Proyecto final
Diseño de un sistema de control realimentado para un circuito RC de segundo orden sobreamortiguado

Daniel Alberto Sáenz Obando
Escuela de Ingeniería Eléctrica
Universidad de Costa Rica
San José, Costa Rica
daniel.saenzobando@ucr.ac.cr

Marlon Jafeth Gutiérrez Vásquez.
Escuela de Ingeniería Eléctrica
Universidad de Costa Rica
San José, Costa Rica
marlon.gutierrezvasquez@ucr.ac.cr

Brandon Daniel Jiménez Campos
Escuela de Ingeniería Eléctrica
Universidad de Costa Rica
San José, Costa Rica
brandon.jimenezcampos@ucr.ac.cr

Rodrigo Evelio Sánchez Araya
Escuela de Ingeniería Eléctrica
Universidad de Costa Rica
San José, Costa Rica
rodrigo.sanchezaraya@ucr.ac.cr

***Resumen*—En este proyecto se desarrolló un sistema de control realimentado mediante el uso del microcontrolador Arduino UNO R3. Se utilizaron las herramientas de software `MATLAB`, `Simulink` y `Sisotool` para realizar el diseño del modelo del controlador, a partir de la técnica LGR, así como para la elaboración de simulaciones en tiempo continuo y discreto para validar su funcionamiento. El modelo del controlador discretizado fue implementado en el Arduino y se demostró el funcionamiento del diseño de una forma experimental para una señal de referencia manual con un potenciómetro y una señal cuadrada periódica.**

***Index Terms*—MATLAB, Simulink, Sisotool, Arduino, control realimentado, lazo de control, LGR.**

I. Introducción

En el presente reporte se presenta el diseño y la implementación de un sistema de control realimentado para un circuito RC de segundo orden sobreamortiguado. El objetivo principal de este sistema de control es cumplir las especificaciones brindadas para una señal de referencia de un escalón unitario, las cuales son error estacionario nulo, tiempo de asentamiento al $2\,\%$ menor a $5\,\mathrm{s}$ y un sobrepaso máximo inferior al $5\,\%$. Se realizaron simulaciones en `MATLAB`, `Sisotool` y `Simulink` para verificar la validez del modelo del controlador implementado, junto con cálculos manuales para la obtención del modelo de la planta. Finalmente, se implementó el diseño en un Arduino UNO R3, donde se probó experimentalmente el diseño.

II. Caracterización del sistema

En primer lugar, es necesario describir el sistema lineal invariante con el tiempo para el cual se diseñó el sistema de control realimentado. La planta consiste en un circuito RC de segundo orden sobreamortiguado, tipo 0, compuesto por dos capacitores y dos resistores. La planta junto con los valores medidos experimentalmente se muestran en la Figura 1. Se requería controlar la tensión eléctrica de salida v_o, a partir de la tensión eléctrica de entrada v_s proporcionada por un Arduino UNO R3.

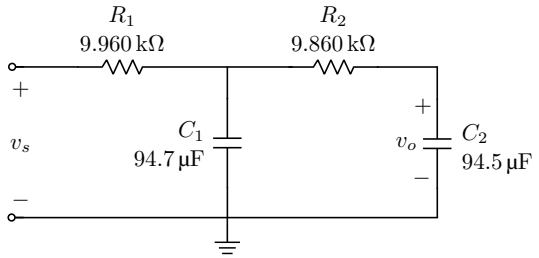

Figura 1. Circuito eléctrico con los valores medidos en el laboratorio del proceso a controlar.

En términos generales, se realizaron dos tipos de análisis: de lazo abierto y de lazo cerrado. En la Figura 2 se muestra el diagrama de bloques de lazo abierto, donde se señalan el controlador C, la planta P y la señal de entrada u y salida y. En este caso, el controlador corresponde al Arduino UNO R3 con un microcontrolador ATmega328P y la planta consiste en el circuito eléctrico mencionado anteriormente.

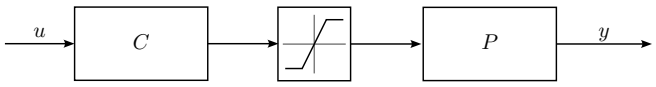

Figura 2. Diagrama de bloques del sistema en lazo abierto.

Ahora bien, para el sistema de control realimentado diseñado, se debe considerar el sistema con el lazo de control cerrado, el cual se muestra en la Figura 3.

En este, se muestran las señales de interés en el proceso de diseño y operación: la referencia r (valor objetivo de tensión

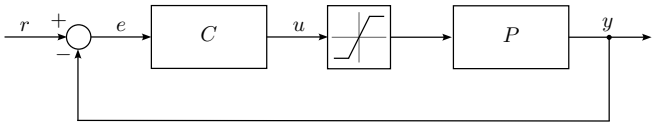

Figura 3. Diagrama de bloques del sistema en lazo cerrado.

eléctrica v_o), la señal de control u (salida del Arduino del pin digital 6 PWM, que corresponde a la tensión de entrada de la planta) y la salida y (tensión eléctrica v_o). Asimismo, se muestran los componentes de interés correspondientes al controlador y la planta.

Además, se considera que el sistema posee dos interfaces: el convertidor A/D en los pines analógicos, que consiste en el sensor de tensión eléctrica del Arduino; y un convertidor D/A, el cual se desempeña como el actuador del sistema que envía la señal de control de tensión eléctrica a la planta, a través de los pines digitales del Arduino.

III. Obtención del modelo de la planta

Ahora bien, con base en la caracterización realizada en la sección anterior, se procede con la obtención del modelo del circuito eléctrico a controlar, según la Figura 1.

Se tienen dos componentes almacenadores de energía (capacitores), lo cual coincide con la teoría de un modelo de segundo orden. En el Apéndice A, se documentó el procedimiento realizado para obtener la ecuación diferencial de la salida v_o en términos de la entrada v_s, así como la función de transferencia del modelo de la planta.

$$R_1R_2C_1C_2\ddot{v_o} + (R_1C_1 + R_1C_2 + R_2C_2)\dot{v_o} + v_o = v_s \quad (1)$$

Luego, se aplica la transformada de Laplace y se resuelve para $\frac{V_o}{V_s}$. Se toman condiciones iniciales nulas para el análisis y se obtiene la función de transferencia que modela la planta en cuestión.

$$H(s) = \frac{1}{R_1R_2C_1C_2s^2 + (R_1C_1 + R_1C_2 + R_2C_2)s + 1} \quad (2)$$

Al reemplazar los valores experimentales de los componentes eléctricos de la Figura 1, resulta la función de transferencia que modela la planta:

$$H(s) = \frac{1}{0.8789s^2 + 2.8162s + 1} \quad (3)$$

Al analizar la forma de (3), se determina que la función de transferencia que modela la planta efectivamente es de segundo orden. Además de que es tipo 0, puesto que no contiene polos en el origen. También, esta representa una respuesta sobreamortiguada, debido a que el valor numérico del factor de amortiguamiento es mayor a 1, específicamente $\xi = 1.5020$.

