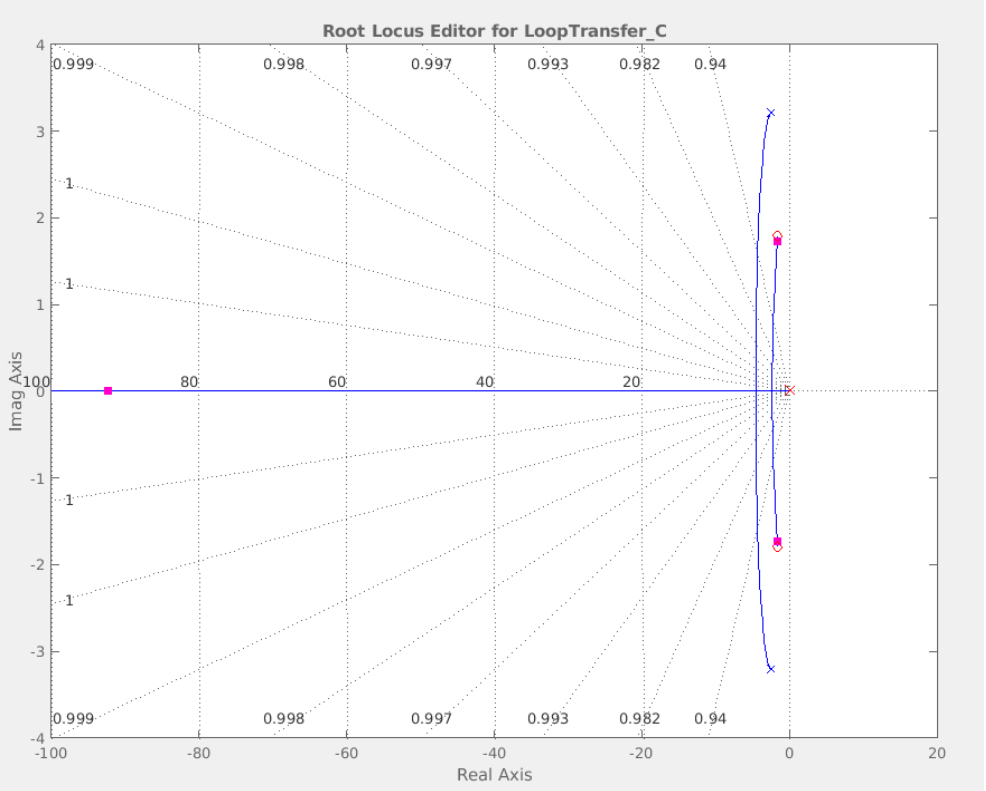
Graficamos el lugar de raíces de $G_{POL}(s)$ a lazo abierto con la herramienta *'controlSystemDesigner'* de Matlab.



Queremos diseñar un PID, por lo cual primero debemos agregar un polo en el origen (integrador).



Proponemos agregar un par de ceros complejos conjugados de tal forma que el polo de trabajo pertenezca al lugar de raíces.



Colocando los ceros en $-\ 1.64\ \pm\ j1.79$ logramos que el polo de trabajo pertenezca al lugar de raíces y ajustamos la ganancia tal que se cumplan los requerimientos. A esto último lo podemos verificar con la herramienta que estamos utilizando, agregando nuestras especificaciones en el mismo plano S.

El compensador que obtuvimos es

$$G_{PID}(s) = \frac{5.9676+3.3481\,s+1.003\,s^2}{s}$$

las constantes del compensador quedan a la vista

$$G_{PID}(s) = \frac{5.9676+3.3481\,s+1.003\,s^2}{s} = \frac{K_D s^2+K_P s+K_i}{s}$$

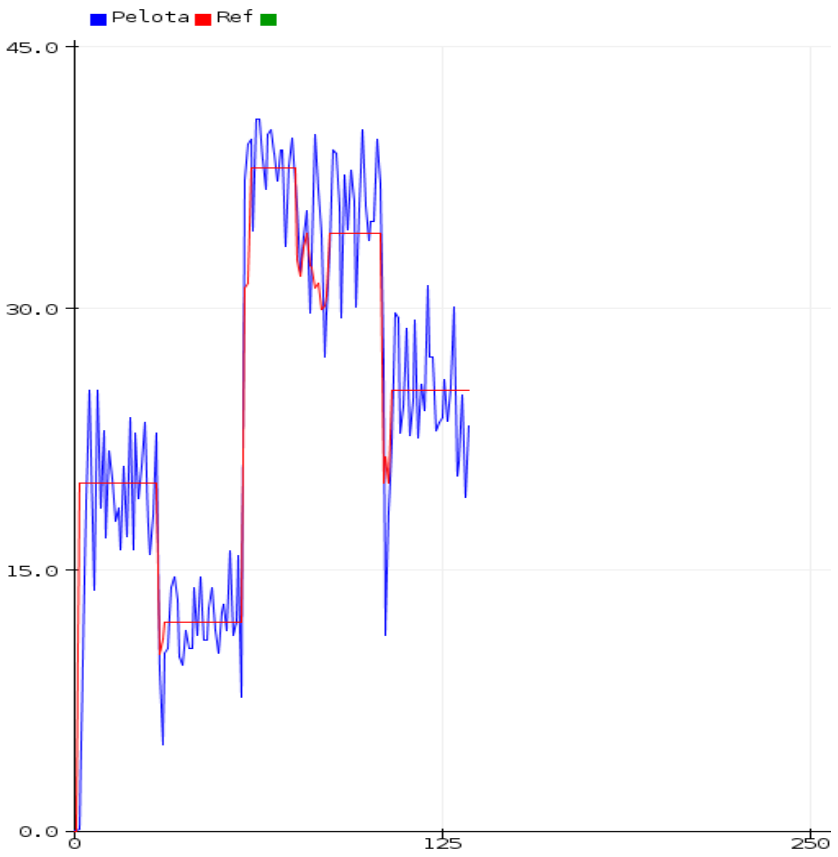
$$K_P = 3.3481 \quad ; \quad K_i = 5.9676 \quad ; \quad K_D = 1.003$$

