

TALLER ESCRITO 3ER CORTE

JOSE ROLDAN

b) y c)

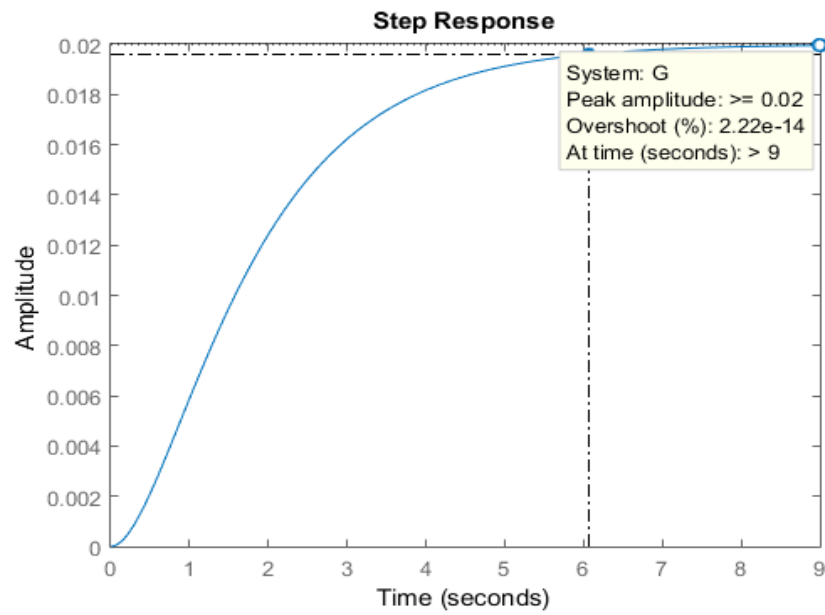


Figura 1. %os en malla abierta.

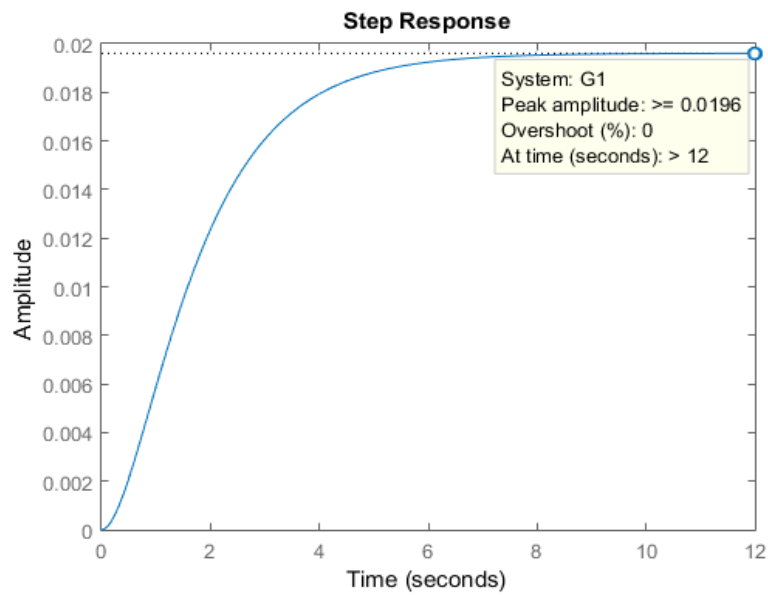


Figura 2. %os en malla cerrada.

El cambio en el overshoot es notable ya que en malla abierta es de $2.2 \times 10^{-14} \%$ y en malla cerrada es del 0% con un error del 100%, además el valor final en malla abierta es de 0.02 y en malla cerrada es de 0.0196 con un error del 2.04%.

