UNIVERSIDAD SIMÓN BOLÍVAR

COORDINACIÓN DE MAESTRÍA EN TECNOLOGÍA E INGENIERÍA ELECTRÓNICA, MENCIÓN MECATRÓNICA

EC7136 - Electrónica de Potencia II

Profesor: José A. Restrepo

Estudiante: William Chacón

Carnet: 20-91334

TAREA 1:

INTRODUCCIÓN

En el presente Proyecto, se presenta el sistema de control de un circuito de filtro de segundo orden, Sallen Key. El circuito junto con sus componentes se muestra en la figura 1. El objetivo del presente escrito es desarrollar un sistema de control que permita mantener un sistema de control que garantice una salida de 25V. las variables dentro del circuito se encuentran definidas: $R_1 = 10k\Omega$, $R_2 = 10k\Omega$, $R_3 = 30k\Omega$, $R_5 = 51k\Omega$, $C_1 = 470nf$, $C_2 = 470nf$. Se asume que el Op-Amp es ideal, así como el resto de elementos del circuito. La propuesta para solventar dicha necesidad es implementar un control PI en la fuente de alimentación, ya que es el único elemento controlable dentro del circuito presentado.

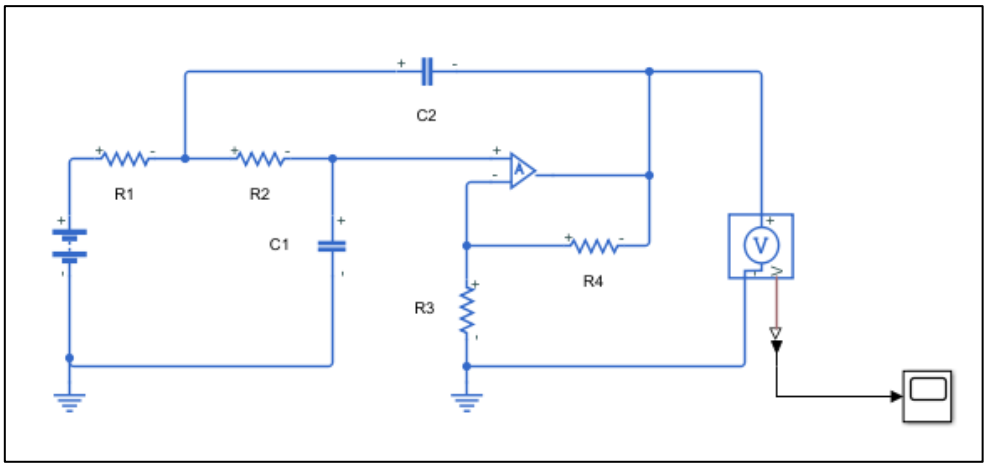

Figura 1: Circuito del Filtro Sallen Key

PLANTEAMIENTO

Para poder implementar un control adecuado, primero es necesario definir el modelo matemático. A continuación, se proceden a definir las expresiones que permitirán encontrar la función de transferencia del sistema.

$$\frac{V_1 - V_{in}}{R_1} = \frac{V_+ - V_1}{R_2} + C_2 \frac{d(V_o - V_1)}{dt} \qquad (1)$$

$$\frac{V_+ - V_1}{R_2} = -C_1 \frac{dV_+}{dt} \qquad (2)$$

$$\frac{V_-}{R_3} = \frac{V_o - V_-}{R_4} \qquad (3)$$

$$V_+ = V_- \qquad (4)$$

Despejando las diferencias de tensión de interés de la ecuación (2) y (3) se tiene

$$V_o = \left(1 + \frac{R_4}{R_3}\right) V_- = \left(1 + \frac{R_4}{R_3}\right) V_+ \qquad (5) \quad \rightarrow \quad V_- = V_+ = \frac{1}{\left(1 + \frac{R_4}{R_3}\right)} V_o \qquad (6)$$