Aquí se puede encontrar el script en Matlab.

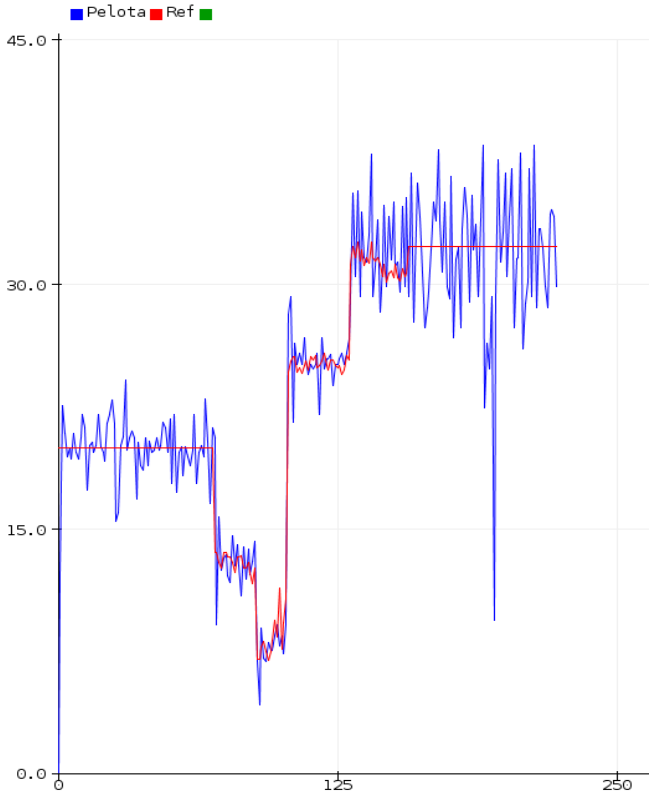
Ajuste del Compensador

Hasta aquí, hemos desarrollado la función de transferencia de la planta y compensador en base a estimaciones que, si bien obtuvimos buenas aproximaciones, es aceptable que en la práctica haya que ajustar algunos detalles. Es por ello que queremos observar cómo responde el sistema a lazo cerrado con el compensador diseñado. Para ello, desarrollamos un código en Arduino. Ver Anexo para más detalles.

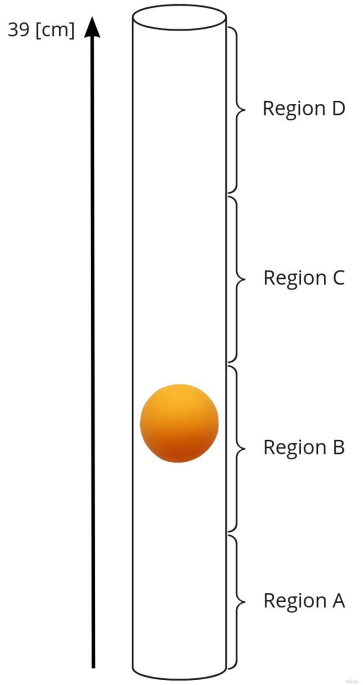
En principio, queremos posicionar la pelota a una altura de 20 cm. Luego con la mano le damos otros puntos de referencia. Observamos que la pelota alcanza rápidamente la altura de referencia, pero oscila mucho y nunca se estabiliza en un punto concreto.