Para el proceso de diseño del controlador, se requiere la ubicación de los polos del modelo de la planta, los cuales se indican en (4).

$$\begin{aligned} s_1 &= -0.4067 \\ s_2 &= -2.7975 \end{aligned} \quad (4)$$

A modo de verificación gráfica, se simuló la respuesta temporal del sistema de lazo abierto a un escalón unitario en la Figura 4, a partir de (3). En esta, se evidencia el comportamiento sobreamortiguado de la planta, debido a la ausencia de un sobrepaso respecto a la entrada.

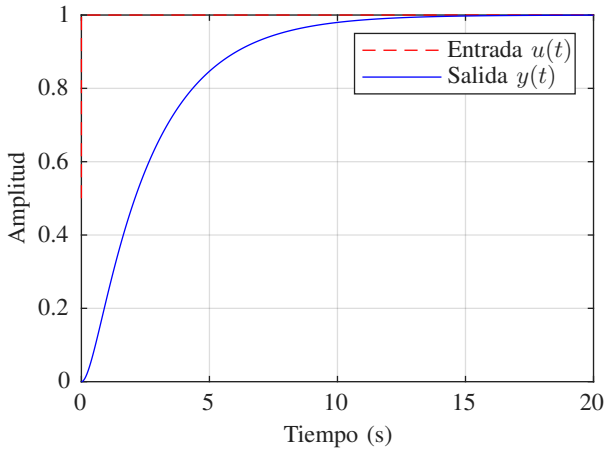

Figura 4. Simulación de la respuesta temporal del modelo de la planta en lazo abierto.

Asimismo, en la Figura 5, se observa la respuesta en frecuencia del modelo de la planta, donde destaca la pendiente de -40 dB/década característica de un sistema de segundo grado.

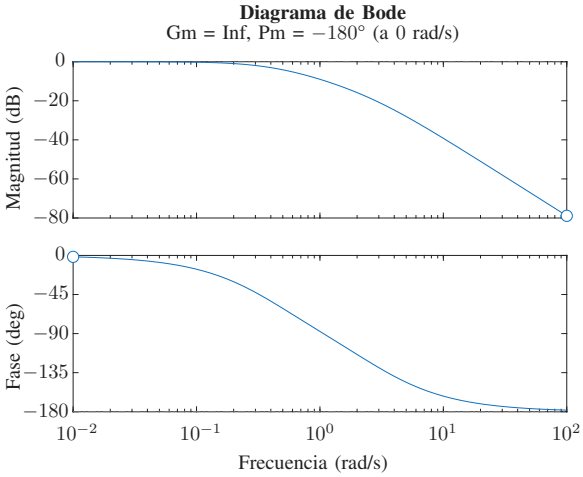

Figura 5. Diagrama de Bode de lazo abierto del sistema.

IV. Diseño del controlador

Para el diseño del controlador C del sistema de control realimentado diseñado, se debieron analizar las especificaciones

dadas en el enunciado respecto a cambios tipo escalón unitario en la referencia:

- Error estacionario igual a cero.
- Tiempo de asentamiento al $2\,\%$ menor o igual a $5\,\mathrm{s}$.
- Sobrepaso máximo menor al $5\,\%$.
- Margen de fase mayor a $30°$ y de ganancia superior a $6\,\mathrm{dB}$.

Con base en dichas consideraciones, se diseñó el controlador utilizando la técnica del *Lugar Geométrico de las Raíces* (LGR). Inicialmente, por la naturaleza del sistema, tipo 0, se requiere colocar un integrador en la función de transferencia del controlador para conseguir un error estacionario nulo. Sin embargo, bajo esta configuración, se tiene un tiempo de asentamiento al $2\,\%$ de $27.9\,\mathrm{s}$.

Luego, se aplicó la teoría de cancelación de polos del modelo de la planta, por medio de la adición de ceros en el controlador, tal que sus ubicaciones coincidan. Esto se realiza de forma que se cancela el polo dominante, según (4), $s = -0.4067$. Así, al agregar un cero en la ubicación anterior, se elimina efectivamente la constante de tiempo asociada a este polo y el tiempo de respuesta del sistema disminuye. [1]

En la Figura 6, se muestra la ubicación de los polos seleccionados para el controlador. Se colocaron requerimientos de diseño para ajustar el diseño del controlador a las especificaciones dadas al inicio.

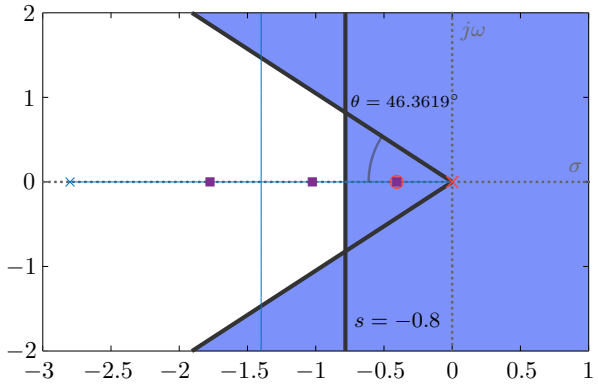

Figura 6. Diseño del controlador utilizando la técnica de LGR.

La línea vertical se coloca de tal forma que se cumpla la especificación de tiempo de asentamiento al $2\,\%$ menor que $5\,\mathrm{s}$. Se debe cumplir que $s < -0.8$. Esto se obtiene a partir de la definición de este requerimiento.

Asimismo, para cumplir con la especificación del sobrepaso máximo, se tiene que esta variable depende de ξ. A partir de la definición, se obtiene que se tiene un ángulo respecto al eje horizontal límite de $\theta = 46.3619°$ para cumplir con esta especificación. En el Apéndice B, se muestran los cálculos detallados de estos valores numéricos según la teoría vista en el curso.

Ahora bien, para cumplir con ambos parámetros, se requiere que la ubicación de los polos esté en la región blanca, lo cual se cumple en el caso del diseño.