$$V_1 = V_+ + R_2 C_1 \frac{dV_+}{dt} \qquad (7)$$

Sustituyendo (7) en (1) y multiplicando ambos lados de la expresión por el término de la razón de las resistencias 3 y 4 se tiene que:

$$\left(1 + \frac{R_4}{R_3}\right)\left(\frac{\left(V_+ + R_2 C_1 \frac{dV_+}{dt}\right) - V_{in}}{R_1}\right) = \left(1 + \frac{R_4}{R_3}\right)\left(\frac{V_+ - \left(V_+ + R_2 C_1 \frac{dV_+}{dt}\right)}{R_2} + C_2 \frac{d\left(V_o - \left(V_+ + R_2 C_1 \frac{dV_+}{dt}\right)\right)}{dt}\right) \qquad (8)$$

Sustituyendo la expresión (5) y (4) en los términos correspondientes de (8) se tiene

$$\frac{\left(V_o + R_2 C_1 \frac{dV_o}{dt}\right) - \left(1 + \frac{R_4}{R_3}\right) V_{in}}{R_1} = \frac{V_o - \left(V_o + R_2 C_1 \frac{dV_o}{dt}\right)}{R_2} + C_2 \frac{d\left(\left(1 + \frac{R_4}{R_3}\right) V_o - \left(V_o + R_2 C_1 \frac{dV_o}{dt}\right)\right)}{dt}$$

Reordenando los términos

$$R_2\left(\left(V_o + R_2 C_1 \frac{dV_o}{dt}\right) - \left(1 + \frac{R_4}{R_3}\right) V_{in}\right) = R_1\left(V_o - \left(V_o + R_2 C_1 \frac{dV_o}{dt}\right)\right) + R_1 R_2 C_2 \frac{d\left(\left(1 + \frac{R_4}{R_3}\right) V_o - \left(V_o + R_2 C_1 \frac{dV_o}{dt}\right)\right)}{dt}$$

$$R_2 V_o + R_2 R_2 C_1 \frac{dV_o}{dt} - R_2\left(1 + \frac{R_4}{R_3}\right) V_{in} = (R_1 V_o - R_1 V_o) - R_1 R_2 C_1 \frac{dV_o}{dt} + R_1 R_2 C_2 \frac{d\left(\left(1 + \frac{R_4}{R_3}\right) V_o\right)}{dt} - R_1 R_2 C_2 \frac{dV_o}{dt} - R_1 R_2 C_2 \frac{d\left(R_2 C_1 \frac{dV_o}{dt}\right)}{dt}$$

$$-R_2\left(1+\frac{R_4}{R_3}\right)V_{in}$$
$$= -R_2V_o - R_2R_2C_1\frac{dV_o}{dt} - R_1R_2C_1\frac{dV_o}{dt} + \left(1+\frac{R_4}{R_3}\right)R_1R_2C_2\frac{dV_o}{dt} - R_1R_2C_2\frac{dV_o}{dt}$$
$$- R_1R_2C_2R_2C_1\frac{d}{dt}\left(\frac{dV_o}{dt}\right)$$

$$-\left(1+\frac{R_4}{R_3}\right)V_{in} = -V_o + \left(-R_2C_1 - R_1C_1 - R_1C_2 + \left(1+\frac{R_4}{R_3}\right)R_1C_2\right)\frac{dV_o}{dt} - R_1C_2R_2C_1\frac{d}{dt}\left(\frac{dV_o}{dt}\right)$$

$$\frac{d}{dt}\left(\frac{dV_o}{dt}\right) = -\frac{\left((R_2+R_1)C_1 - \left(1-\left(1+\frac{R_4}{R_3}\right)\right)R_1C_2\right)}{R_1R_2C_2C_1}\frac{dV_o}{dt} + \frac{1}{R_1R_2C_2C_1}\left(-V_o + \left(1+\frac{R_4}{R_3}\right)V_{in}\right) \quad (9)$$