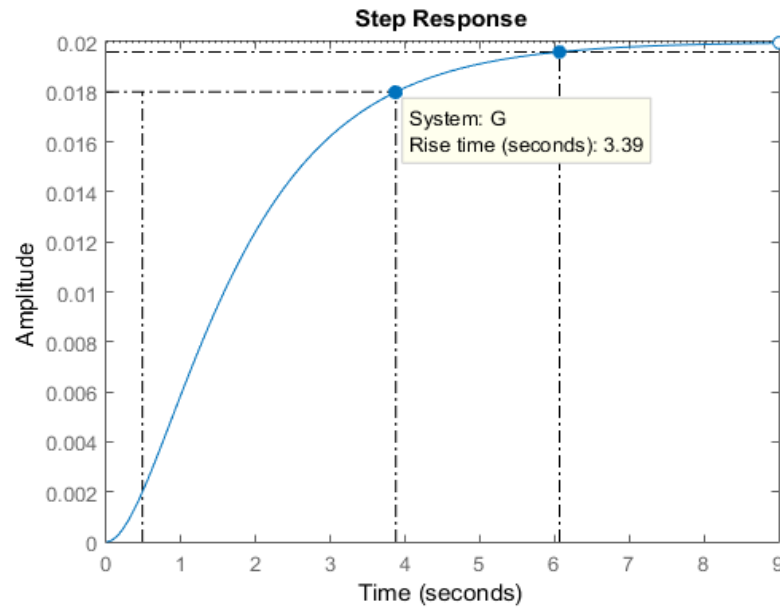


Figura 3. Rising time en malla abierta.

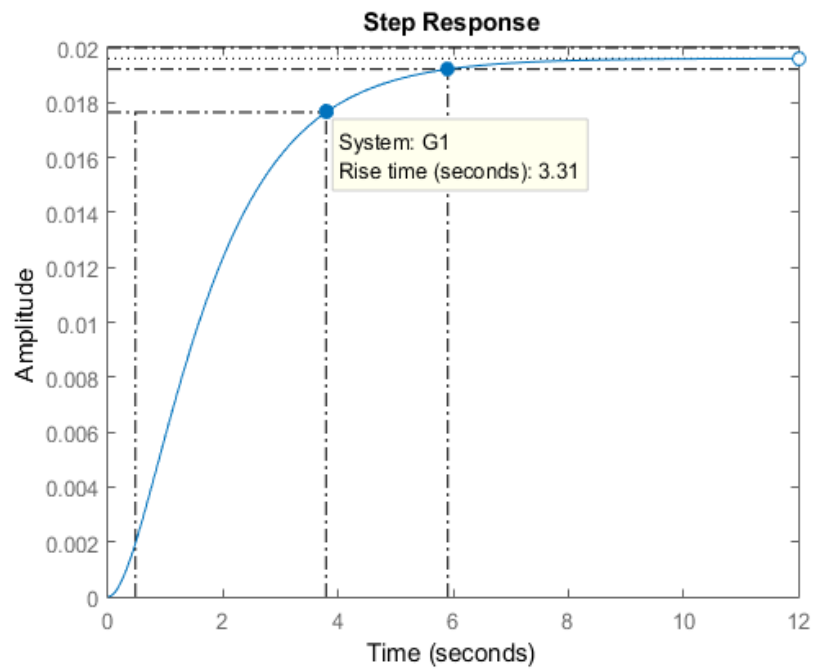


Figura 4. Rising time en malla cerrada.

El rising time en malla abierta es de 3.39s mientras que en malla cerrada es de 3.31s con un error del 2.41%.