Con base en el análisis descrito anteriormente, se determinó el controlador resultante.

$$C(s) = \frac{1.5996\,(s + 0.4060)}{s} \tag{5}$$

V. Simulaciones en tiempo continuo y discreto

Para la verificación de los resultados correspondientes al diseño del controlador, se realizaron las simulaciones en tiempo continuo y discreto mediante el uso de las herramientas `Sisotool` y `Simulink` de MATLAB.

V-A. Tiempo continuo

Primero, en la Figura 7, se muestra el resultado de la simulación del controlador continuo exportado de `Sisotool` con la planta en cuestión y el lazo de realimentación. Esta simulación está asociada al diagrama de bloques de la Figura 3.

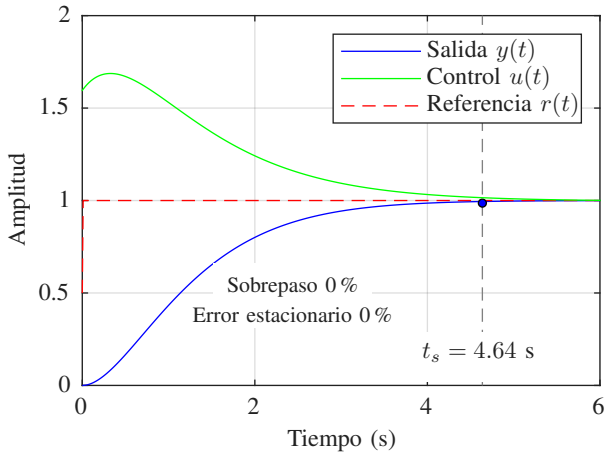

Figura 7. Simulación de la respuesta temporal en lazo cerrado con el modelo del controlador y de la planta en tiempo continuo.

Según el diseño realizado, se tiene una respuesta sobreamortiguada, lo cual indica que no existe sobrepaso respecto a la señal de referencia del escalón unitario. Asimismo, se obtuvo un tiempo de asentamiento al $2\,\%$ de $4.64\,\mathrm{s}$, lo cual cumple con la especificación proporcionada. Como se agregó un integrador al controlador, se determinó que efectivamente, el error estacionario es de $0\,\%$.

Además, la señal de control u en respuesta al escalón unitario posee una amplitud máxima menor a $2\,\mathrm{V}$ ($1.6906\,\mathrm{V}$ específicamente). Este corresponde a un valor aceptable respecto a lo que puede proporcionar el controlador utilizado, entre 0 y $5\,\mathrm{V}$. También, se observa que conforme la salida se aproxima a la referencia, la magnitud de la señal de control disminuye, lo cual es esperable según la teoría y tiende a 1.

Ahora bien, aparte del análisis en el dominio temporal, se requiere estudiar el comportamiento en el dominio de la frecuencia, para el cual interesan dos cantidades: el margen de ganancia y de fase.

Según la Figura 8, se determinó en MATLAB un margen de ganancia infinito (no se da el cruce en la fase de $-180°$). Esto implica que es posible aumentar la ganancia indefinidamente sin llegar a una inestabilidad según este criterio.

Respecto al margen de fase, se reportó un valor de 77.2°. Típicamente, se busca que el margen de fase se encuentre entre 30° y 60°; sin embargo, este resultado numérico implica una respuesta estable, pero más lenta respecto a un margen de fase menor, lo cual no es un inconveniente respecto a las especificaciones conseguidas con la respuesta temporal.

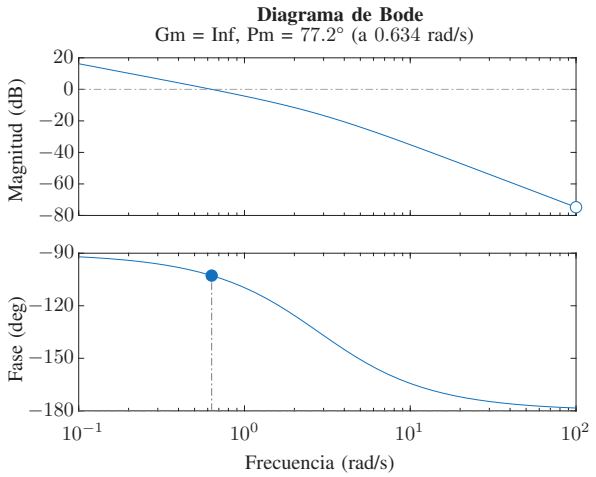

Figura 8. Diagrama de Bode del sistema en lazo cerrado.

V-B. Tiempo discreto

Para la implementación del sistema de control realimentado en Arduino, es necesario convertir el controlador diseñado a una forma discreta, debido a la naturaleza digital de los microcontroladores. Para ello, se discretizó el controlador continuo diseñado de (5) con el comando `cd2` de MATLAB, específicamente con el método `zoh` correspondiente a *zero-order hold*.

Respecto al tiempo de muestreo seleccionado, se observa que el pin digital 6 PWM permite muestrear a una frecuencia de 980 Hz, lo cual equivale a un tiempo de 1.0204 ms. Asimismo, como se deben medir los valores de tensión eléctrica de referencia, de control y de salida al menos 2 veces por segundo, según el enunciado, esto implica un límite superior de 500 ms. Ahora bien, con los límites establecidos anteriormente, hay que considerar que a menores tiempos de muestreo, es más probable que se presente inestabilidad al discretizar el controlador y probarlo experimentalmente. Por lo tanto, por las constantes de tiempo asociadas a la planta (cercanas a 1 s) y para evitar tener *aliasing* significativo, se utilizó un tiempo de muestreo de 100 ms. [2]

El resultado del proceso de discretización resulta en la función de transferencia para el controlador en el dominio z:

$$C(z) = \frac{1.5996\,(z - 0.9594)}{z - 1} \tag{6}$$

A partir de esta expresión, se utilizó la herramienta de MATLAB `Simulink` para simular la respuesta temporal ante una entrada de escalón unitario. En la Figura 9, se muestra el resultado obtenido junto con las especificaciones de tiempo de asentamiento al 2 %, porcentaje de sobrepaso y error estacionario.

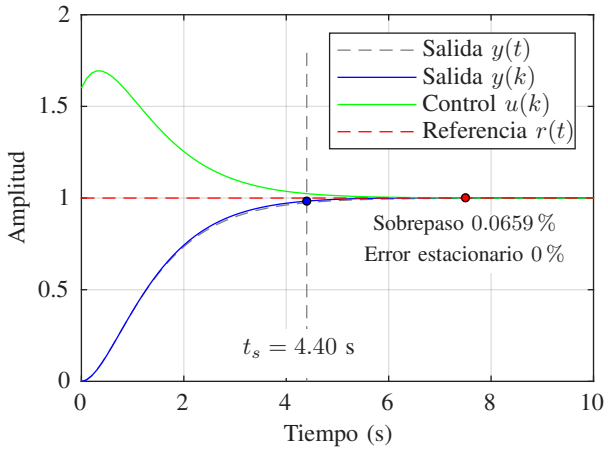

Figura 9. Simulación de la respuesta temporal en lazo cerrado con el modelo del controlador discreto y de la planta continua.

