LABORATORIO No 1: CONTROL 2
Emulación de un sistema de tercer orden
(Octubre de 2023)

Mariana Jimenez Duarte
mariana.jimenezd@uqvirtual.edu.co
Universidad del Quindío

Resumen -

En esta práctica se emula un sistema de tercer orden en ESP 32 mediante la obtención de una función de transferencia discreta de acuerdo a un sistema asignado. Para comprobar la emulación implementada, se realiza la simulación en LabVIEW, donde se comparan los datos arrojados de ambos sistemas, los cuales son ingresados y grafiados en MatLab, evidenciando una coincidencia entre ambas señales.

I. INTRODUCCIÓN

A partir de la función de transferencia de un sistema de tercer orden se espera obtener mediante diferentes procesos, su simulación y emulación representadas en una misma gráfica que demuestra similitudes entre sí, por lo tanto, esta práctica es desarrollada en primera instancia mediante el software de simulación LabVIEW y el sistema digital ESP 32 para la obtención de los datos, y posteriormente se emplea MatLab para lograr graficar ambos sistemas en una misma gráfica.

Lo anterior, se desarrolla con el fin de instruirse en el manejo de softwares para simular sistemas dinámicos, emularlos mediante sistemas digitales y acondicionar herramientas de hardware y software para obtener datos que puedan ejemplificar la respuesta de un sistema.

Para alcanzar estos objetivos, es necesario discretizar la función mediante MatLab y un proceso matemático que permita conseguir la función ya discretizada en variables de estado acompañadas de atrasos. Luego, formar el sistema dinámico en LabViEW mediante las variables y conectar dicho programa con la ESP 32 que también contiene dichas variables, para observar mediante un gráfico proporcionado por el simulador la respuesta de los sistemas enfrentados ante un escalón.

Con respecto a lo visto en clase, se propone una secuencia del proceso para elaborar pan, donde se deben proponer diferentes controles en cada sección que especifiquen las variables a controlar para automatizar dicho proceso, lo que se tuvo en cuenta, fueron las tareas más complejas o que requieren una mayor vigilancia para emplear sensores y actuadores específicos.

En el caso del cultivo, se propone automatizar el transporte del trigo hasta el lugar de tratamiento según el peso de cada contenedor, en la industrialización, se vigilan los rangos de temperaturas que se manejan al interior de la fábrica, en el transporte se plantea la adquisición de camiones automáticos y para finalizar, en la comercialización, se tiene en cuenta la calidad del producto final, llevando un inventario de las fechas de ingreso y cuando caducan.

A continuación, se aprecia la estructura del documento:

1. Introducción.
2. Métodos e instrumentos.
3. Resultados y discusión.
4. Conclusiones.
5. Referencias.
6. Anexos

II. MÉTODOS E INSTRUMENTOS

Para obtener la función de transferencia discreta se emplea el comando c2d de MatLab con un tiempo de muestreo de 0.1s elegido arbitrariamente, luego, es expresada en ecuaciones de diferencia consiguiendo una ecuación para la salida del sistema, la cual es utilizada para el hardware en la emulación (ESP 32).

En el caso de la simulación, se utiliza LabVIEW, donde la ecuación ya obtenida es implementada en variables de estado. Posteriormente, el programa es modificado para recibir los datos de la señal emulada, y además, para guardar los datos solicitados (tiempo, entrada, señal simulada y señal emulada) en un archivo de texto.

Ambas señales, ya graficadas, son analizadas visualmente en los parámetros más relevantes: puntos de equilibrio, sobre impulso, tiempo de establecimiento y ganancia.

Por último se ejecuta en MatLab un código que en primera instancia encuentre un modelo lineal de los datos obtenidos de la señal emulada, y finalmente graficar las 3 señales frente a un escalón de 1 a 2 que posibilite realizar comparaciones entre sí.