Probamos que disminuir la constante derivativa e integrativa, ambas un 10%



Para una referencia menor a 30 cm, vemos que el tiempo de establecimiento es rápido y la posición de la pelota es relativamente estable, pero una referencia mayor, es decir una altura más elevada, presenta oscilaciones. Esto nos llevó a la conclusión de que la planta no es un sistema lineal, y esto se puede corroborar con el gráfico *'Input-Output Data'*. Entonces, debemos linealizarlo de alguna manera para implementar el controlador PID. La solución que encontramos para minimizar el error es dividir el tubo en 4 partes distintas, como se muestra en la siguiente figura:



Tenemos las regiones A={0, 9}, B={9.1, 18}, C={18.1, 30} y D={30.1, 40} por lo tanto las constantes para el PID serán distintas. De esta manera, tendremos las constantes:

regionA = {Kpa, Kia, Kda}, regionB = {Kpb, Kib, Kdb}, regionC = {Kpc, Kic, Kdc}, regionD = {Kpd, Kid, Kdd}

que se determinan de manera experimental. Es intuitivo pensar que en la región A necesitamos una constante proporcional más grande, porque el flujo de aire es más turbulento y se requiere más fuerza para elevar la pelota desde el reposo.

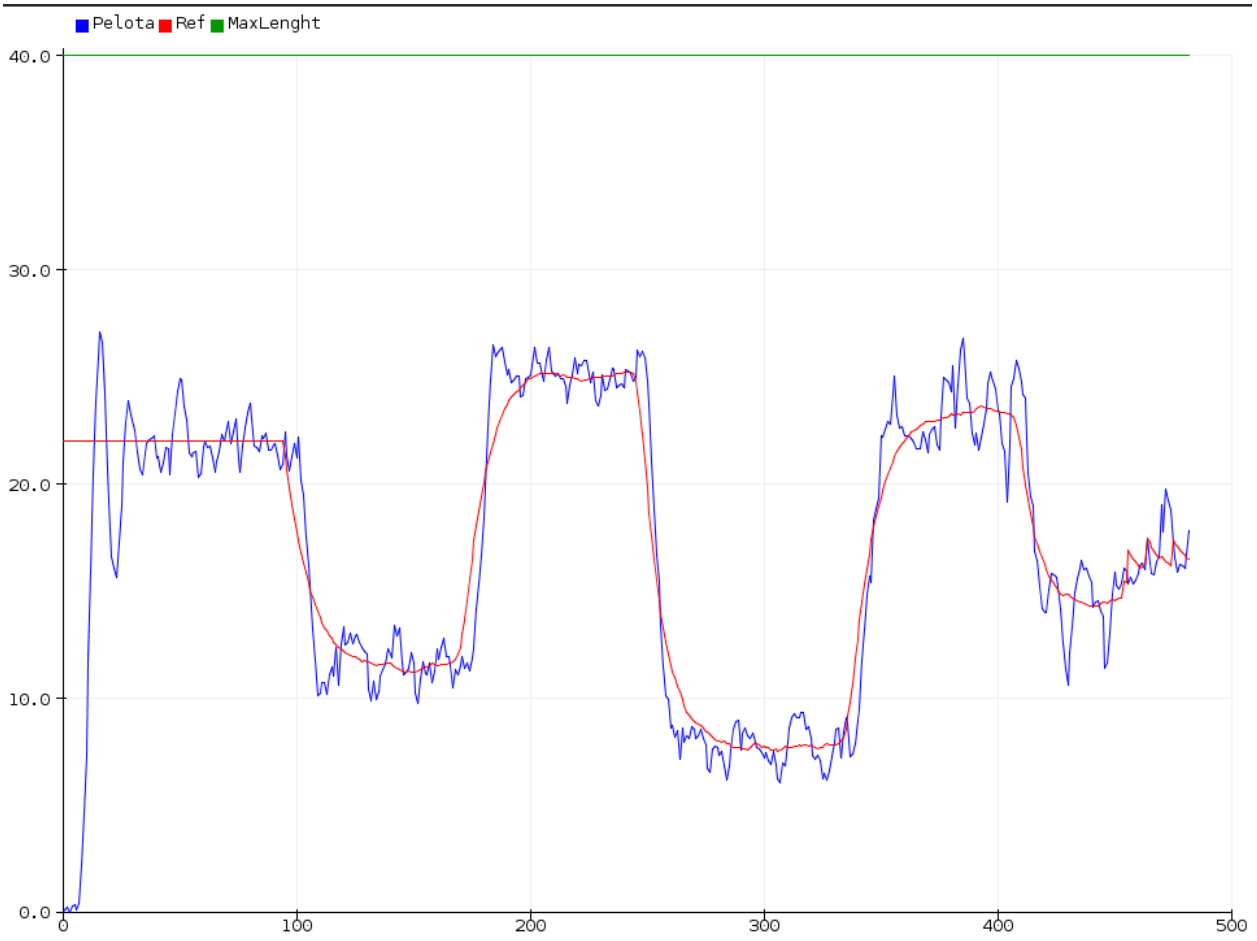
Por otro lado, en la parte superior tenemos que el aire fluye más estable y el movimiento de la pelota es sensible a los cambios en la energía del motor.