Transformando la expresión (9) en un conjunto de variables de estado (V_o), se tiene que:

$$\begin{bmatrix}\frac{\partial}{\partial t}v_o\\ \frac{\partial^2}{\partial t^2}v_o\end{bmatrix} = \begin{bmatrix}0 & 1\\ -\frac{1}{R_1R_2C_2C_1} & -\frac{\left((R_2+R_1)C_1 - \left(1-\left(1+\frac{R_4}{R_3}\right)\right)R_1C_2\right)}{R_1R_2C_2C_1}\end{bmatrix}\begin{bmatrix}v_o\\ \frac{\partial}{\partial t}v_o\end{bmatrix} + \begin{bmatrix}0\\ \frac{\left(1+\frac{R_4}{R_3}\right)}{R_1R_2C_2C_1}\end{bmatrix}V_i$$

Ahora, si se reduce la expresión a partir de la creación de nuevas variables auxiliares

$$K_2 = -\frac{1}{R_1R_2C_2C_1}$$

$$G = \left(1+\frac{R_4}{R_3}\right)$$

$$K_1 = \frac{-1}{R_1R_2C_2C_1}\left((R_2+R_1)C_1 - \left(1-\left(1+\frac{R_4}{R_3}\right)\right)R_1C_2\right) = \left((R_2+R_1)C_1 - G\,R_1C_2\right)K_2$$

$$x = \frac{dV_o}{dt}$$

El mismo sistema puede expresarse como:

$$\begin{bmatrix}\frac{\partial}{\partial t}v_o\\ \frac{\partial}{\partial t}x\end{bmatrix} = \begin{bmatrix}0 & 1\\ K_2 & K_1\end{bmatrix}\begin{bmatrix}v_o\\ x\end{bmatrix} + \begin{bmatrix}0\\ -G\,K_2\end{bmatrix}V_i \quad (10)$$

Y su correspondiente función de transferencia se presenta a continuación:

$$\frac{v_{o(s)}}{V_{i(s)}} = \frac{\left(\left(1+\frac{R_b}{R_a}\right)\frac{1}{C_1C_2R_1R_2}\right)}{s^2 + s\left(\frac{1}{C_1R_2} + \frac{1}{C_1R_1} - \left(1-\left(1+\frac{R_4}{R_3}\right)\right)R_1C_2\right) + \frac{1}{C_1C_2R_1R_2}} = \frac{-GK_2}{s^2 - sK_1 - K_2} \quad (11)$$

Partiendo de la expresión (10), se procede a diseñar un controlador PI.

Para este caso, se desarrolló un código en C con la finalidad de cumplir esta tarea. En esta ocasión, se decidió implementar un control PI para mantener la salida deseada del sistema.