Respecto a las especificaciones obtenidas de la simulación, se tuvo un tiempo de asentamiento al 2 % de 4.40 s, el cual es menor respecto al caso con el controlador continuo. Esto se debe a que, cuando se realiza la discretización, la señal de salida presenta un ligero aumento en su valor, lo cual también afecta el porcentaje de sobrepaso respecto a la señal de referencia, pues se midió un 0.0659 %. Asimismo, al igual que en el caso continuo, se presenta un error estacionario nulo.

Además, respecto a la señal de control, se observa un aumento poco significativo en su amplitud máxima requerida (1.6936 V), lo cual no corresponde a un inconveniente para el controlador. La señal de salida y de control tardan más tiempo en estabilizarse en el valor de referencia, pero esto ocurre para valores menores a la banda del 2 %.

En la Figura 9 se agregó la señal de salida continua a modo de comparación para poder determinar su parecido. Con base en los resultados obtenidos para el caso discreto respecto al continuo, se determinó que el proceso de discretización cumple con los parámetros requeridos y se asemeja a la señal de salida en tiempo continuo.

VI. Implementación del controlador en Arduino

En cuanto a la implementación del controlador en el dispositivo Arduino UNO R3, se requirió obtener una ecuación en diferencias que modele el comportamiento del controlador. Para ello, a partir de (6), en el Apéndice C se obtuvo detalladamente la función que se implementó como controlador en tiempo discreto.

$$u(k) = u(k-1) + 1.5996e(k) - 1.5996 \cdot 0.9594e(k-1) \tag{7}$$

Ahora bien, se empleó una biblioteca correspondiente a `TaskScheduler.h`, la cual fue utilizada para la designación de tareas dentro del programa de Arduino que se repiten en un tiempo específico. Para cambiar entre la referencia del

potenciómetro a una señal cuadrada con período de 12 s, se utilizó un conmutador. [3]

El archivo contiene las siguientes funciones:

- `leer_referencia`: Lectura de la tensión de referencia. Valida si está en el modo del potenciómetro o de señal cuadrada y registra el valor en cada caso.
- `leer_salida`: Lectura de la tensión de la salida para realizar la realimentación y calcular el error.
- `controlador`: Cálculo y escritura del valor de la señal de control.
- `cambiar_referencia`: Cambio de referencia entre el valor alto y bajo de la señal cuadrada.
- `imprimir_datos`: Impresión de los datos en el *Serial monitor*.
- `definir_entrada_switch`: Cambio de referencia entre potenciómetro y señal cuadrada mediante el *switch*.

De las anteriores, se definió una tarea para `controlador` y una para `imprimir_datos` que se repiten cada 100 ms (tiempo de muestreo). En el caso de `cambiar_referencia`, se definió una tarea que se repite cada 6 s para realizar el cambio entre tensión eléctrica alta (3.5 V) y baja (1 V). Finalmente, para `definir_entrada_switch`, se utilizó una tarea para registrar si se da un cambio en el *switch* cada 500 ms, para cambiar la referencia.

Respecto a la asignación de pines, para escribir la tensión eléctrica de entrada a la planta se asigna el pin digital PWM `6`, para medir la salida del circuito se asigna el pin analógico `A2`, para la referencia del potenciómetro el pin `A1` y, finalmente, se asigna la entrada del *switch* en el pin digital `7`.

Específicamente en la función `controlador`, se establecen variables del algoritmo de control a utilizar como la referencia r, la salida y y el error e. También, se consideró que como la señal de control no puede salirse de los límites físicos de tensión eléctrica establecidos por los pines digitales del Arduino, se utilizó el comando `constrain` para limitar la salida entre 0 y 5 V.

Respecto a la resolución de los pines analógicos (lectura) y digitales (escritura), se utilizó el comando `map` para asociar valores entre 0 y 1023 para la lectura a tensiones eléctricas entre 1 y 3.5 V. De forma similar, se tiene una resolución de 0 a 255 para la escritura de tensiones eléctricas entre 0 y 5 V.

Para consultar el código detallado, ver el Apéndice D.

VII. Resultados experimentales

En esta sección, se realizó el análisis de los resultados obtenidos experimentalmente respecto a los simulados en MATLAB con el controlador en tiempo discreto.

El primer requerimiento para el Arduino era poder modificar la tensión eléctrica de referencia a partir de un potenciómetro. En la Figura 10, se muestra la señal de referencia, la de control y la salida resultantes del muestreo realizado cada 100 ms.

En el primer movimiento realizado, se aumenta la tensión eléctrica de referencia hasta 2.02 V. Para este tramo, se registró un tiempo de asentamiento al 2 % de 4.50 s al llegar al valor de salida de 1.98 V. Posteriormente, se realizó otro movimiento

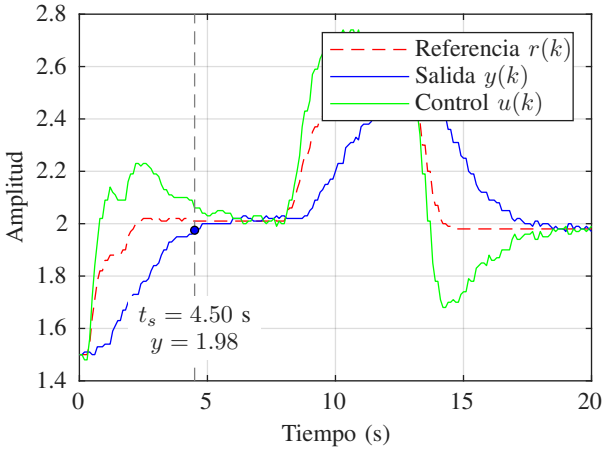

Figura 10. Resultados experimentales obtenidos para el potenciómetro como tensión eléctrica de referencia.

para aumentar la tensión y se volvió a disminuir a 2 V aproximadamente. En esta parte se evidencia el funcionamiento esperado del controlador bajo un tiempo de asentamiento al 2 % aceptable.

