Diseño de Controlador PID

Nicolás Torres Muñoz
nitorresm@udistrital.edu.co
20201005046
Universidad Distrital Francisco José de Caldas
Facultad de Ingeniería
Bogotá D.C, Colombia

Mateo Salamanca Pulido
matsalamancap@udistrital.edu.co
20211005107
Universidad Distrital Francisco José de Caldas
Facultad de Ingeniería
Bogotá D.C, Colombia

I. Introducción

Se propone el proceso de diseño de un controlador PID para el modelo matemático de la velocidad angular del motor trabajado durante el laboratorio mediante los métodos de ubicación geométrica de las raíces, respuesta en frecuencia con diagrama de Bode, Ziegler-Nichols, optimización computacional y diseño de PID por Simulink.

II. Solución

La planta del motor trabajado en el laboratorio es la siguiente:

$$\frac{\omega}{V_{PWM}} = \frac{25560}{s^2 + 222.2s + 4938} \tag{1}$$

A. Método por ubicación geométrica de las raíces, R-Locus:

1) Graficar la ubicación geométrica de las raíces del sistema del motor, y ubicar el punto donde el *overshoot* sea igual al 20%:

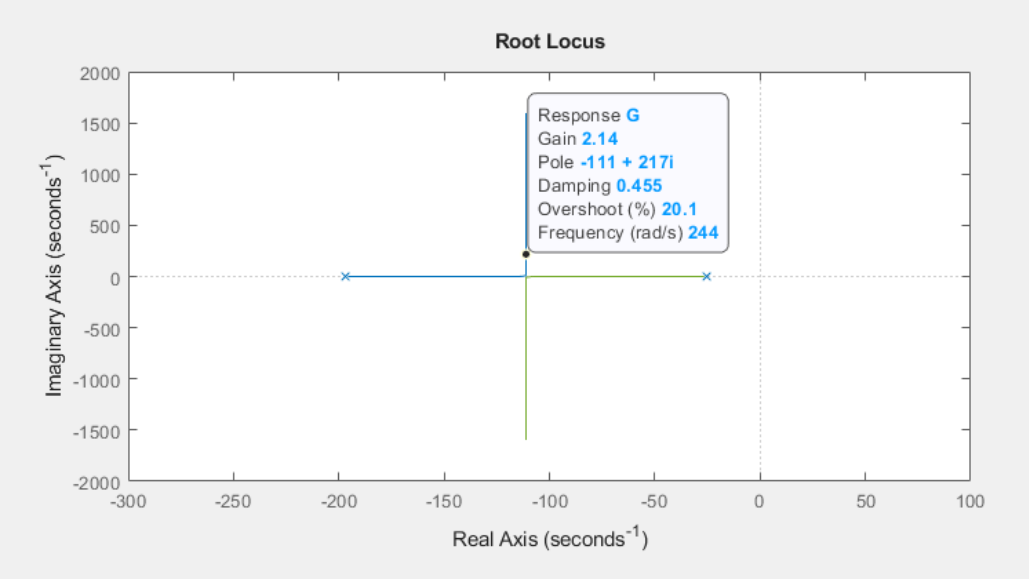

Fig. 1. Ubicación geométrica de las raíces de la función de transferencia del sistema sin compensación. *Autoría Propia.*

Obteniendo así una ganancia proporcional del sistema de $2,14$ y un polo dominante $s_D = -111 \pm \text{j}217$.

2) Determinar el tiempo de establecimiento del sistema sin compensación:

$$\begin{aligned} t_s &= \frac{4}{\Re(s_D)} \\ t_s &= \frac{4}{111} \approx 0,036\,\text{s} \end{aligned} \tag{2}$$

3) Calcular la parte real del polo requerido para cumplir con el parámetro de reducción del tiempo de establecimiento a $\frac{2}{3}$ del tiempo del sistema en lazo cerrado:

$$\begin{aligned} \sigma_r &= \frac{4}{\frac{2}{3} \cdot t_s} \\ \sigma_r &= \frac{4}{\frac{2}{3} \cdot 0,036} \approx 166,6667 \end{aligned} \tag{3}$$

4) Calcular la parte imaginaria del polo a partir del ángulo que forma con el eje horizonal:

$$\begin{aligned} \beta &= \tan^{-1}\left(\frac{\omega_n \cdot \sqrt{1-\zeta^2}}{\omega_n \cdot \zeta}\right) \\ \beta &= \tan^{-1}\left(\frac{244 \cdot \sqrt{1-(0,455)^2}}{244 \cdot 0,455}\right) \\ \beta &\approx 62,9351 \end{aligned} \tag{4}$$

$$\begin{aligned} \omega_r &= \sigma \cdot \tan(\beta) \\ \omega_r &= 166,6667 \cdot \tan(62,9351) \\ \omega_r &\approx 326,1879 \end{aligned} \tag{5}$$

5) Ubicar en el plano s los polos del sistema sin compensador y el polo requerido, y determinar los ángulos internos:

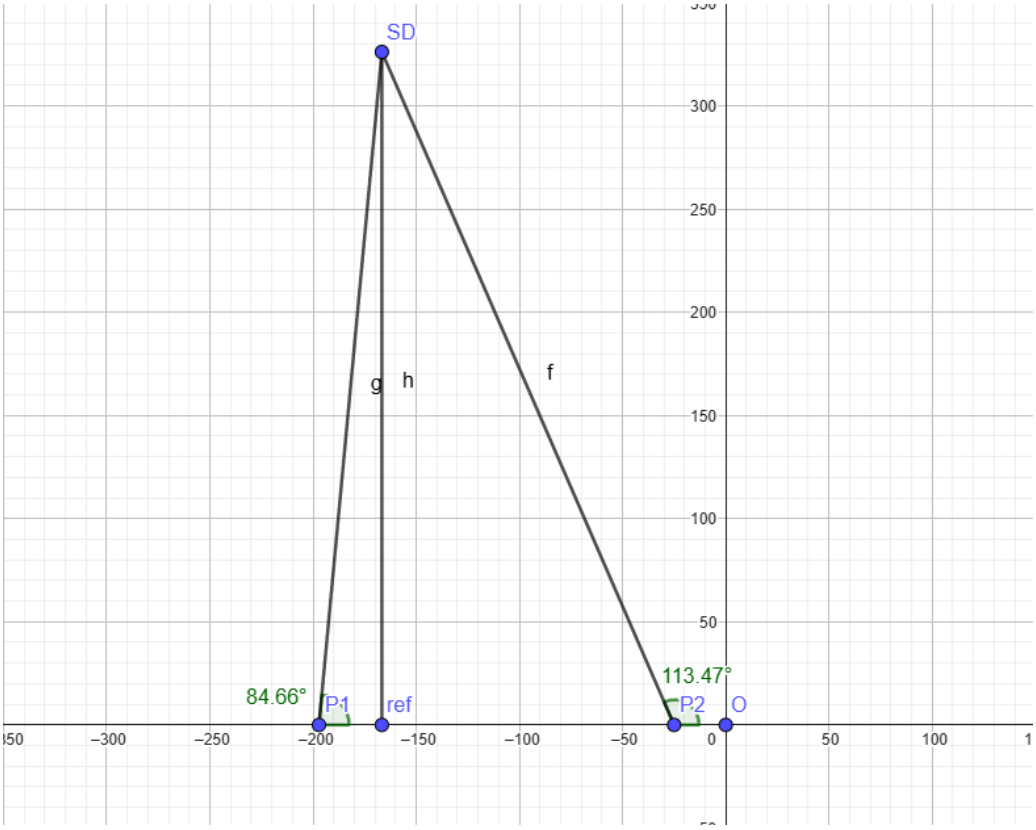

Fig. 2. Ubicación geométrica de los polos del sistema y el polo requerido. *Autoría Propia.*