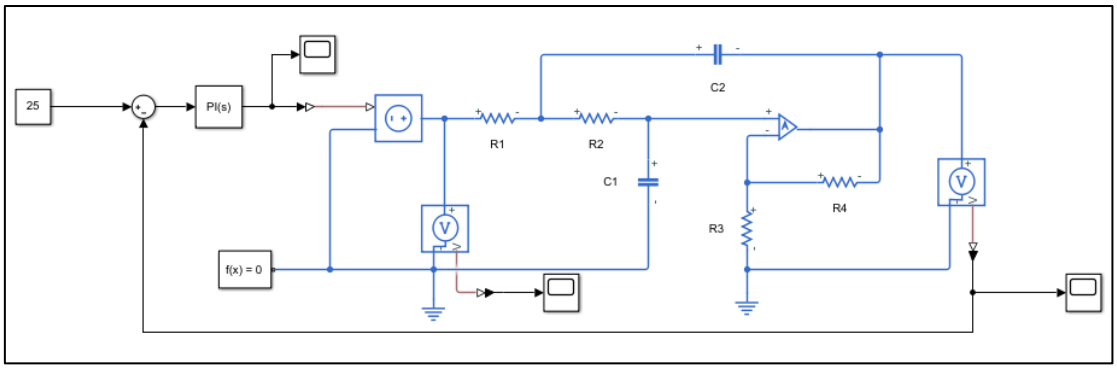

Figura 2: Diagrama del Sistema con su correspondiente sistema de control

Para la resolución. Se implementaron diversos algoritmos de solución de ecuaciones diferenciales, con la finalidad de tener diferentes soluciones que permita validar el comportamiento de la respuesta, así como visualizar los ligeros cambios que poseen los métodos más sencillos y menos precisos respecto a los más complejos y precisos. Estos métodos son:

- Método de Euler (EU)
- Método de Runge-Kutta de Segundo Orden (RK2)
- Método de Runge-Kutta de cuarto Orden (RK4)
- Método de Runge-Kutta de quito Orden (RK5)
- Sarafyan's Runge-Kutta (SRK4)

Hubo problemas con la implementación de los métodos Sarafyan's Runge-Kutta de orden 6, y Runge-Kutta-Fehlberg de cuarto Orden y Runge-Kutta-Fehlberg de quinto Orden debido a las divisiones de algunos de su coeficientes, que sobrepasan el límite de los números flotantes.

CÓDIGO

El código, enviado en el proyecto "*Tarea_1*", con un nombre del archivo principal "*main.c*". En el mismo se realizaron las declaraciones pertinentes para las variables y funciones a implementar. Dentro de la función principal, la variable "*NMtdStr*" se encarga de seleccionar el método de solución que será almacenado en el puntero "**NumMethod[]*", de acuerdo al valor de la misma. A continuación, se especifica el valor correspondiente para cada uno de los métodos ofrecidos con el respectivo nombre de cada una de estas funciones.

- 0: ODE_EU (EU)
- 1: ODE_RK2 (RK2)
- 2: ODE_RK4 (RK4)
- 3: ODE_RK5 (RK5)
- 4: ODE_SRK4 (SRK4)
- 5: ODE_RKF5 (RKF5)

Posteriormente a ello, dentro de la función principal, se salta a la función *"Solve"*, que contempla el código encargado de ejecutar el proceso de solución del problema.

La idea de escribir el código de esta forma, es que un usuario que desee modificarlo para implementarlo solo tenga que realizar modificaciones en las primeras partes del código (y, de requerirlo, agregar métodos de solución al final), sin necesidad de tener que modificar el resto del código, cambiando nombres o valores, para que el mismo pueda ejecutarse correctamente.

El código en Matlab consta de un conjunto de un conjunto de funciones para hacer gráficas y calcular los errores RMS para su correspondiente comparación. Se recomienda correr la simulación en simulink y todos los métodos de solución del código de C para que se pueda efectuar de forma apropiada el archivo .m sin necesidad de modificarlo.

Por conveniencia, se recomienda mantener todos los códigos en la misma carpeta.

RESULTADOS

Para esta experiencia, se procedió realizar un afinamiento manual del control, basado en las diferentes salidas obtenidas. Para este proceso solo se implementó el algoritmo de resolución de Euler como método de solución de la ecuación diferencial. Al ser el más impreciso de los métodos, las constantes del controlador encontradas para garantizar el correcto funcionamiento del controlador tenderán a ser ligeramente más robustas, por lo que, en teoría, con los métodos más precisos se deberían obtener resultados mejores. De este proceso, los resultados obtenidos para el controlador PI fueron:

$$K_p = 0.02; \quad K_I = 15$$

A continuación, se mostrarán las figuras obtenidas de graficar los resultados arrojados por el código en C para cada uno de los métodos, manteniendo los mismos valores para las constantes del controlador. En la Figura 3 se muestra la salida para el método de Euler; en la Figura 4 la salida del método de Runge-Kutta de orden 2; en la Figura 5 la salida del método de Runge-Kutta de orden 4; en la Figura 6 la salida del método de Runge-Kutta de orden 6; y en la Figura 7 la salida del método de Sarafyan's Runge-Kutta de orden 4;