En general para esta sección experimental en el Arduino, es importante considerar que las especificaciones obtenidas previamente eran para una entrada de un escalón unitario, como lo solicita el enunciado, bajo condiciones iniciales nulas. Ahora bien, tanto la entrada como las condiciones iniciales son distintas respecto a la suposición realizada. Por ejemplo, con el potenciómetro como referencia, el aumento o disminución de tensión eléctrica no corresponde a un escalón.

El segundo requerimiento experimental corresponde a poder seleccionar la tensión eléctrica de referencia como una señal cuadrada con un período de 12 s y un ciclo de trabajo del 50 %, donde la tensión eléctrica en bajo es 1 V y en alto es 3.5 V. Primeramente, se realizó la simulación del controlador discreto con la planta continua con la entrada descrita anteriormente en `Simulink`. El resultado se muestra en la Figura 11.

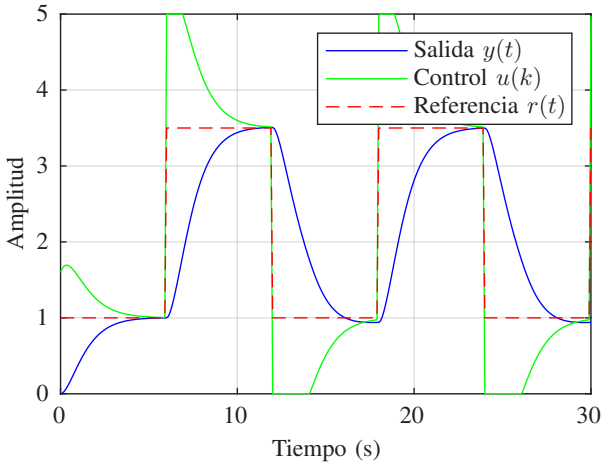

Figura 11. Simulación de la señal cuadrada como tensión eléctrica de referencia.

Según la simulación realizada, se observan dos tramos de interés: la referencia en alto y en bajo. Para la tensión eléctrica en alto de la señal cuadrada, se tiene un valor final aproximado de $3.4992\,\mathrm{V}$. Asimismo, se presenta un tiempo de asentamiento al $2\,\%$ de $4.1\,\mathrm{s}$ aproximadamente, lo cual es más rápido que para el caso del escalón unitario, pues afecta la magnitud mayor de la entrada y las condiciones iniciales.

Por otro lado, para la tensión en bajo, se presenta un valor final de $0.9619\,\mathrm{V}$, la señal no llega a asentarse al $2\,\%$, y se presenta un sobrepaso del $6.18\,\%$. Nuevamente, debido a que el cambio en la entrada presenta una magnitud de 2.5, tiene sentido que no se presente el mismo comportamiento que para el escalón unitario. En este caso, la señal de control se satura durante más tiempo, lo cual afecta también al valor final alcanzado para este tramo.

Ahora bien, respecto a los resultados experimentales alcanzados con el Arduino UNO R3, estos se muestran en la Figura 12. Para la tensión en alto, se presenta un sobrepaso máximo de $0.5714\,\%$ con un valor final de $3.52\,\mathrm{V}$, bajo un tiempo de asentamiento al $2\,\%$ de $4.5\,\mathrm{s}$. Este resultado se asemeja al simulado, con la diferencia de un ligero sobrepaso al alcanzar el valor final respecto a la referencia.

Para la tensión en bajo de la señal cuadrada, se observa un sobrepaso máximo de $7\,\%$ al alcanzar un valor final de $0.93\,\mathrm{V}$. Nuevamente, este resultado es similar al simulado y se verifica la precisión y exactitud del modelo de la planta respecto a la planta real.

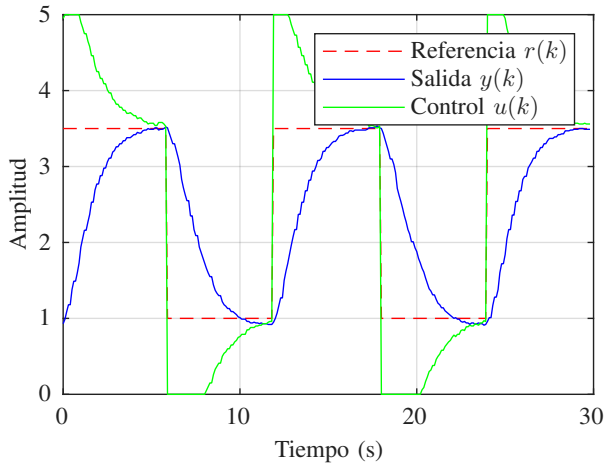

Figura 12. Resultados experimentales obtenidos para la señal cuadrada como tensión eléctrica de referencia.

Como anotación final, el modelo del controlador fue ajustado para obtener los resultados más exactos y precisos de forma experimental con la planta real. Por lo tanto, se varió la ganancia ligeramente hasta llegar al controlador que proporcionó resultados más consistentes con los esperados.

VIII. Conclusiones

Las principales conclusiones determinadas a partir del proceso de diseño e implementación experimental del sistema de control realimentado del proyecto se enumeran a continuación:

1. Se determinó que el circuito RC de la planta corresponde a un sistema de segundo orden, tipo 0 y sobreamortiguado, lo cual se verificó analíticamente mediante la obtención de la función de transferencia. Este entendimiento permitió modelar con precisión la dinámica del sistema y justificó la necesidad de incluir un integrador en el controlador para alcanzar error estacionario nulo, en concordancia con el objetivo de especificación dado en el enunciado del proyecto.
2. El análisis del sistema en lazo abierto evidenció una respuesta lenta y sin sobrepaso, debido al polo dominante en $s = -0.4067$. Este hallazgo fue clave para aplicar la técnica del *Lugar Geométrico de las Raíces* (LGR), mediante la cual se colocó un cero en el controlador que canceló el polo dominante, para reducir el tiempo de asentamiento al $2\,\%$. La ubicación final de los polos en lazo cerrado cumplió simultáneamente con las restricciones de margen de ganancia, fase y sobrepaso máximo, así como la amplitud de la señal de control requerida, lo cual validó el diseño del controlador propuesto.
3. La discretización del controlador se realizó utilizando un tiempo de muestreo de $100\,\mathrm{ms}$, elegido de forma tal que se preserve la estabilidad y fidelidad del sistema al implementarse en el microcontrolador del Arduino UNO R3. Las simulaciones en tiempo discreto demostraron que la respuesta obtenida mantiene las especificaciones requeridas (error estacionario nulo, tiempo de asentamiento al $2\,\%$ menor a $5\,\mathrm{s}$ y sobrepaso menor al $5\,\%$), lo cual validó la eficacia del proceso de discretización y su viabilidad de acuerdo con los límites del hardware.
4. La implementación experimental en el Arduino corroboró el comportamiento previsto por el modelo discreto simulado, tanto para entradas suaves (potenciómetro) como abruptas (señal cuadrada). Se logró reproducir con precisión el tiempo de asentamiento y el sobrepaso esperado, salvo ligeras diferencias atribuibles a la saturación de la señal de control o a condiciones iniciales distintas de las simuladas. Estos resultados experimentales validan tanto el modelo de la planta como la técnica de diseño del controlador.