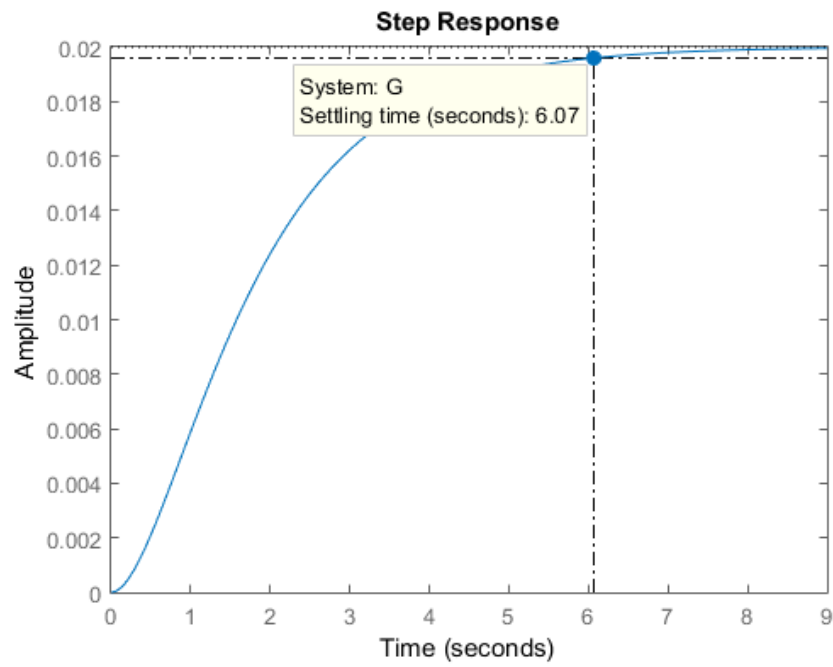


Figura 5. Settling time en malla abierta.

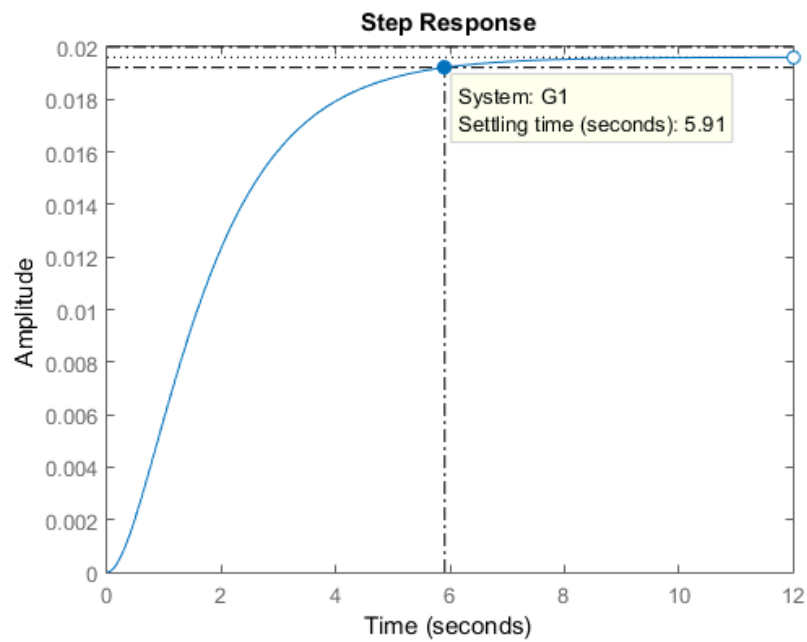


Figura 6. Settling time en malla cerrada.

El settling time en malla abierta es de 6.07s mientras que en malla cerrada es de 5.91s con un error del 2.7%.