6) Calcular la suma de los ángulos de la figura 2:

$$\begin{aligned} \angle P &= 113,47 + 84,66 \\ \angle P &= 198,13 \end{aligned} \tag{6}$$

Por lo tanto, el ángulo que se debe formar entre el eje horizontal y el eje requerido es de $198,13 - 180 = 18,13$

7) Calcular el cero requerido en el sistema, equivalente a la función del controlador PD:

$$\begin{aligned} \frac{\omega_r}{\sigma_r - \sigma_{PD}} &= \tan(180 - \angle P) \\ \sigma_{PD} &= \sigma_r - \frac{\omega_r}{\tan(180 - \angle P)} \\ \sigma_{PD} &= 166,6667 - \frac{326,1978}{\tan(180 - 198,13)} \\ \sigma_{PD} &\approx 1162,9034 \end{aligned} \tag{7}$$

Con lo cual se determina que la etapa derivativa del controlador es $s + 1162,9034$

8) Graficar la ubicación geométrica de los polos de la planta con el compensador derivativo en lazo abierto:

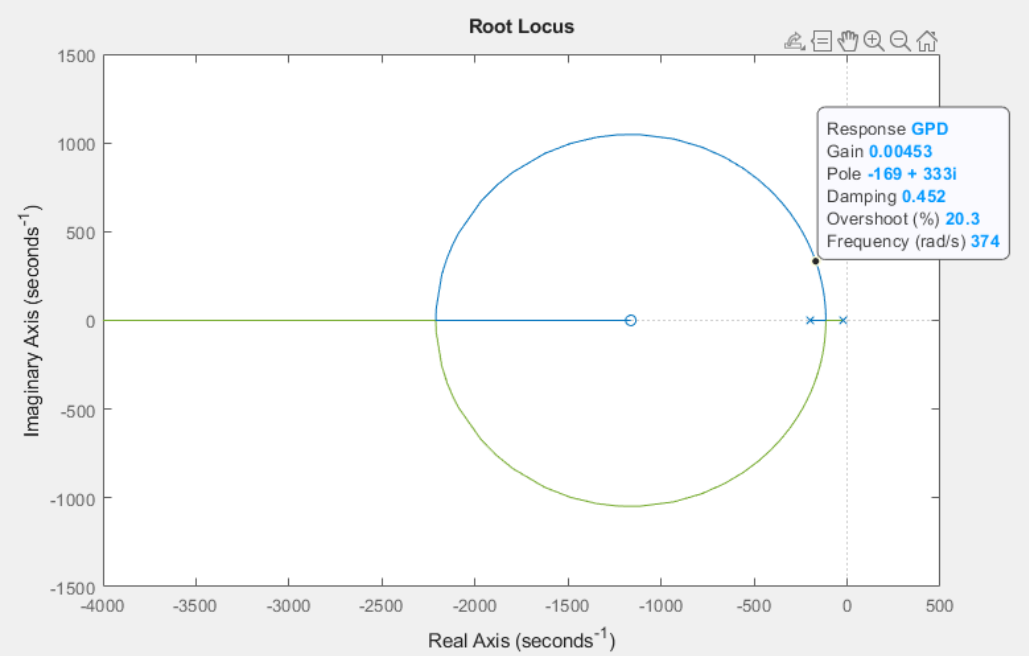

Fig. 3. Ubicación geométrica de los polos del sistema con compensador derivativo en lazo abierto. *Autoría Propia.*

De allí, se ubica el punto que cumpla con el parámetro de diseño del *overshoot*, adicionalmente, el polo de dicho punto debe coincidir con el calculado en los pasos 3 y 4 para garantizar el parámetro del tiempo de establecimiento; obteniendo así, una ganancia proporcional de $0,0045$.

9) Comparar la respuesta al escalón de la planta con realimentación y ganancia determinada en el paso 1, con la respuesta al escalón del sistema compensado con ganancia determinada en el paso 8:

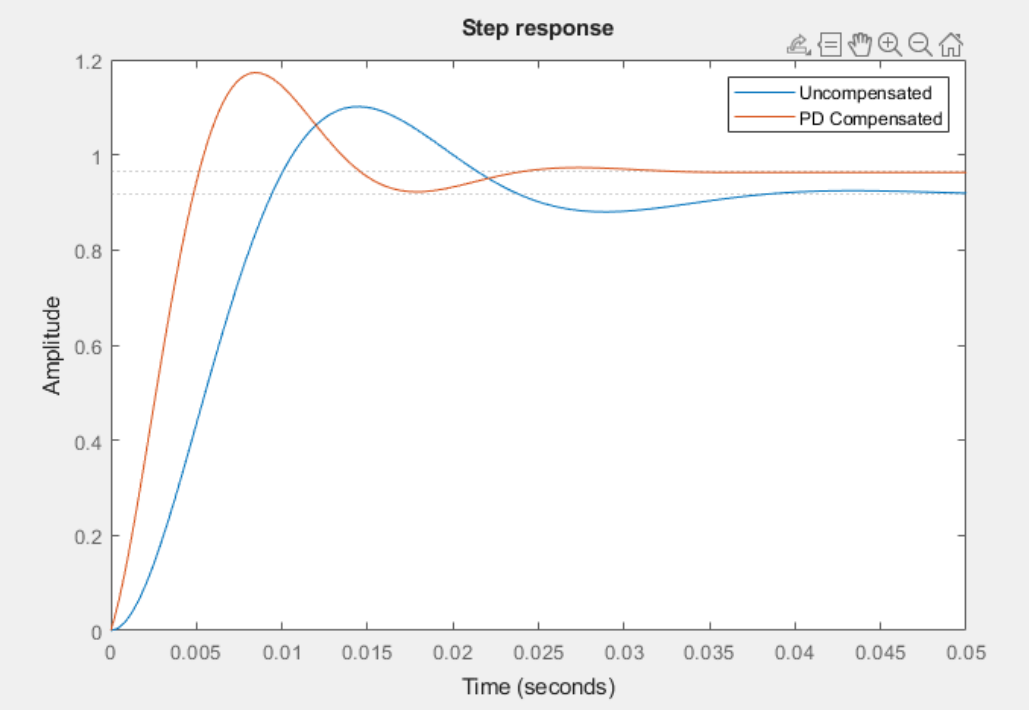

Fig. 4. Respuesta al escalón del sistema en lazo cerrado y con compensación. *Autoría Propia.*