Proponemos

regionA = {4.3325, 4.8, 0.1}, regionB = {3.4821, 5.6, 0.2},

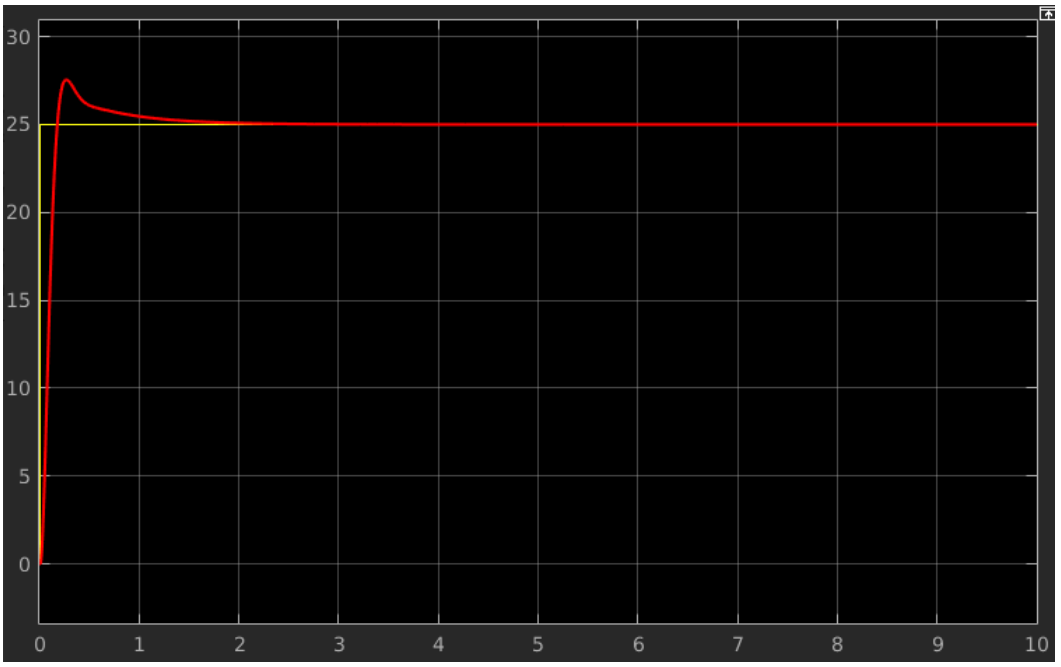
regionC = {3.3481, 5.2, 0.4}, regionD = {1.25, 1, 0.01}

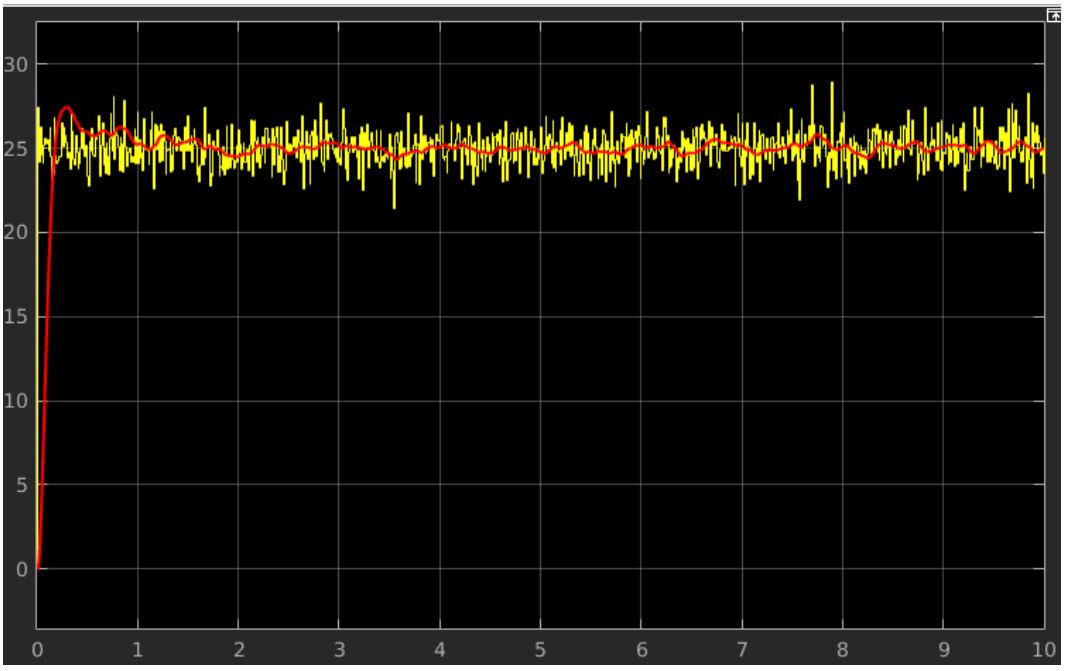
y los resultados fueron aceptables, teniendo en cuenta las imprecisiones de los sensores ultrasónicos. Para reducir estos errores, implementamos un filtro pasabajos denominado '*filtro media móvil exponencial*'. De todas formas, el compensador actúa rápido y sin oscilaciones relevantes.