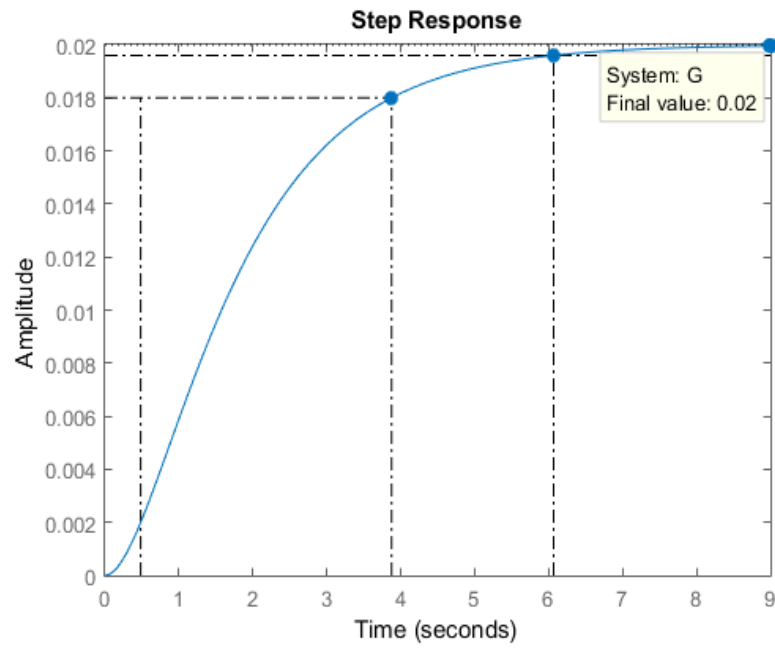


Figura 7. Error en estado estable en malla abierta.

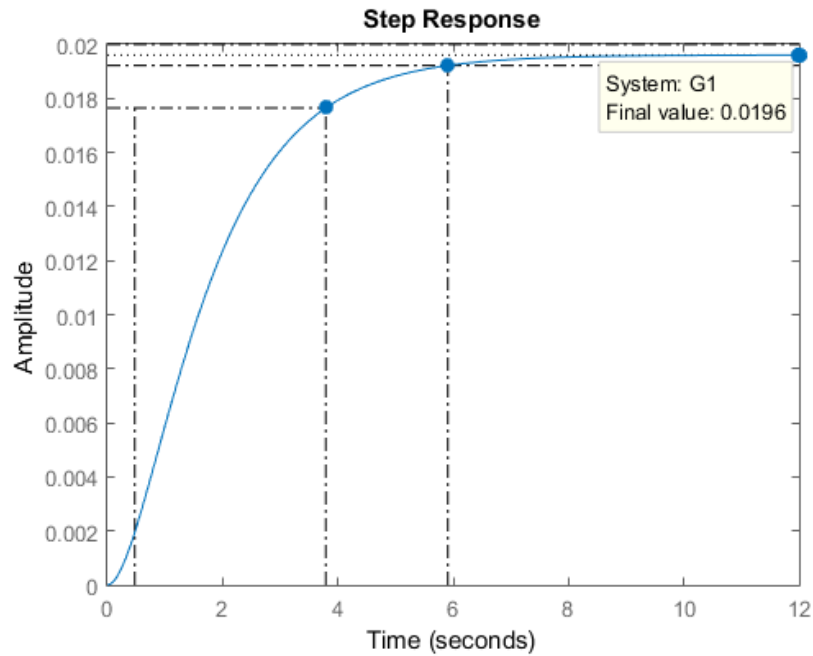


Figura 8. Error en estado estable en malla cerrada.

El error en estado estable en malla abierta es de 0.98 mientras que en malla cerrada es de 0.9804 con un error del 0.04%.

En conclusión, mejora la velocidad de respuesta y los tiempos de asentamiento y subida al realimentarlo unitariamente.

f)

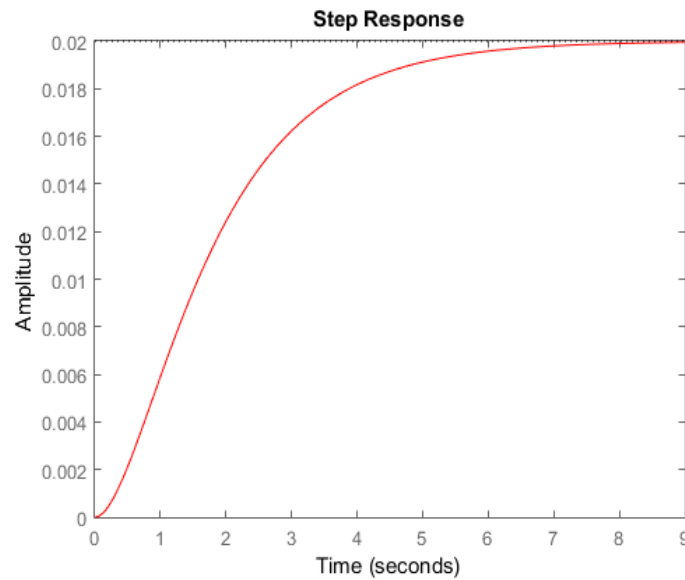


Figura 9. Respuesta del sistema en malla abierta.