10) Para la etapa de compensación integral, se agrega al sistema un cero libremente cerca al origen y un polo en el origen, por lo cual la función de transferencia del compensador integral es:

$$\text{tf}(PI) = \frac{s + 0,5}{s} \tag{8}$$

11) Seguidamente, se grafica la ubicación geométrica de las raíces del sistema con los compensadores calculados para determinar la ganancia que garantice el parámetro de diseño del *overshoot*:

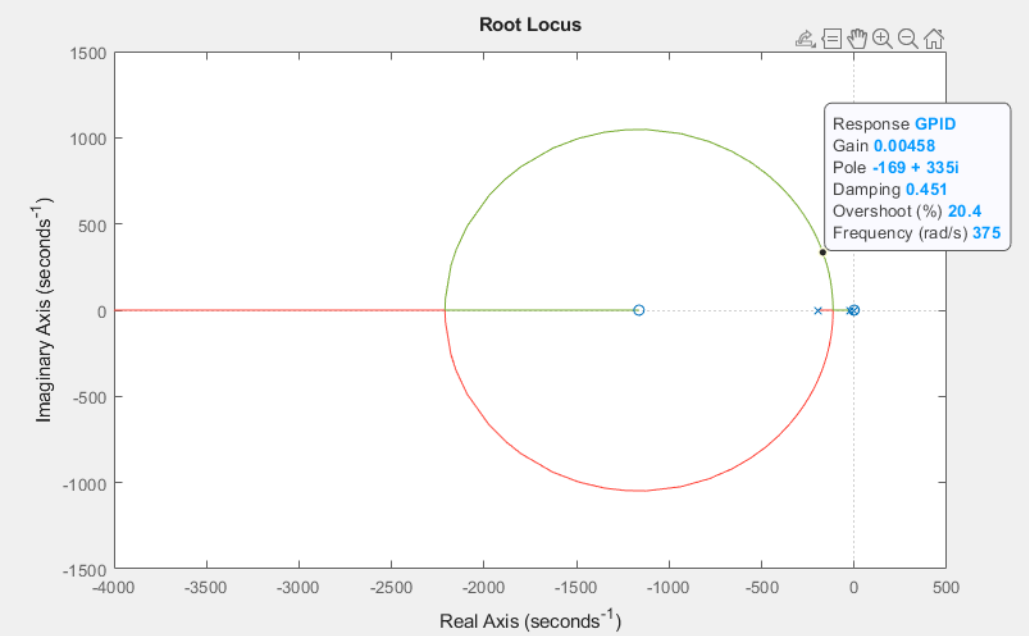

Fig. 5. Ubicación geométrica de las raíces del sistema con compensador PID en lazo abierto. *Autoría Propia.*

De allí, se determina que la ganancia proporcional debe ser $0,00458$.

12) Finalmente, se obtiene que el siguiente sistema con compensador PID posee la siguiente siguiente respuesta al escalón unitario:

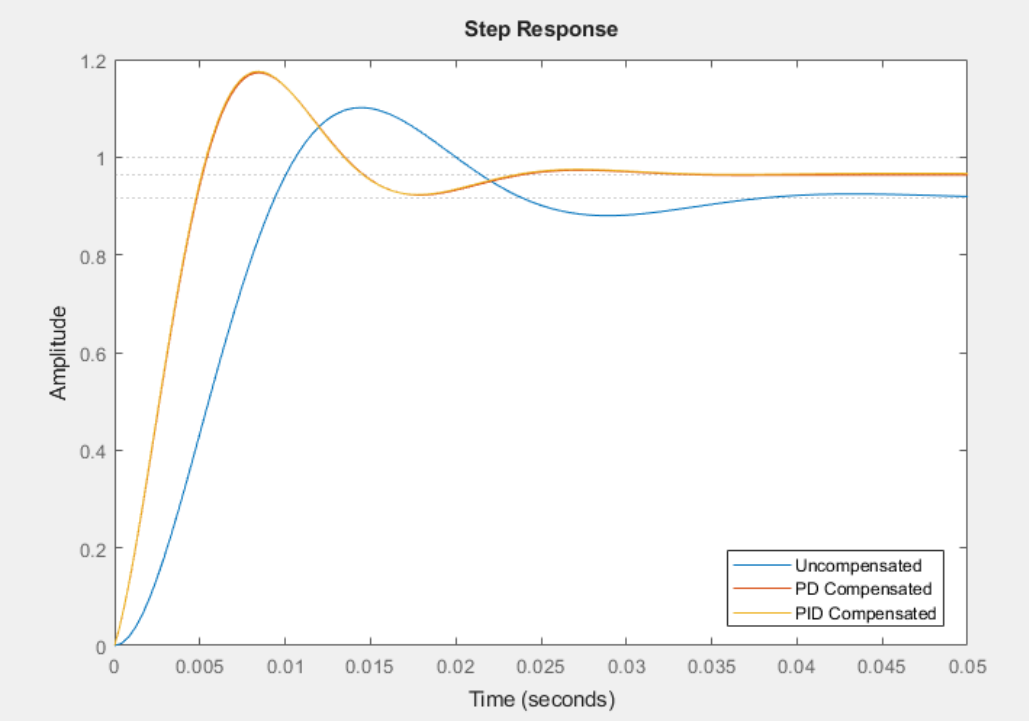

Fig. 6. Respuesta al escalón unitario del sistema sin compensador, y con compensadores PD y PID. *Autoría Propia.*

Las gráficas realizadas que contribuyeron al desarrollo del diseño del controlador PID por este método, se realizaron en el software de MATLAB con el siguiente código:

```
% Modelo matematico, funcion de transferencia
clc;

num = 2.556e04;
den = [1 222.2 4938];
G = tf(num,den)
[fp,p,m] = residue(num,den)
roots(den)

%% R-locus
close all;

k = 2.14;
subplot(2,2,1)
pzmap(G);
subplot(2,2,2)
rlocus(G);
subplot(2,2,3)
impulse(G);
subplot(2,2,4)
step(G);

%% Feedback
close all;

G_fd = feedback(k*G,1)
subplot(2,2,1)pzmap(G_fd)
subplot(2,2,2)
rlocus(G_fd)
subplot(2,2,3)
impulse(G_fd)
subplot(2,2,4)
step(G_fd)

%% Compensador PD
clc; close all;

PD = tf([1 1162.9034],1)
GPD = G*PD
rlocus(GPD)

%% Comparaci n del sistema sin compensaci n y con PD
close all;

k_pd = 0.00453;
step(G_fd)
hold on
step(feedback(GPD*k_pd,1))
legend("Uncompensated","PD Compensated")

%% Compensador PID
clc; close all;

k_pid = 0.00458;
PI = tf([1 0.5],[1 0])
GPID = GPD*PI
rlocus(GPID)

%% Sistema con compensador PID
clc; close all;

F = feedback(k_pid*GPID,1)

step(G_fd)
hold on;
step(feedback(GPD*k_pd,1))
hold on;
step(F)
hold on;
legend("Uncompensated","PD Compensated", "PID Compensated")
```