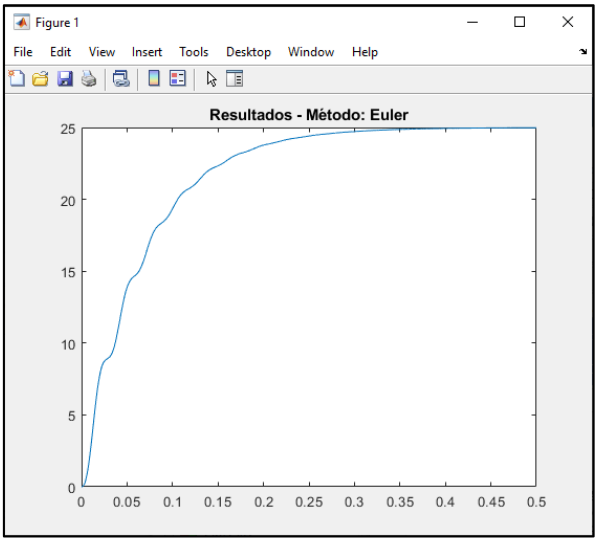

Figura 3: resultados de la salida del filtro para el método ODE_EU

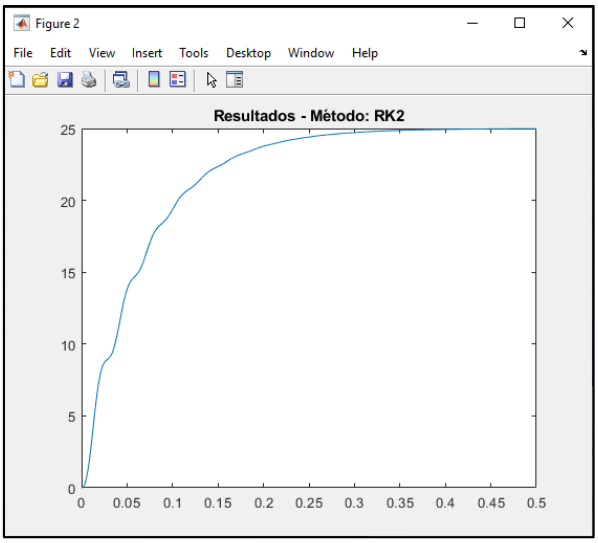

Figura 4: resultados de la salida del filtro para el método ODE_RK2

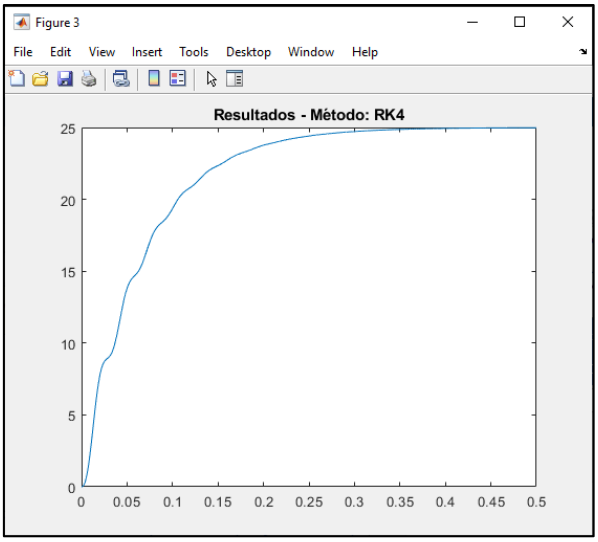

Figura 5: resultados de la salida del filtro para el método ODE_RK4

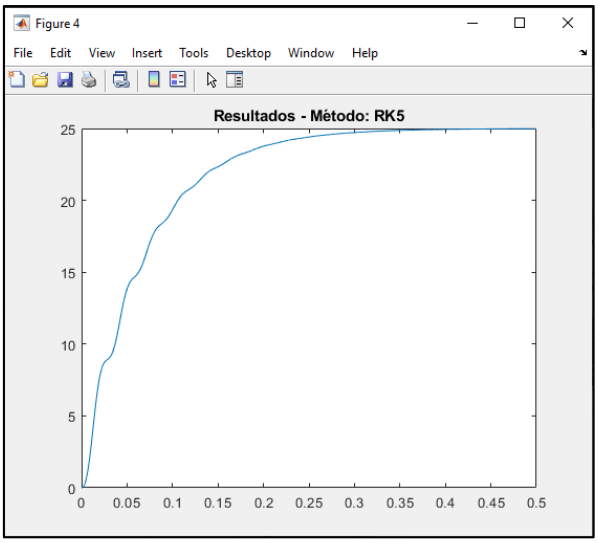

Figura 6: resultados de la salida del filtro para el método ODE_RK5

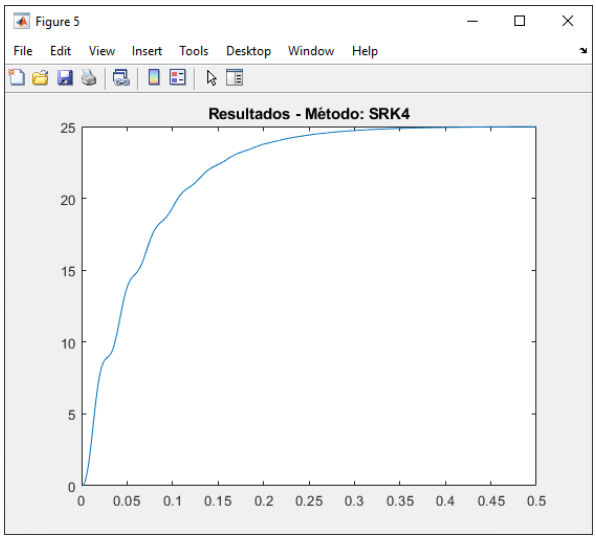