Respuesta del sistema compensado para una entrada escalón

Dado que los resultados teóricos/estimados difieren levemente con la práctica, graficamos la respuesta al escalón de ambos casos con la ayuda de Simulink, agregando ruido blanco a la entrada escalón (que simula el sensor de la mano) para el caso práctico:

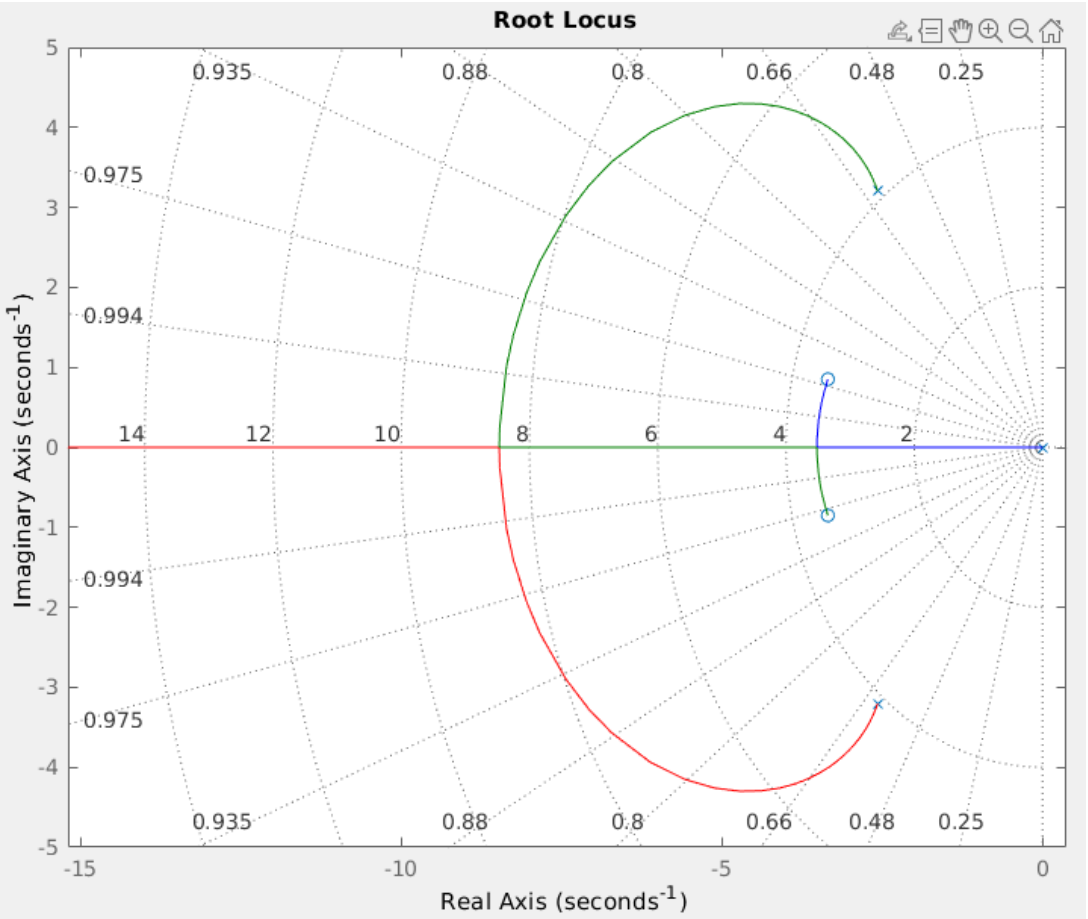




Para ambas gráficas, se utilizaron las constantes de la región C.

Lugar de Raíces

El siguiente gráfico muestra el lugar de raíces del sistema compensado $G_{OL}(s)$



donde encontramos 2 polos pertenecientes a la planta, 1 polo en el origen y dos ceros complejos conjugados pertenecientes al PID.

Análisis del error de régimen permanente

Teniendo en cuenta la FdT del sistema compensado,

$$G_{OL}(s) = \frac{89.56\,(s^2+3.3481s+5.9676)}{s(s^2+5.145\,s+16.88)}$$

vemos que se trata de un sistema de tipo 1, por lo tanto, podemos decir que

- Entrada escalón u(t):

$$k_p = \lim_{s\to 0} G_{OL}(s) = \infty \quad \rightarrow \quad ess = \frac{1}{1+kp} = 0$$

- Entrada rampa r(t)

$$k_v = \lim_{s\to 0} sG_{OL}(s) = \frac{89.56\,(0^2+3.3481x0+5.9676)}{(0^2+5.145x0+16.88)} = 49.43 \quad \rightarrow \quad ess = \frac{1}{49.43} = 0.02$$

- Entrada parábola p(t):

$$k_a = \lim_{s\to 0} s^2G_{OL}(s) = 0^2\frac{89.56\,(0^2+3.3481x0+5.9676)}{(0^2+5.145x0+16.88)} = 0 \quad \rightarrow \quad ess = \frac{1}{ka} = \infty$$

Análisis de estabilidad de Routh-Hurwitz

Para analizar si el sistema es estable, nuevamente utilizaremos el método de Routh