B. Método de respuesta en frecuencia con diagrama de Bode

Para el diseño del controlador con el método de respuesta en frecuencia, se asume una constante de error estático 5 seg-1 y los parámetros impuestos anteriormente, tener un $OS = 20\%$ es equivalente a un $\zeta = 0.456$, con esto calculamos el margen de fase como parámetro para el diseño de la siguiente manera:

$$\phi_M = \arctan \frac{2\zeta}{\sqrt{-2\zeta^2 + \sqrt{1 + 4\zeta^4}}} \tag{9}$$

$$\phi_M = 48.15^\circ \tag{10}$$

El modelo de controlador PID escogido es el siguiente:

$$G_c(s) = \frac{K(as+1)(bs+1)}{s} \tag{11}$$

Para poder hallar la constante K, se hace uso del parámetro de K_v, su respectiva ecuación es:

$$K_v = \lim_{s \to 0} sG_c(s)G(s) \tag{12}$$

Del cálculo anterior se obtiene:

$$K_v = \lim_{s \to 0} s\frac{K(as+1)(bs+1)}{s}\frac{25560}{s^2 + 22.2s + 4938} = K\frac{25560}{4938} \tag{13}$$

$$5 = K\frac{25560}{4938} \tag{14}$$

$$K = 0.965 \tag{15}$$

Por lo tanto:

$$G_c(s) = \frac{0.965(as+1)(bs+1)}{s} \tag{16}$$

Primero se hace el diagrama de bode para ver el comportamiento sin los terminos de a y b, es decir:

$$G(s) = \frac{0.965 * 25560}{s(s^2 + 22.2s + 4938)} \tag{17}$$

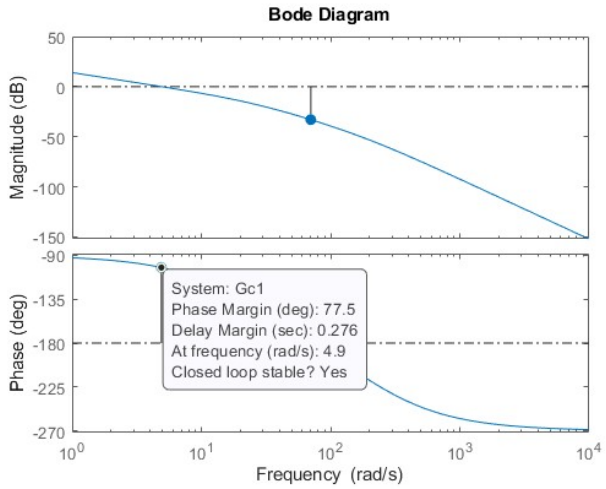

Fig. 7. Diagrama de Bode G(s) sin parámetros a y b

Del diagrama de Bode se observa que la frecuencia de cruce de ganancia es $w = 3,95$ rad/s. Se asume un valor de a =70. Se realiza el diagrama de bode de la nueva función de transferencia G(s):

$$G(s) = \frac{0.965 * (70s + 1) * 25560}{s(s^2 + 22.2s + 4938)} \tag{18}$$

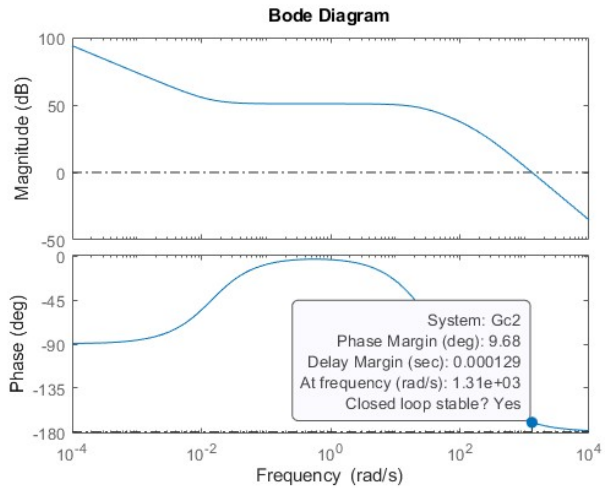

Fig. 8. Diagrama de Bode de G(s) con la influencia del parámetro a

Basándose en el diagrama de Bode de la Figura 8 se escoge el valor de b. El término $(bs + 1)$ tiene que dar el margen de fase de al menos 48.15°. Mediante simples tanteos en MATLAB se comprueba que $b = 0.00075$ proporciona un margen de fase de al menos 48.15°. Por tanto, seleccionando $b = 0.00075$ se obtiene:

$$G_c(s) = \frac{0.965(70s + 1)(0.00075s + 1)}{s} \tag{19}$$

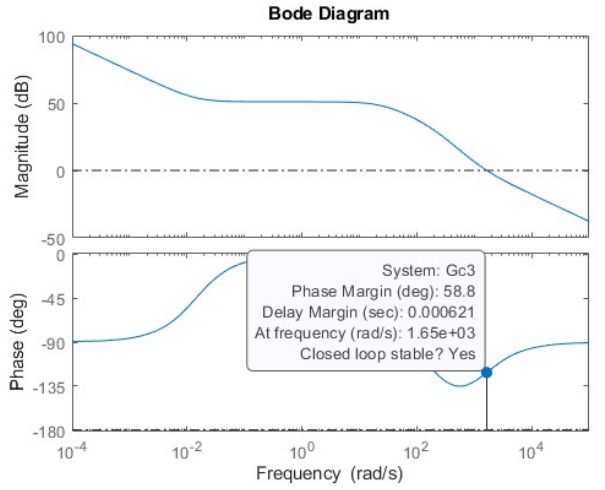

Fig. 9. Diagrama de Bode G(s) con parámetros a y b

Como se observa en el diagrama de Bode de la figura 9, tenemos un error estático en velocidad de $5seg^-1$ y un margen de fase de 58.8° que cumple con ser mayor al requerido.Por lo tanto, el sistema diseñado satisface todas las especificaciones y se puede considerar aceptable. Luego se obtiene la respuesta a un escalón unitario, con la función de transferencia en lazo cerrado:

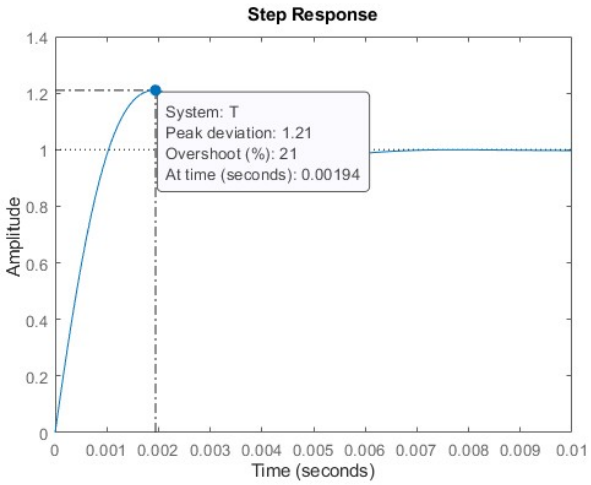

Fig. 10. Respuesta a escalón unitario del sistema con controlador

C. *Método de Ziegler-Nichols*

Inicialmente, se graficó mediante MATLAB la respuesta al escalón del sistema dado en lazo abierto, y se trazó la recta tangente a la curva en el punto de inflexión de la respuesta:

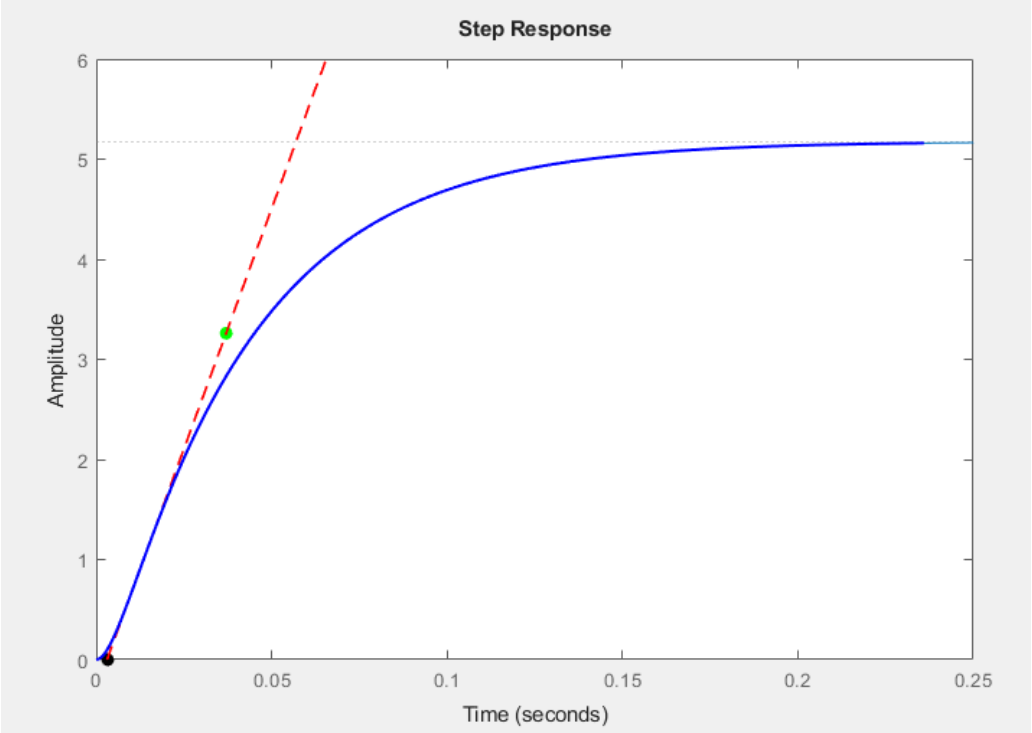

Fig. 11. Respuesta al escalón de la planta con recta tangente a la curva en el punto de inflexión. *Autoría Propia.*

A partir de la anterior gráfica, es posible determinar los parámetros de tiempo de delay L y la constante de tiempo T:

$$\begin{aligned} L &= 0,0033 \\ T &= 0,037 \end{aligned} \tag{20}$$

Ahora bien, siguiendo con el diseño dado por el autor Ogata en la quinta edición de su libro *Ingeniería de Control Moderno*, las constantes de un compensador PID a calcular son:

$$\begin{aligned} K_p &= 1,2 \cdot \frac{T}{L} = 1,2 \cdot \frac{0,037}{0,0033} \approx 13,4545 \\ T_i &= 2L \approx 0,0066 \\ T_d &= 0,5L \approx 0,0017 \end{aligned} \tag{21}$$

De esta forma, la función de transferencia de la planta del sistema puede ser aproximada a un sistema de primer orden de la siguiente forma:

$$\frac{C(s)}{U(s)} = \frac{Ke^{-Ls}}{Ts+1} \tag{22}$$

Donde K es el valor en estado estable de la planta.

$$\frac{C(s)}{U(s)} = \frac{5,18e^{-0,0033s}}{0,037s+1} \tag{23}$$

Finalmente, la función de transferencia del controlador PID ajustado para el primer método de Ziegler-Nichols es la siguiente:

$$\begin{aligned} G_c(s) &= K_p \left(1 + \frac{1}{T_i s} + T_d s\right) \\ G_c(s) &= 1,2 \cdot \frac{T}{L} \left(1 + \frac{1}{2Ls} + 0,5s\right) \\ G_c(s) &= 0,6T \frac{\left(s + \frac{1}{L}\right)^2}{s} \\ G_c(s) &= 0,6(0,037) \frac{(s+303,03)^2}{s} \\ G_c(s) &= 0,0222 \cdot \frac{(s+303,03)^2}{s} \end{aligned} \tag{24}$$

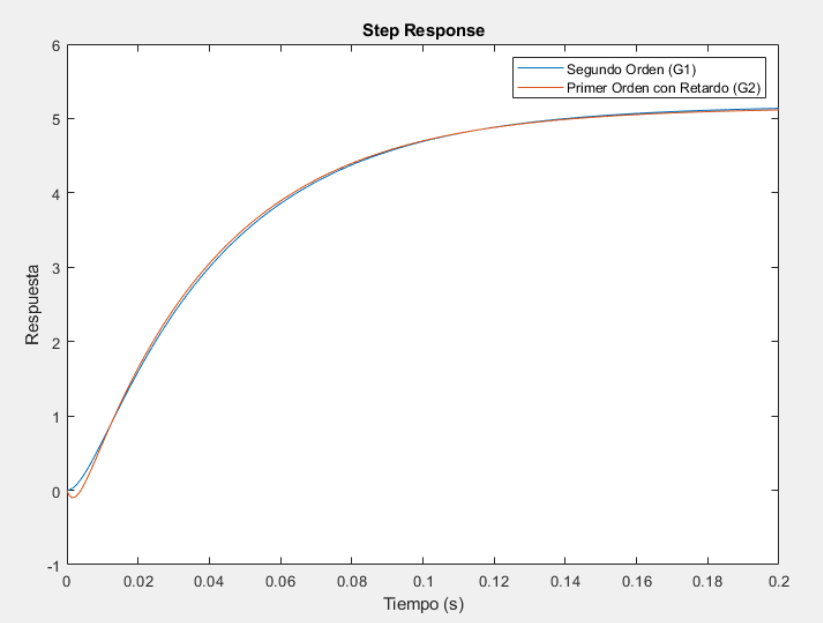

Fig. 12. Comparación de respuesta al impulso del sistema de segundo orden, y de primer orden con el retado calculado. *Autoría Propia.*