Figura 7: resultados de la salida del filtro para el método ODE_SRK4

A continuación, en la Figura 8, se muestran los datos de la simulación realizada en Simulink/Matlab, partiendo de la simulación con los parámetros originales del mismo y aplicando un controlador con las mismas variables.

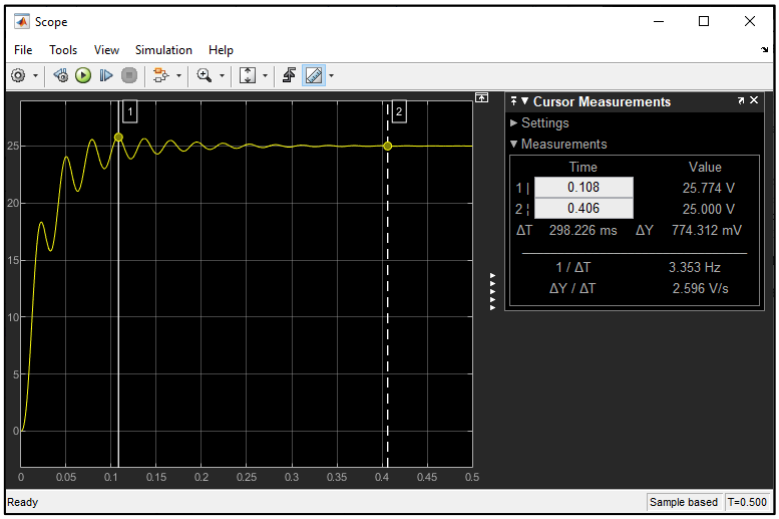

Figura 8: resultados de la salida del filtro simulado en Simulink/Matlab. Valores del controlador: $K_p = 0.02;\ K_I = 15$.