Referencias

[1] K. Ogata, “Ingeniería de Control Moderna”, 5ta ed. Ciudad de México, México: Pearson Educación, 2010.

[2] G. S. Flores y J. J. F. Uriarte, “MS: Obtención de voltaje DC con la salida PWM de Arduino UNO”, ResearchGate, 2022. [En línea]. Disponible: https://www.researchgate.net/profile/Gonzalo-Soberanes/publication/363107881

[3] Arkhipenko, “TaskScheduler: Cooperative multitasking for Arduino, ESPx, STM32 and other microcontrollers”, GitHub repository, 2022. [En línea]. Disponible: https://github.com/arkhipenko/TaskScheduler. Accedido: Jul. 10, 2025.

APÉNDICE A
CÁLCULOS PARA LA OBTENCIÓN DE LA FUNCIÓN DE TRANSFERENCIA DEL MODELO DE LA PLANTA

En esta sección, se muestra el proceso para obtener la ecuación diferencial para tensión eléctrica de salida v_o en términos de la entrada v_s respecto al tiempo y la función de transferencia que modela el sistema de la planta.

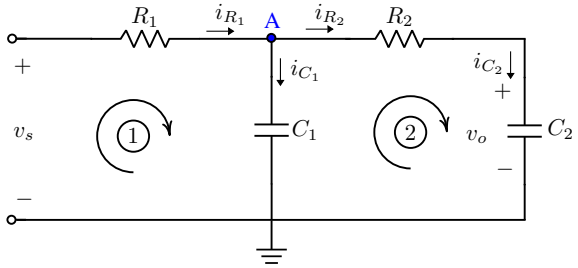

Figura 13. Circuito con leyes de tensiones y corrientes.

Primeramente, se obtienen las ecuaciones utilizando la *ley de Tensiones Eléctricas de Kirchhoff* y la *ley de Corrientes Eléctricas de Kirchhoff*, las cuales se indican en la Figura 13.

LTK siguiendo la trayectoria 1:

$$-v_s + R_1 i_{R_1} + v_{C_1} = 0 \tag{8}$$

LTK siguiendo la trayectoria 2:

$$-v_{C_1} + R_2 i_{R_2} + v_{C_2} = 0 \tag{9}$$

LCK en el punto A:

$$i_{C_1} = i_{R_1} - i_{R_2} \tag{10}$$

Además, se observa que:

$$v_{C_2} = v_o \tag{11}$$

$$i_{R_2} = i_{C_2} \tag{12}$$

Mediante la ecuación que relaciona la tensión y corriente eléctrica del capacitor se tiene:

$$i_{C_1} = C_1 \dot{v_{C_1}} \tag{13}$$

$$i_{C_2} = C_2 \dot{v_{C_2}} \tag{14}$$

Sustituyendo (12) y (14) en (9):

$$v_{C_1} = R_2 C_2 \cdot \dot{v_{C_2}} + v_{C_2} \tag{15}$$

Luego, (12) en (10):

$$i_{R_1} = i_{C_1} + i_{C_2} \tag{16}$$

(16) en (8):

$$-v_s + R_1(i_{C_1} + i_{C_2}) + v_{C_1} = 0 \tag{17}$$

(13) y (14) en (17):

$$-v_s + R_1(C_1 \cdot \dot{v_{C_1}} + C_2 \cdot \dot{v_{C_2}}) + v_{C_1} = 0 \tag{18}$$

Derivando (15):

$$\dot{v_{C_1}} = R_2 C_2 \cdot \ddot{v_{C_2}} + \dot{v_{C_2}} \tag{19}$$

Después, (15) y (19) en (18):

$$-v_s + R_1 C_1 (R_2 C_2 \cdot \ddot{v_{C_2}} + \dot{v_{C_2}}) + R_1 C_2 \dot{v_{C_2}} + (R_2 C_2 \cdot \ddot{v_{C_2}} + \dot{v_{C_2}}) \tag{20}$$

Despejando v_s y simplificando en (20):

$$v_s = R_1 R_2 C_1 C_2 \ddot{v_{C_2}} + (R_1 C_1 + R_1 C_2 + R_2 C_2)\dot{v_{C_2}} + v_{C_2} \tag{21}$$

Finalmente, utilizando (11) en (21):

$$v_s = R_1 R_2 C_1 C_2 \ddot{v_o} + (R_1 C_1 + R_1 C_2 + R_2 C_2)\dot{v_o} + v_o \tag{22}$$

Con esto, se obtiene la ecuación diferencial buscada para el modelo de la planta. Se aplica la Transformada de Laplace a (22) para obtener la función de transferencia que modela la planta, donde se suponen condiciones iniciales cero.

$$V_s = s^2 R_1 R_2 C_1 C_2 \cdot V_o + s(R_1 C_1 + R_1 C_2 + R_2 C_2) V_o + V_o \tag{23}$$

Se acomoda a (23) de la forma $H(s) = \frac{V_o}{V_s}$:

$$H(s) = \frac{1}{R_1 R_2 C_1 C_2 s^2 + (R_1 C_1 + R_1 C_2 + R_2 C_2)s + 1} \tag{24}$$

Finalmente, (24) corresponde a la función de transferencia buscada.

APÉNDICE B
CÁLCULOS PARA LOS LÍMITES DE LAS ESPECIFICACIONES EN EL LGR

En este apartado, se detalla la memoria de cálculos realizados para el proceso de diseño del controlador en tiempo continuo por medio de la técnica LGR, para cumplir con las especificaciones de tiempo de asentamiento al $2\,\%$ y porcentaje de sobrepaso.