A partir del polinomio en el denominador de $G_{CL}(s)$, es decir,

$$p(s) = s^3 + 49.92\,s^2 + 316.7\,s + 534.5$$

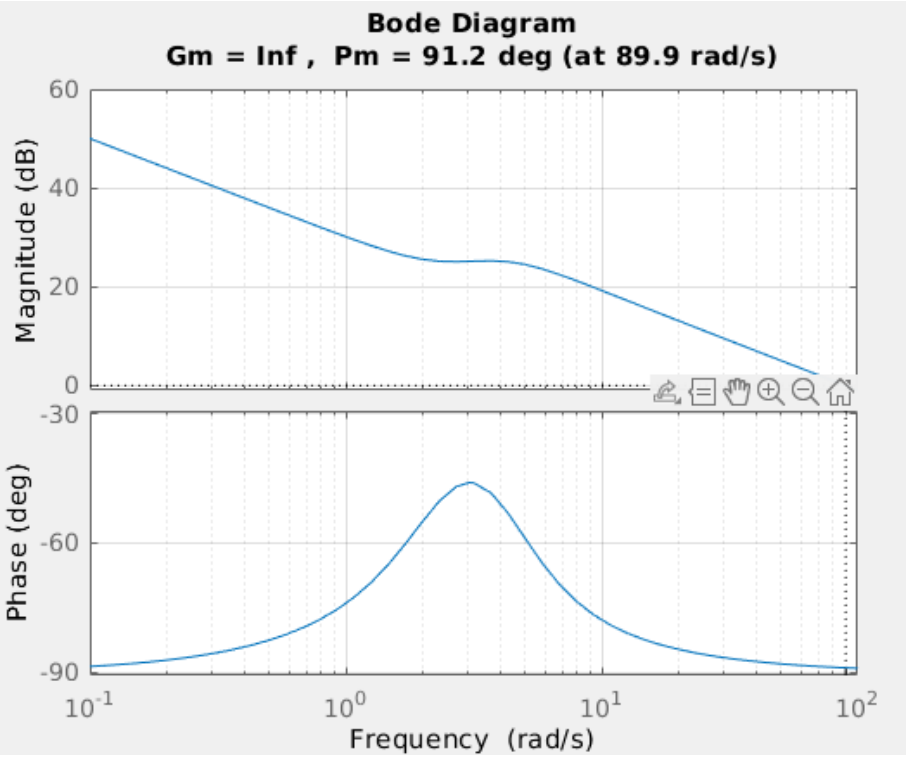
Analizamos si en la primera columna de la siguiente tabla hay cambio de signo

	a_0	a_2
s^3	1.0	3.167
	a_1	a_3
s^2	49.92	534.5
	b_0	b_1
s^1	305.99	0
	c_0	c_1
s^0	534.5	0

como no lo hay, entonces decimos que el sistema es estable.

Diagrama de Bode del sistem

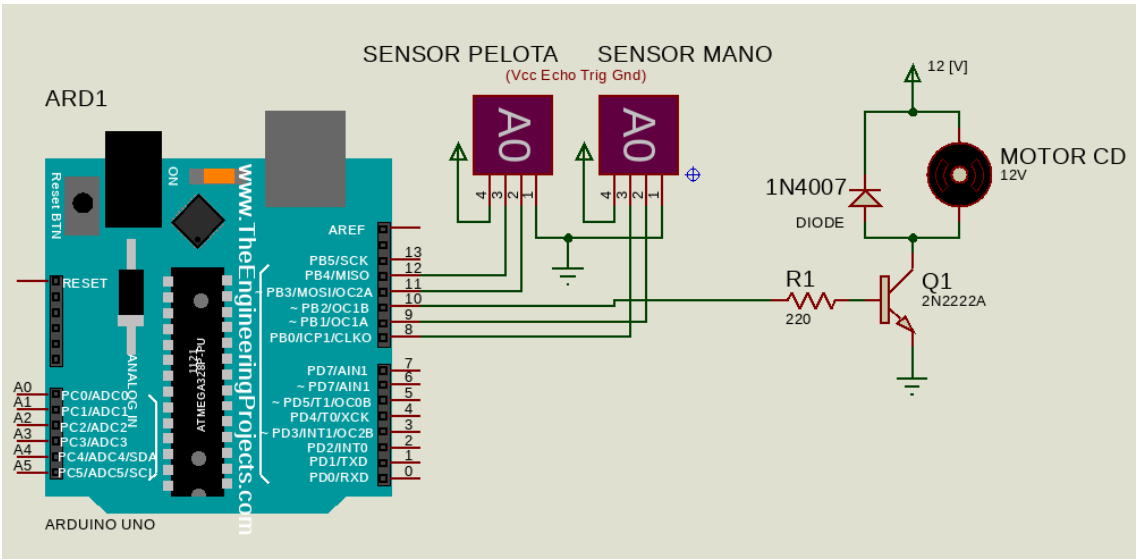
El diagrama de bode de la función de transferencia del sistema (a lazo abierto) es el siguiente:



donde encontramos que el margen de ganancia Gm es infinito y el margen de fase Pm es de 91.2° (o bien, 89.9 rad/s). Ambos factores son positivos, por lo cual podemos decir que el sistema es estable.