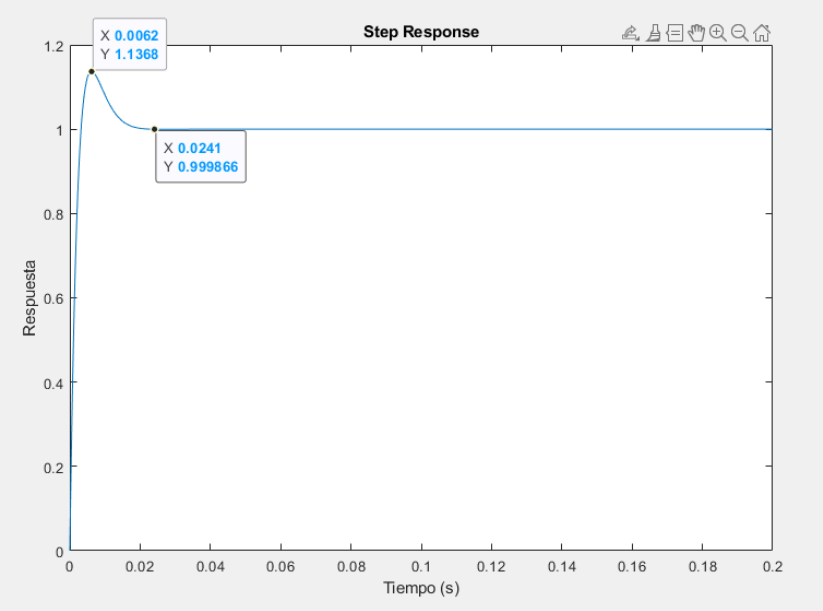

Fig. 13. Respuesta al impulso del sistema con controlador diseñado por método 1 de Zigler-Nichols. *Autoría Propia.*

Para realizar este diseño, se implementaron las siguientes líneas de código en MATLAB:

```
%% Step Response

close all;

num = 25560;
den = [1 222.2 4938];
G = tf(num, den);

[y, t] = step(G);

dy_dt = diff(y) ./ diff(t);
[max_slope, idx] = max(dy_dt);
t_inflexion = t(idx);
y_inflexion = y(idx);

tangent = max_slope * (t - t_inflexion) + y_inflexion;

idx_L = find(tangent >= 0, 1);
L = t(idx_L);

y_final = y(end);
T_val = (y_final * 0.632 - y_inflexion) / max_slope + t_inflexion;

figure;
plot(t, y, 'b', 'LineWidth', 1.5); hold on;
plot(t, tangent, 'r--', 'LineWidth', 1.2);
plot(L, 0, 'ko', 'MarkerFaceColor', 'k');
plot(T_val, y_final*0.632, 'go', 'MarkerFaceColor', 'g');
xlabel('Tiempo (s)');
ylabel('Respuesta');
title('Respuesta al Escal n y M todo de Ziegler-Nichols');
legend('Respuesta al Escal n', 'Tangente', 'L', 'T');
grid on;

fprintf('L (Tiempo muerto): %.4f s\n', L);
fprintf('T (Constante de tiempo): %.4f s\n', T_val);
step(G)

%%

clc; close all
GCP = tf(0.0234*([1 400 40000]),[1 0])
```

```
43
44      K = 5.15;
45      L = 0.005;
46      T = 0.039;
47      G2 = tf(K, [T 1]);
48
49      [num_pade, den_pade] = pade(L, 1);
50      Pade_tf = tf(num_pade, den_pade);
51
52      G2_delayed = G2 * Pade_tf;
53
54      t = 0:0.0001:0.2;
55      [y1, t1] = step(G1, t);
56      [y2, t2] = step(G2_delayed, t);
57      [y3, t3] = step(feedback(GCP*G1,1), t);
58      [y4, t4] = step(5.15*feedback(GCP*G1,1), t);
59
60      figure;
61      plot(t1, y1);
62      hold on;
63      plot(t2, y2);
64      % plot(t4, y4, 'Y', 'LineWidth', 1);
65      xlabel('Tiempo (s)');
66      ylabel('Respuesta');
67      title('Step Response');
68      legend('Segundo Orden (G1)', 'Primer Orden con Retardo (G2)');
69
70      figure;
71      plot(t3, y3);
72      xlabel('Tiempo (s)');
73      ylabel('Respuesta');
74      title('Step Response');
```

D. Optimización Computacional

Sea el sistema que se muestra en la Figura 8-19 que está controlado por un controlador PID. El controlador PID está dado por:

$$G_c(s) = K\frac{(s+a)^2}{s} \tag{25}$$

Se busca encontrar combinaciones de K y a tal que el sistema cumpla con los requerimientos necesarios, para este ejemplo los requerimientos son que tenga un $OS = 20\%$ y su tiempo de establecimiento menor a 1 segundo.
Para lograr los requerimientos tomamos los siguientes rangos para K y a:

$$1 \leq K \leq 10 \qquad\qquad 0.1 \leq a \leq 5 \tag{26}$$

Ahora se configura el código de tal manera que obtengamos los valores de K y a, que cumplan con los requisitos. Para este caso K tendrá saltos de 0,1 y a tendrá saltos de 0.2.
A continuación, se muestra el código usado:

```
1       clear; clc;
2
3       % Definir rangos para K y a
4       K_values = 1:0.1:10;
5       a_values = 0.1:0.2:5;
6       t = 0:0.001:10;          % Tiempo de evaluaci n
7       g = tf([25560],[1 222.2 4938]);
8       % Inicializar matriz para soluciones
9       solution = [];
10
11      % Iterar valores de K y a
12      for K = K_values
13          for a = a_values
14              % Definir funci n de transferencia en lazo cerrado
15              gc = tf(K*[1 2*a a^2],[1 0]);
16              G = gc*g/(1+gc*g);
17
18              %Respuesta al escal n
19              y = step(G, t);
20
```

```
                % Calcular tiempo de establecimiento (ts)
                s = length(t);
                while s > 1 && y(s) > 0.98 && y(s) < 1.02
                s = s - 1;
            end
            ts = (s - 1) * 0.001; % Convertir a segundos

            % Calcular el overshoot
            OS = (max(y) - 1) * 100; % Overshoot en porcentaje

            % Verificar si cumple con los criterios
            if  ts < 1 && OS < 22 && OS > 17
                solution = [solution; K, a,OS, ts];
            end
        end
    end

    % Mostrar soluciones encontradas
    if isempty(solution)
    disp('No hay soluciones para este requerimiento.');
    disp('Intenta modificar los rangos de K y a.');
    else
    disp('Soluciones que cumplen con los requisitos de dise o:');
    disp('   K          a        Overshoot(%)    Ts (s)');
    disp(solution);
    sortsolution = sortrows(solution,3);
    disp(sortsolution);

    end
```