Para determinar en qué posición se debe encontrar la línea vertical que limita el tiempo de asentamiento al $2\,\%$ menor que $5\,\mathrm{s}$, a partir de la definición de este requerimiento:

$$t_{s,2\,\%} = \frac{4}{\xi \omega_n} \tag{25}$$

Como se requiere que sea menor a $5\,\mathrm{s}$, se realiza una inecuación, la cual es presentada en (26).

$$\frac{4}{\xi \omega_n} < 5\,\mathrm{s} \tag{26}$$

Al resolver para $\xi\omega_n$, se obtiene:

$$-\xi \omega_n < -0.8 \tag{27}$$

Ahora, para poder obtener el requerimiento del sobrepaso máximo, se va a emplear,

$$M_p = e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}} \tag{28}$$

donde se observa que depende solamente de ξ. Como el sobrepaso debe ser menor que $5\,\%$, entonces se realiza la inecuación y se obtiene el valor de ξ con $M_p = 0.05$.

$$e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}} < 0.05 \tag{29}$$

Con esto, resulta que el valor de ξ debe ser:

$$\xi > 0.6901 \tag{30}$$

Ahora, para poder obtener el ángulo respecto al eje horizontal en el LGR para cumplir con el requerimiento de sobrepaso, se calcula con (31).

$$\theta = \pm \arctan\left(\frac{\sqrt{1-\xi^2}}{\xi}\right) \tag{31}$$

Utilizando el valor del ξ en un sobrepaso del $5\,\%$ (caso límite de $\xi = 0.6901$) en (31), se tiene:

$$\theta = 46.3619^\circ \tag{32}$$

Entonces, este corresponde al ángulo de las líneas oblicuas con respecto a la horizontal, las cuales delimitan los valores para poder cumplir el porcentaje de sobrepaso menor al $5\,\%$.

Apéndice C
Determinación de la ecuación en diferencias para la implementación en Arduino

Para la obtención de la ecuación en diferencias a partir de la función de transferencia discreta obtenida en (6), se debe expresar de la siguiente forma:

$$C(z) = \frac{U(z)}{E(z)} = \frac{1.5996\,(z-0.9594)}{z-1} \tag{33}$$

donde la señal de control U representa la salida del controlador y la entrada consiste en la señal de error E.

Posteriormente, se debe reacomodar la expresión para obtener la ecuación en diferencias de la forma:

$$U(z)\,(z-1) = 1.5996\,(z-0.9594)\,E(z)$$

$$zU(z) - U(z) = 1.5996zE(z) - 1.5996 \cdot 0.9594E(z) \tag{34}$$

Luego, se aplica la Transformada Z Inversa para obtener la ecuación en diferencias.

$$u(k+1) - u(k) = 1.5996e(k+1) - 1.5996 \cdot 0.9594e(k) \tag{35}$$

Finalmente, se aplica la propiedad de desplazamiento en tiempo discreto y se despeja para la señal del controlador $u(k)$:

$$u(k) = u(k-1) + 1.5996e(k) - 1.5996 \cdot 0.9594e(k-1) \tag{36}$$

A modo de aclaración, se resolvió en términos de la magnitud de la ganancia y el cero del controlador discreto para simplificar la implementación en Arduino.

Apéndice D
Implementación del controlador en Arduino

```
#include <TaskScheduler.h>

// Resolucion de lectura y escritura en
    Arduino
#define RESOLUCION_LECTURA 1023
#define RESOLUCION_ESCRITURA 255

// Tiempos de tasks
#define TIEMPO_MUESTREO 100
#define TIEMPO_ALTO_CUADRADA 6000
#define TIEMPO_LECTURA_SWITCH 500

// Referencia de valores para senal cuadrada y
     pot
#define TENSION_ALTO 3.5
#define TENSION_BAJO 1
#define TENSION_MAX 5.0
#define TENSION_MIN 0.0

// Cero y ganancia (4 decimales)
#define Z 0.9594
#define K 1.88

// Entradas y salidas del Arduino
const int Vs = 6;         // Pin PWM
const int Vo = A2;        // Pin Vo
const int Ref = A1;       // Pin Pot
const int Switch = 7;     // Pin Switch

// Variables del sistema de control
float referencia = 0.0;  // Referencia
float salida = 0.0;      // Salida actual
float e = 0.0;           // Error actual
float e_previo = 0.0;    // Error previo
float u;                 // Control actual
float u_previo = 0.0;    // Control previa

// Instanciar el objeto de Scheduler
Scheduler RealTimeCore;

// Senal cuadrada: para alternar entre 1 V y
    3.5 V
bool tension_alta = true;

// Variable para elegir entre:
// potenciometro (true)
// senal cuadrada (false)
bool usar_pot = false;

// Definicion de la funcion del controlador
void controlador() {
  referencia = leer_referencia();
  salida = leer_salida();
  e = referencia - salida;

  // Ecuacion en diferencias del controlador
  u = u_previo + K*e - Z*K*e_previo;

  // Actualizacion de valores de instantes
    previos
  e_previo = e;
  u_previo = u;

  // Limites de senal de control
  u = constrain(u, TENSION_MIN, TENSION_MAX);

  // Escribir senal de control en la entrada
    de la planta
  float u_8b = (u * RESOLUCION_ESCRITURA) /
    TENSION_MAX;
  analogWrite(Vs, u_8b);
```

```
}

// Para determinar switch ON or OFF
void definir_entrada_switch(){
  int estado = digitalRead(Switch);
  if (estado == HIGH) {
    usar_pot = true; // Si el switch esta encendido
  }else if (estado == LOW) {
    usar_pot = false; // Si el switch esta apagado
  }
}

// Funcion para leer la referencia
float leer_referencia() {
  if (usar_pot) {
    // Realizar lectura del potenciometro
    int lectura = analogRead(Ref);

    return map(lectura, 0, RESOLUCION_LECTURA, TENSION_BAJO * 1000, TENSION_ALTO * 1000) / 1000.0;
  } else {
    // Alternar tension para la senal cuadrada
    if (tension_alta == true) {
      return TENSION_ALTO;
    }
    else {
      return TENSION_BAJO;
    }
  }
}

// Funcion para leer la salida de la planta
float leer_salida() {
  return (analogRead(Vo) * TENSION_MAX) / RESOLUCION_LECTURA;
}

// Alternar ref cuadrada
void cambiar_referencia() {
    if (!usar_pot) {
        // Se realiza el cambio periodicamente
        tension_alta = !tension_alta;
    }
}

// Funcion para imprimir los valores al monitor serial
void imprimir_datos() {
  Serial.print("r:");
  Serial.print(referencia);
  Serial.print("\t");
  Serial.print("y:");
  Serial.print(salida);
  Serial.print("\t");
  Serial.print("u:");
  Serial.println(u);
}

// Definir los tiempos de las tareas
Task taskControlador(TIEMPO_MUESTREO, TASK_FOREVER, &controlador, &RealTimeCore, true);

// Periodo de 12 s para la senal cuadrada
Task taskReferencia(TIEMPO_ALTO_CUADRADA, TASK_FOREVER, &cambiar_referencia, &RealTimeCore, true);

// Imprimir datos de r, y, u, cada lectura
Task taskImprimirDatos(TIEMPO_MUESTREO, TASK_FOREVER, &imprimir_datos, &RealTimeCore, true);

// Comprobar si hay cambio en el switch
Task taskEntradaSwitch(TIEMPO_LECTURA_SWITCH, TASK_FOREVER, &definir_entrada_switch, &RealTimeCore, true);

void setup() {
  Serial.begin(9600);

  // Configuracion de switch
  pinMode(Switch, INPUT);

  // Configuracion temporal
  RealTimeCore.startNow();
}

void loop() {
  RealTimeCore.execute();
}
```

Listing 1. Código implementado en Arduino para el sistema de control realimentado.