Como se puede apreciar en la figura, existe una diferencia claramente apreciable. Para el caso del modelo desarrollado en Simulink/Matlab, el control con esos valores se tradice en una fuerte subida oscilatoria (amortiguada), mientras que para los valores obtenidos por los modelos codificados en C presentan una subida más pausada, que se estabiliza poco tiempo después (menos de 0.15 segundos si se consideran los periodos oscilatorios posteriores), pero posee una subida (tiempo de respuesta) que, si bien también presenta algunas oscilaciones, es mucho menos amortiguada, un poco más lenta (un retardo ligeramente mayor) y no presenta sobre pico. Es decir, la respuesta obtenida por el código en C es más conservadora, pudiendo evitar daños por sobretensión a equipos que no sean capaces de soportarlos.

Por otra parte, en vista de la inconsistente respuesta de la simulación hecha en ambos sistemas, realizando pruebas en ambos, se encontró una configuración de la simulación en Simulink/Matlab que ofrece una respuesta muy parecida a la obtenida (ver la Figura 9).

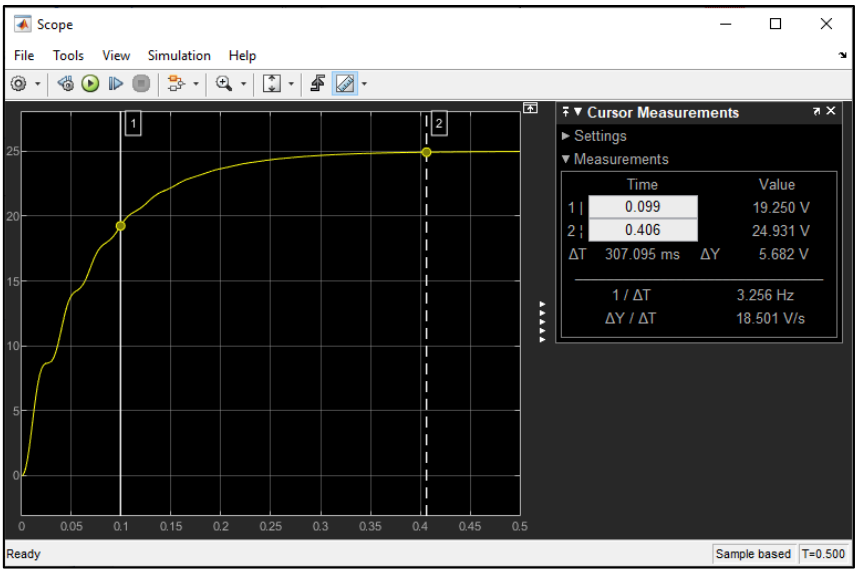

Figura 9: resultados de la salida del filtro simulado en Simulink/Matlab. Valores del controlador: $K_p = 0.02;\ K_I = 15/2.7$.

De dichas pruebas, la única relación que se pude encontrar entre ambas respuestas es la existencia de un factor cercano a 3 de relación entre las proporciones del valor del coeficiente del integrado entre ambos modelos implementados. Hace falta realizar una investigación más profunda sobre la diferencia en el circuito de Matlab que ocasiona dicha discrepancia en el integrador.

Una vez, se procede a determinar los errores cuadráticos medios de los métodos respecto a los resultados arrojados por Simulink/Matlab. En la siguiente tabla se pueden apreciar los resultados de este cálculo. Como era de esperar, los métodos con mayor orden de precisión son aquellos que presentan los mejores resultados. No obstante, cabe destacar que las diferencias entre los resultados no se comienzan a apreciar sino hasta el quinto decimal en una magnitud del orden de las decenas. Esto podría indicar que, si bien, los métodos RK son considerablemente más precisos, para el presente problema pueden no generar algún aporte significativo en el mismo, por lo que el método de Euler sería suficiente para su resolución. Sin embargo, es igualmente importante destacar que el método más preciso es el RK5

Método de Solución	Error cuadrático medio.
EU	0.002511045590645
RK2	0.002577493689175
RK4	0.002578423321844
RK5	0.002544657997127
SRK4	0.002578421244963

Tabla 1: error cuadrático medio de cada método respecto a los resultados de Simulink

Por último, se muestra una figura donde se presentan todos los métodos utilizados con la finalidad de mostrar una comparación visual de los comportamientos de cada uno de ellos (ver Figura 10). De aquí también se puede concluir que una mayor precisión no representa una

ganancia significativa, en términos de precisión de información, para el problema planteado en el presente trabajo.