El código anterior entrega 3 soluciones posibles:

```
Soluciones que cumplen con los criterios de diseño:
   K          a        Overshoot(%)    Ts (s)
    1.8000    4.9000    17.3596    0.9430
    1.7000    4.9000    17.8651    0.9660
    1.6000    4.9000    18.4043    0.9910
```

Fig. 14. Soluciones entregadas por el programa

Se optó por tomar la solucion 3, debido a que el *overshoot* es más cercano al requerido de 20%, en la siguiente gráfica se muestra el resultado de tomar esos datos para el controlador, donde se evidencia que los datos son correctos:

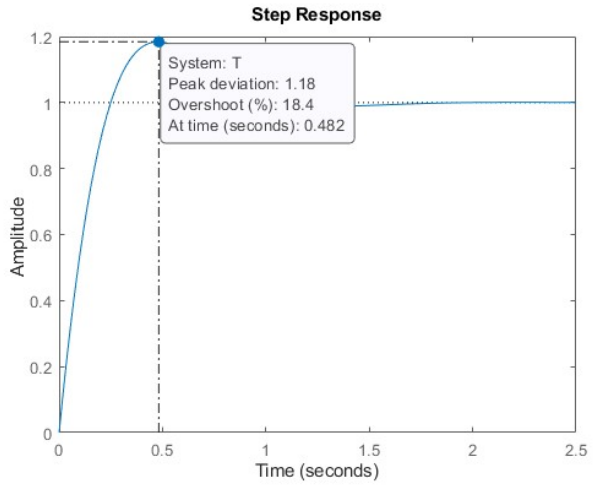

Fig. 15. Respuesta al escalón del sistema con la solución del programa

E. PID con Simulink

Para este método, se utilizó el bloque PID de Simulink, el cual requiere los parámetros mostrados a continuación:

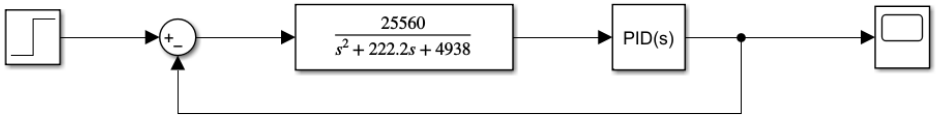

Fig. 16. Esquema del sistema con controlador PID en Simulink. *Autoría Propia.*

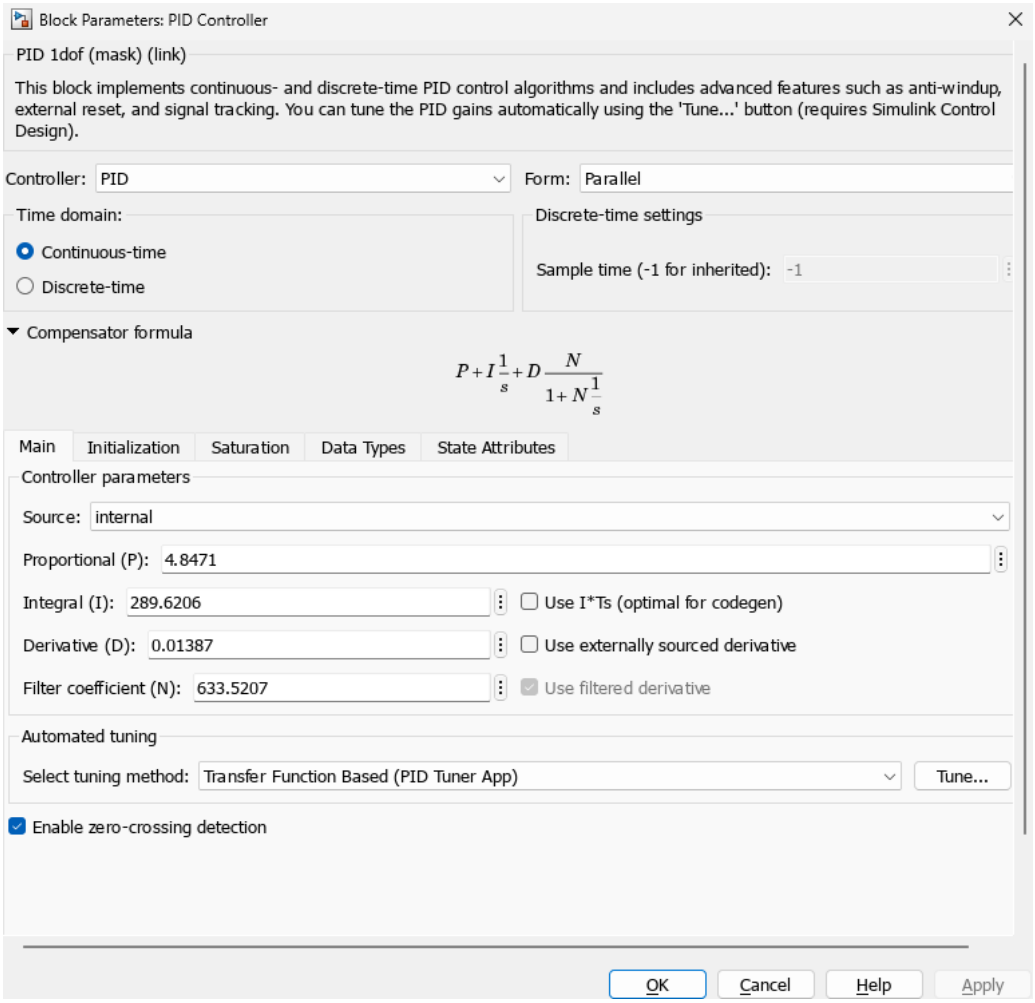

Fig. 17. Parámetros del controlador PID desde simulink. *Autoría Propia.*

Las constantes de proporción, integración y derivación se encuentran parametrizadas por la ecuación mostrada en la figura anterior, por lo cual es necesario usar la herramienta *Tune*, la cual permite calcular estas constantes variando el tiempo y comportamiento de respuesta. Estas se determinaron como se muestra en las siguientes figuras:

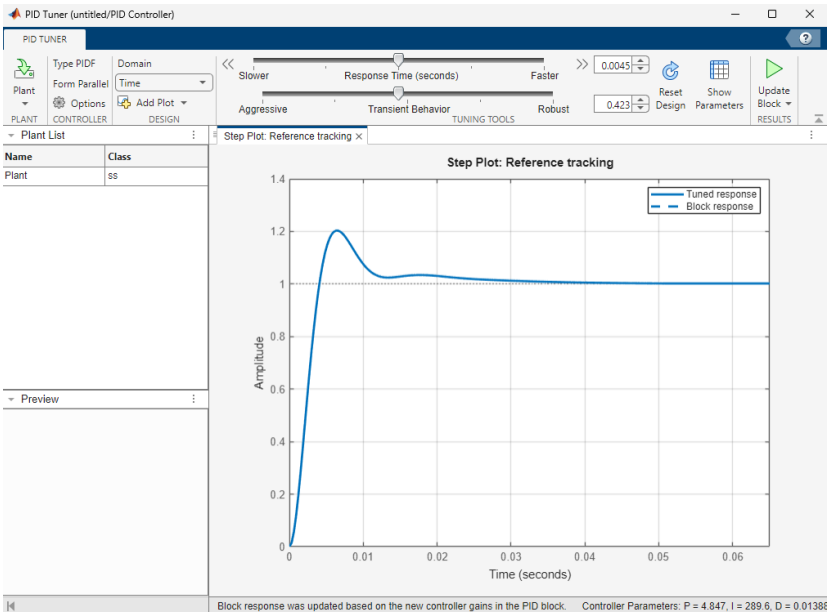

Fig. 18. Variación del tiempo y comportamiento de la respuesta al escalón del sistema. *Autoría Propia.*

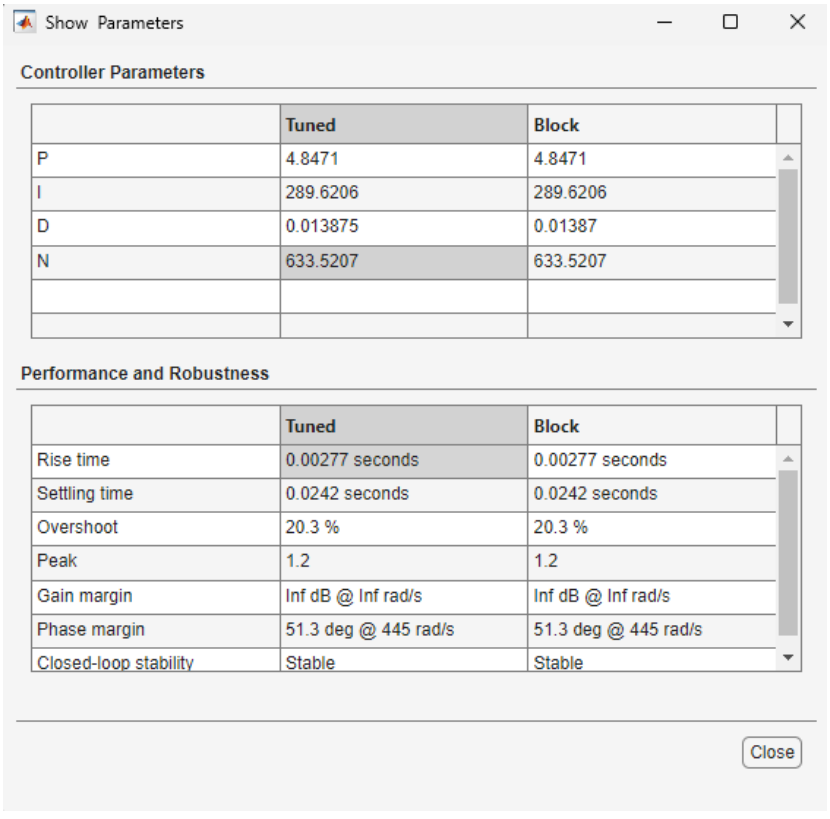

Fig. 19. Constantes del controlador PID de Simulink. *Autoría Propia.*

Ahora bien, se compara la respuesta al escalón del sistema con compensador obtenido mediante ubicación geográfica de las raíces con la obtenida mediante la herramienta *Tune* del PID desde Simulink:

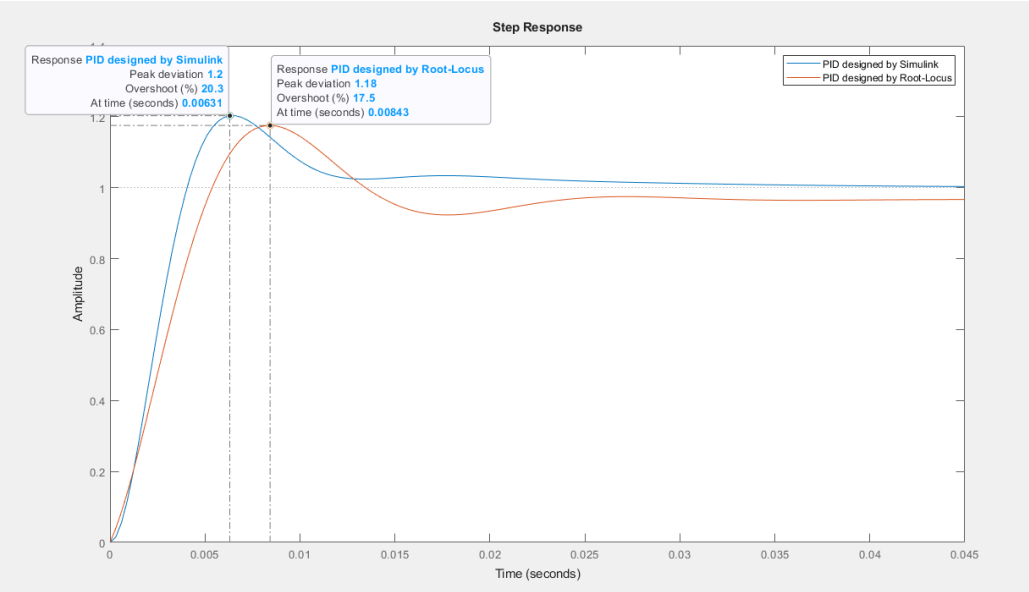

Fig. 20. Respuesta al impulso del sistema con compensación mediante ubicación geométrica de las raíces, y mediante PID de Simulink. *Autoría Propia.*

Para obtener la gráfica de la figura 20, se agregaron las siguientes líneas de código en MATLAB:

```
%% PID Simulink
clc; close all;

s = tf("s");
PID = 4.8471 + (1/s)*289.6206 + (0.013875)*(633.5207/(1+633.5207*1/s))
GPIDSIM = G*PID
step(feedback(GPIDSIM,1))
hold on;
step(F)
legend("PID designed by Simulink", "PID designed by Root-Locus")
```

Adicionalmente, se le agregó un bloque de saturación al sistema con valores entre ± 10.7, para limitar la señal de entrada a los valores de voltaje máximos que recibe el puente H, con el fin de dar un acercamiento a la realidad desde la simulación:

Fig. 21. Aproximación del sistema del motor al realidad con saturador. *Autoría Propia.*