APÉNDICE E
CÓDIGO DE MATLAB UTILIZADO

```

s=tf('s');

% Valores experimentales
R1 = 9.960*10^(3);
R2 = 9.860*10^(3);
C1 = 94.5*10^(-6);
C2 = 94.7*10^(-6);

% Modelo de la planta
P=1/(R1*R2*C1*C2*s^2 + (R1*C1 + R1*C2 + R2*C2)*s + 1)

% Diagrama de Bode de lazo abierto
fig = figure
margin(P);

% Exportar svg
set(fig, 'Units', 'inches');
fig.Position(3:4) = [5 4]; % Ancho = 5 in, alto = 4 in %
exportgraphics(fig, 'bode_lazo_abierto.svg', 'ContentType', 'vector');

% Grafica de la respuesta al escalon del lazo abierto
% Tiempo
t = 0:0.001:20;

% Funcion escalon unitario
r = 1 * heaviside(t);

% Simular el sistema de lazo abierto
y = lsim(P, r, t);

% Graficar
fig = figure
plot(t, r, 'r--', t, y, 'b')

%Rotulacion de los ejes
xlabel('Tiempo (s)');
ylabel('Amplitud');

% title('Respuesta de lazo abierto');
legend('Referencia r(t)', 'Salida y(t)');
grid on;

```

```
% Sisotool
sisotool(P);

% Convertir el controlador obtenido a discreto
t_muestreo = 0.1;
Cz = c2d(C, t_muestreo, 'zoh')

% Respuesta al escalon del lazo cerrado en tiempo continuo
% Tiempo
t = 0:0.01:6;

% Funcion escalon unitario
r = 1 * heaviside(t);

[y, t1] = step(IOTransfer_r2y, t); % Salida y
[u, t2] = step(IOTransfer_r2u, t); % Senal de control u

fig = figure;
plot(t1, y, 'b');
hold on;
plot(t2, u, 'g');
plot(t, r, 'r--');

xlabel('Tiempo (s)');
ylabel('Amplitud');
legend('Salida y(t)', 'Control u(t)', 'Referencia r(t)');
% title('Respuesta al escalon y senal de control');
grid on;

ts = tiempo_asentamiento.Position(1);
ys = tiempo_asentamiento.Position(2);

% Linea vertical con estilo personalizado (en vez de xline)
plot([ts ts], ylim, '--', 'Color', [0.5 0.5 0.5], 'HandleVisibility', 'off');

% Etiqueta visual con estetica limpia
text(ts, min(ylim) + 0.05, ...
    sprintf('t_s = %.2f s', ts), ...
    'VerticalAlignment', 'bottom', ...
    'HorizontalAlignment', 'center', ...
    'FontSize', 10, ...
    'BackgroundColor', 'white', ...
    'Margin', 2);

% Marcador sobre la curva
plot(ts, ys, 'ko', 'MarkerFaceColor', 'b', 'MarkerSize', 3, 'HandleVisibility', 'off');
hold off;

% Respuesta al escalon de lazo cerrado en tiempo discreto
fig = figure

plot(out.y.Time, out.y.Data, '--', 'Color', [0.5, 0.5, 0.5]);
hold on;
plot(out.yz.Time, out.yz.Data, 'b');
plot(out.uz.Time, out.uz.Data, 'g');
plot(out.r.Time, out.r.Data, 'r--');

% Extraer datos de la senal yz
yz_y = out.yz.Data;
yz_t = out.yz.Time;

% Calcular el valor final y la tolerancia del 2%
yz_final = yz_y(end);
yz_tol = 0.02 * abs(yz_final);

% Determinar los puntos dentro de la banda de 2%
yz_within_bounds = abs(yz_y - yz_final) <= yz_tol;

% Encontrar el tiempo de asentamiento
for k = 1:length(yz_within_bounds)
    if all(yz_within_bounds(k:end))
        yz_ts = yz_t(k);
        break;
    end
end

% Calcular sobrepaso maximo respecto al valor final
[yz_max, idx_max] = max(yz_y);
yz_tmax = yz_t(idx_max);
yz_overshoot = ((yz_max - yz_final) / abs(yz_final)) * 100;

% Indicar tiempo de asentamiento al 2%
plot([yz_ts yz_ts], ylim, '--', 'Color', [0.5 0.5 0.5], 'HandleVisibility', 'off');
plot(yz_ts, yz_y(find(yz_t == yz_ts, 1)), 'ko', 'MarkerFaceColor', 'b', 'MarkerSize', 3);
text(yz_ts, min(ylim) + 0.05, ...
    sprintf('t_s = %.2f s', yz_ts), ...
    'VerticalAlignment', 'bottom', ...
    'HorizontalAlignment', 'center', ...
    'FontSize', 10, ...
    'BackgroundColor', 'white', ...
    'Margin', 2);

% Graficar el punto maximo
plot(yz_tmax, yz_max, 'ko', 'MarkerFaceColor', 'r', 'MarkerSize', 3);
text(yz_tmax-0.25, yz_max-0.25, ...
    sprintf('Overshoot\n%.4f%%', yz_overshoot), ...
    'HorizontalAlignment', 'center', ...
    'FontSize', 9, ...
    'BackgroundColor', 'white', ...
    'Margin', 2);

hold off;

% Labels y leyenda
xlabel('Tiempo (s)');
ylabel('Amplitud');
legend('Salida y(t)', 'Salida y(k)', 'Control u(k)', 'Referencia r(t)');
grid on;

% Diagrama de Bode de lazo cerrado
fig = figure
margin(LoopTransfer_C);

% LGR de lazo cerrado
fig = figure
rlocus(LoopTransfer_C);
```

Listing 2. Código de MATLAB utilizado en el proyecto.