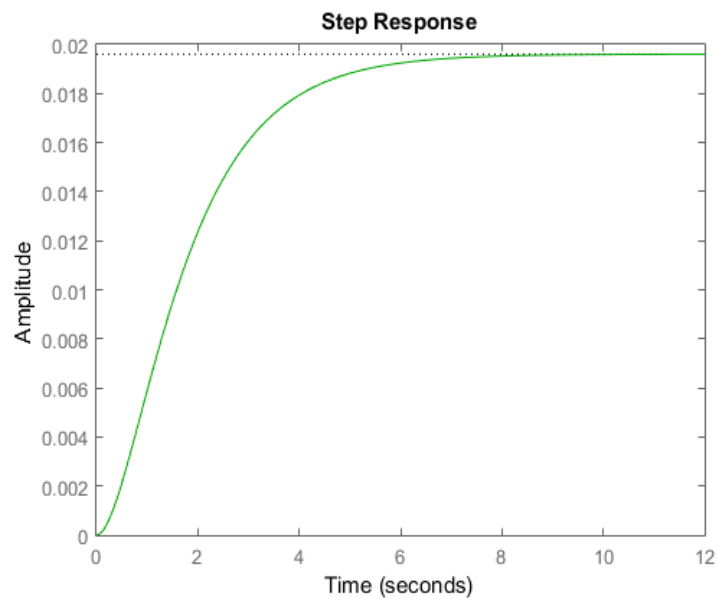


Figura 10. Respuesta del sistema en malla cerrada.

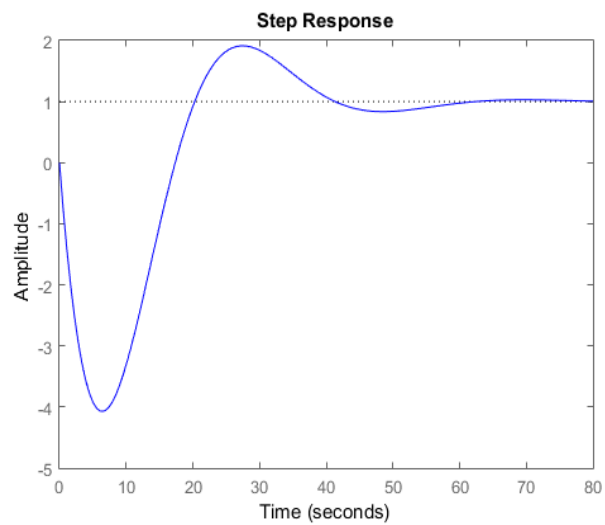


Figura 11. Respuesta del sistema con el controlador PID en malla cerrada.

Finalmente la comparación de las respuestas con respecto a la entrada es la siguiente:

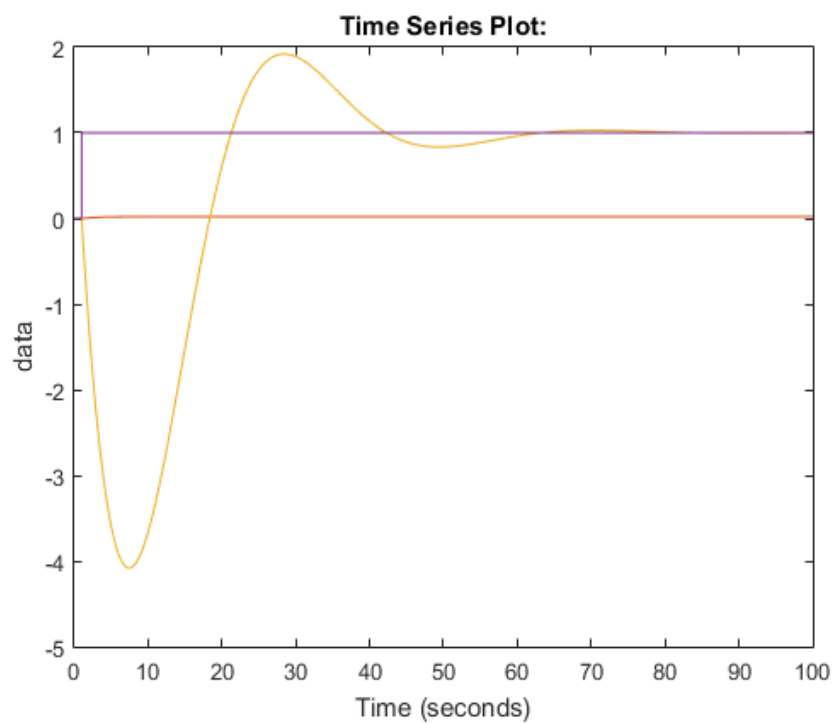


Figura 12. Comparación de las respuestas con respecto a la entrada.

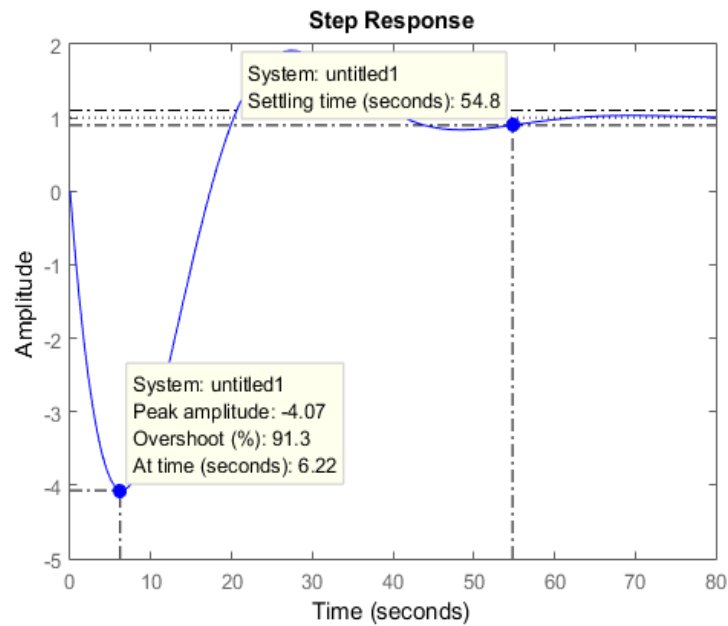


Figura 13. Indicadores de desempeño del PID.

El porcentaje de overshoot debería ser del 18%, pero debido a los cálculos hechos para el diseño del controlador, éste nos indica uno del 91.3% con un error del 80.28%, mientras que el settling time resulto muy cercano a lo que queríamos que era el valor de 49s con un valor de 54.8s con un error del 10.58%.

Modelo en simulink:

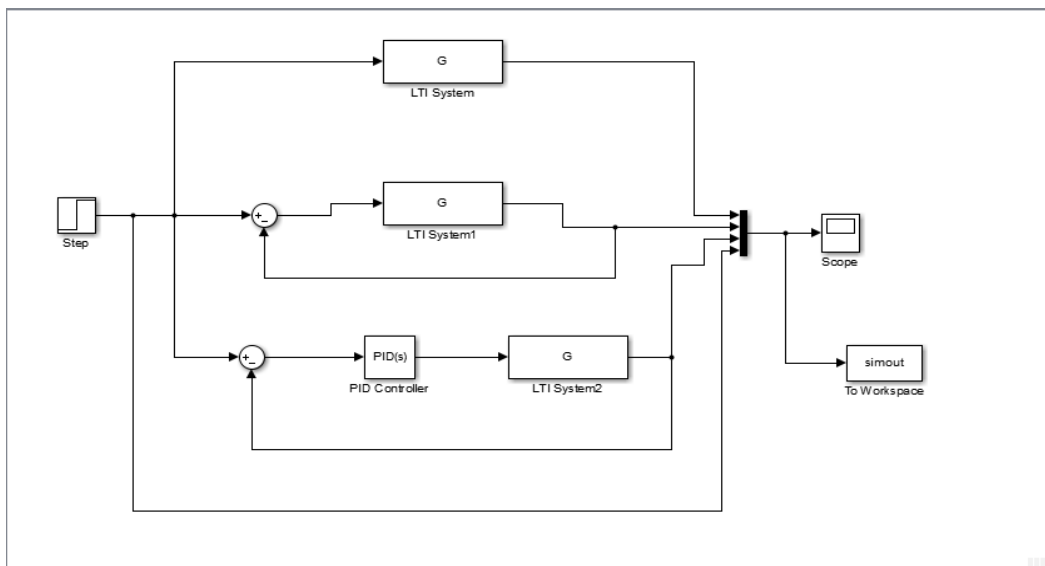
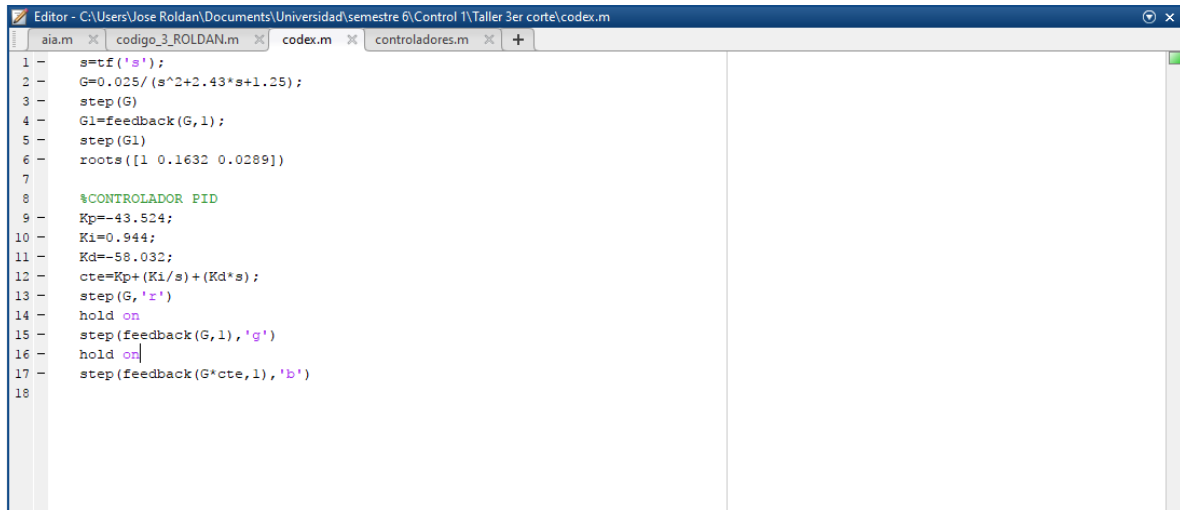


Figura 14. Modelo en simulink.

Código en matlab:



```
Editor - C:\Users\Jose Roldan\Documents\Universidad\semestre 6\Control 1\Taller 3er corte\codex.m
aia.m x codigo_3_ROLDAN.m x codex.m x controladores.m x +
1 - s=tf('s');
2 - G=0.025/(s^2+2.43*s+1.25);
3 - step(G)
4 - G1=feedback(G,1);
5 - step(G1)
6 - roots([1 0.1632 0.0289])
7
8 %CONTROLADOR PID
9 - Kp=-43.524;
10 - Ki=0.944;
11 - Kd=-58.032;
12 - cte=Kp+(Ki/s)+(Kd*s);
13 - step(G,'r')
14 - hold on
15 - step(feedback(G,1),'g')
16 - hold on
17 - step(feedback(G*cte,1),'b')
18
```

Figura 15. Código en matlab.