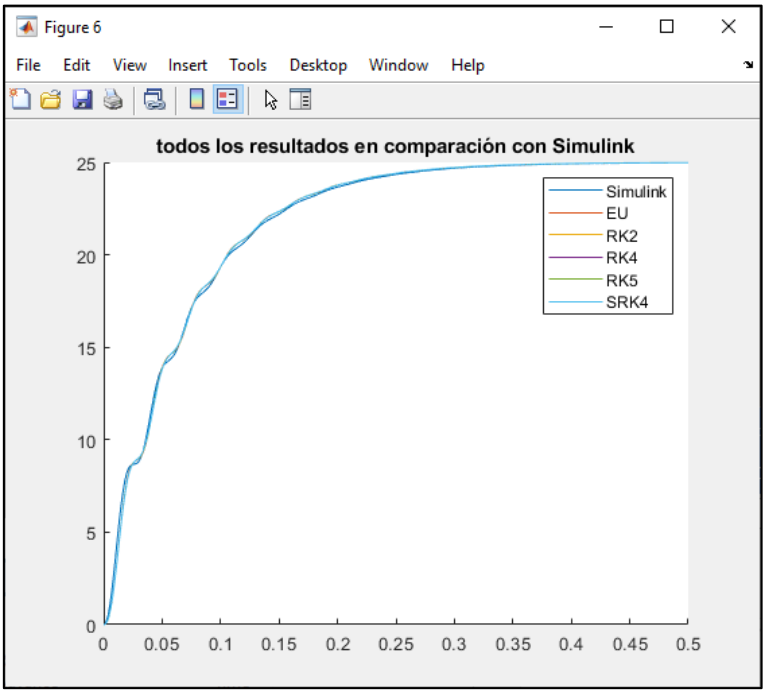

Figura 10: gráfico comparativo de todos los resultados por los métodos numéricos programados en C y los resultados obtenidos en la simulación.

CONCLUSIONES

De los siguientes resultados se pudo implementar un controlador PI capaz de garantizar una salida aproximadamente de 25.0V sin compromisos de sobre picos que pudiese comprometer el funcionamiento de los equipos. Así mismo, todos los métodos implementados para su solución resultaron ser lo suficientemente precisos como para arrojar valores satisfactorios. No obstante, se pudo apreciar que, en vista de ser un problema lineal sencillo, puede no ser necesario aplicar métodos tan complejos para su resolución, por lo que el algoritmo de Euler sería suficiente.

Otro punto interesante a resaltar es el hecho de la discrepancia inicial entre la simulación de Simulink/Matlab y las realizadas en lenguaje C, donde, después de muchas pruebas, se logró identificar que, por alguna razón, existe un factor de proporcionalidad entre los valores del integrador de Simulink y los usados en "*main.c*", de aproximadamente 2.7. la razón de dicha discrepancia y por qué solo afecta al integrador no pudo ser descifrada para la entrega del presente informe, pero permite demostrar la necesidad de siempre corroborar los resultados obtenidos por softwares avanzados mediante cálculos más simples y aproximados, pero igualmente válidos.

Para concluir con el proyecto, se pudo diseñar satisfactoriamente un control PI que permite una respuesta rápida, del orden de los 0.3s, sin presentar oscilaciones ni sobre picos significativos, lo que garantiza la vida útil y el correcto funcionamiento de los elementos del circuito estudiado.