Esquema del circuito físico

En la siguiente figura se muestra las conexiones del circuito físico

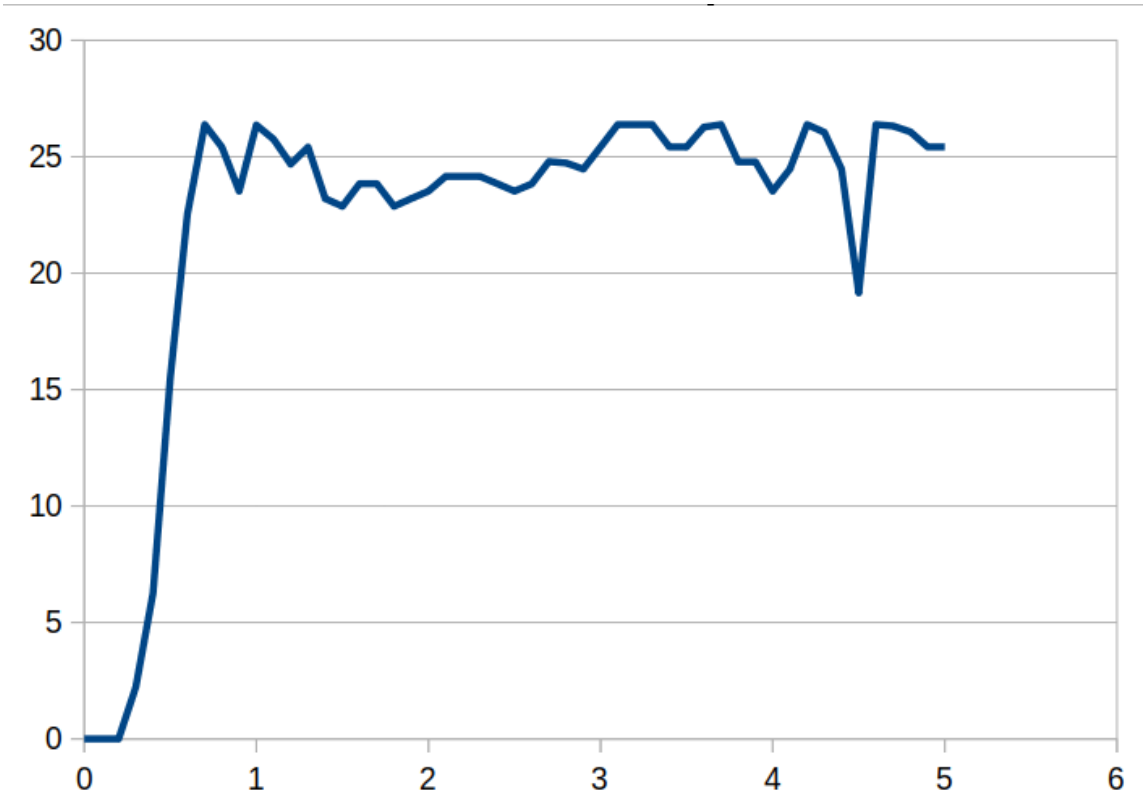


Aquí se puede encontrar el archivo en Proteus.

Es importante que los pines *trig* y el pin que controla el motor sean pines con salida PWM (~). Los sensores ultrasónicos se alimentan con 5 [V], mientras que el motor se alimenta con una fuente externa de 12 [V] / 1 [A].

Conclusión

Pudimos diseñar y calibrar un sistema LTI con los conocimientos adquiridos en la asignatura de Sistemas de Control. Aprendimos a obtener la función de transferencia de un sistema a lazo abierto de manera experimental, ya que en este caso nos encontramos con un sistema difícil de modelar matemáticamente. Logramos cumplir con los requerimientos de diseño propuestos. Además, aprendimos cómo funcionan, cómo implementar y cómo ajustar los controladores PID, por medio de software. A pesar de los inconvenientes con el hardware (sensores ultrasónicos) logramos resultados aceptables y la pelota logra seguir la referencia ingresada (altura de la mano), con una respuesta rápida, sin sobrepasamientos importantes y leves oscilaciones. Podemos observar en la siguiente figura que inicialmente se establece una altura de referencia de 25 cm. La pelota alcanza este punto a los 0,7 segundos, con un sobrepasamiento del 5.5%. En el régimen permanente, no oscila más de un 5.6%, lo cual es aceptable. Se ve que aparentemente la pelota decae aproximadamente a los 19 cm, pero como mencionamos anteriormente, es debido al error de los sensores.



Es importante mencionar que linealizar el sistema fue sumamente importante para conseguir tales resultados, ya que el aire que larga el soplador y eleva la pelota no es lineal. De esta manera, el sistema es más efectivo en la región B y C. Un mejor diseño físico podría llevar a mejores resultados.

Otro factor importante fue el agregado del filtro pasabajos, que ayudó a mejorar la señal de salida en el régimen permanente, ya que sin tener presente esto, el PID lee que la pelota baja o sube mucho y se generan oscilaciones indebidas.

Lo más difícil de calibrar fue la región D porque la pelota es más sensible a los cambios del motor. Un tubo más largo y/o una pelota más pesada podría solucionar este inconveniente.

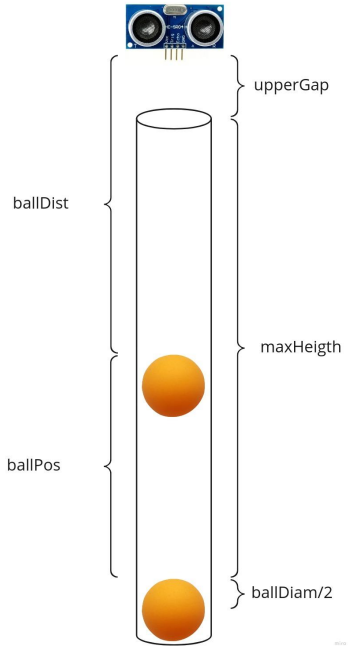
ANEXO

Obtención de datos para calcular la función de transferencia de la planta

Con Arduino energizamos la entrada con un escalón de 7,058 [V], equivalente a un PWM de 150. Esto se determinó en base a observaciones hechas sobre el armado físico, ya que con esa señal el tiempo en que la pelota tarda en subir desde el reposo hasta la altura máxima es considerablemente aceptable. Una señal PWM más grande hará que la pelota llegue más rápido a la altura máxima, pero luego presentará problemas para el diseño del compensador, ya que se presentan más oscilaciones.

Los pasos fueron los siguientes (Ver código en Arduino):

1. Medir la distancia entre el sensor y la pelota cuando esta se encuentra en la altura máxima (upperGap):
    a. Ingresamos una entrada escalón para posicionar la pelota en el tope del tubo. Luego, tomamos 5 mediciones de distancia y sacamos un promedio para establecer el upperGap. Hicimos esto teniendo en cuenta que el sensor presenta imprecisiones para medir distancias.
2. Medir la distancia máxima del tubo (maxHeight), o bien, la distancia que puede recorrer la pelota.
    a. Medimos la distancia desde el sensor hasta la pelota en reposo, y le restamos el upperGap y le sumamos el radio de la pelota. Así, obtenemos la distancia que puede recorrer la pelota al elevarse desde el reposo hasta el tope del tubo, tomando como referencia el punto medio de la pelota.
3. Ingresar una entrada escalón, con un valor de PWM de 150 (equivalente a 7,058 V), e imprimir por puerto serial la altura de la pelota, con un tiempo de muestreo de 20 ms.



$$maxHeight = ballDist - upperGap$$

$$ballPos = maxHeight + upperGap - ballDist - ballDiam/2$$

Repetimos 4 veces el paso 3 para tomar un promedio de los datos obtenidos y así reducir los errores de medición. Luego, estos datos obtenidos por el puerto serial se copian y pegan en una hoja de cálculos para luego leer la misma desde Matlab y hacer los cálculos correspondientes de la función de transferencia.

Implementación de PID en Arduino

Para implementar el controlador PID, utilizamos la librería <PID_v1.h> de Arduino. Para su uso, le asignamos una variable de entrada, una de salida, la referencia y las constantes Kp, Ki y Kd. Como alternativa, podemos indicar el tiempo de muestreo. En nuestro caso, utilizamos un Ts=50 ms. Además, presenta la alternativa de cambiar las constantes mientras el sistema está en funcionamiento. Esto nos ayudó a dividir las 3 regiones con sus respectivas constantes.

Dado que hay distintas formas de implementar un PID, esta librería es de código abierto e investigamos que funciona de la siguiente manera:

$$PID(s) = K_P + \frac{K_i}{s} + K_D s$$

Aquí se puede encontrar el código en Arduino del proyecto completo, que incluye el PID, sensores, entre otros.

Filtro de Media Móvil Exponencial

Un filtro de media móvil exponencial es un filtro pasabajos. La señal filtrada $f(t)$ presenta las siguientes condiciones

$$f(t) = f(0) \qquad si \quad t = 0$$

$$f(t) = \alpha f(t) + (1 - \alpha) f(t-1) \qquad si \quad t > 0$$

donde $\alpha \in (0, 1)$ es un coeficiente de suavizado, $f(t)$ es la señal a filtrar en un tiempo actual t y $f(t-1)$ es la señal filtrada en un instante $t - 1$ anterior. Esto tiene una pequeña desventaja, pues genera un retardo en la señal filtrada respecto a la original.

En nuestro caso, ajustamos el valor de α a prueba y error, hasta llegar a un equilibrio entre las imprecisiones del sensor y el tiempo de retardo, que a grandes rasgos, no es grande.