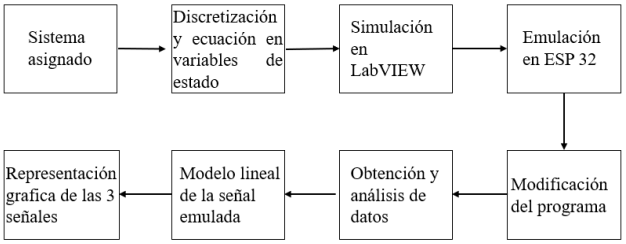

Figura 1. Esquema a seguir.

Tabla 1. Instrumentos utilizados.	
Componente	Referencia
MatLab	N/A
LabVIEW	Q3
IDE Arduino	N/A
(2) ESP	32

MatLab: Es una plataforma que une la programación con cálculos matemáticos para analizar una serie datos y crear modelos que correspondan a dichos datos. Contiene diferentes herramientas en distintas presentaciones, lo que la hacen aplicable al control y otras muchas áreas. [1]

LabView: Es un entorno de programación gráfico que permite realizar pruebas y medidas automatizadas. Cuenta con una amplia gama de herramientas que la hacen útil en muchas aplicaciones de la ingeniería. [2]

III. RESULTADOS Y DISCUSIÓN

Partiendo del siguiente sistema: Gs = tf([0.002706, 0.003666],[1, 0.451666, 0.067500, 0.003333])

Se obtiene la función de transferencia discreta a través del comando c2d en MatLab, el cual acompañado del tiempo de muestreo arroja la función.

```
>> Gz= c2d(Gs,0.1)

Gz =

  1.393e-05 z^2 + 2.19e-06 z - 1.254e-05
  ---------------------------------------
    z^3 - 2.955 z^2 + 2.911 z - 0.9558
```

Figura 2. Función de transferencia discreta.

Posteriormente, se aplica el comando format long para el numerador y denominador de la función para lograr mayor exactitud en la simulación y emulación. Además, a cada número del numerador y denominador se les asigna una variable con el fin de facilitar el procedimiento matemático quedando de la siguiente manera:

$$bo = 1.393197227419905e-05$$
$$b1 = 2.190291844751882e-06$$
$$b2 = -1.253802360601173e-05$$
$$a1 = -2.955176638235637$$
$$a2 = 2.911018122821760$$
$$a3 = -0.955838225919012$$

Ahora, la función es multiplicada y dividida por su mayor exponente elevado de forma negativa para reducir su grado y dar paso a las ecuaciones de diferencia.

$$Gz = \frac{b0\,z^2+b1\,z+b2}{z^3+a1\,z^2+a2\,z+a3} * \frac{z^{-3}}{z^{-3}} \quad (1)$$

Dado que Gz relaciona la salida sobre la entrada, puede ser expresada como: $\frac{Y(z)}{U(z)}$ (2)

$$\frac{Y(z)}{U(z)} = \frac{b0\,z^{-1}+b1\,z^{-2}+b2\,z^{-3}}{1+a1\,z^{-1}+a2\,z^{-2}+a3\,z^{-3}}$$

Solucionando la fracción resulta:

$$Y(z) * (1 + a1\,z^{-1} + a2\,z^{-2} + a3\,z^{-3}) = U(z) * (b0\,z^{-1} + b1\,z^{-2} + b2\,z^{-3})$$

Realizando distributiva:

$$Y(z) + a1\,z^{-1}Y(z) + a2\,z^{-2}Y(z) + a3\,z^{-3}Y(z) = b0\,z^{-1}U(z) + b1\,z^{-2}U(z) + b2\,z^{-3}U(z)$$

Se reemplazan los z y sus respectivos exponentes por atrasos representados por k-

$$Y(z) + a1Y(k-1) + a2Y(k-2) + a3\,Y(k-3) = b0\,U(k-1) + b1\,U(k-2) + b2U(k-3)$$

Finalmente se despeja en términos de Y(k):

$$Y(k) = b0\,U(k-1) + b1\,U(k-2) + b2U(k-3) - a1Y(k-1) - a2Y(k-2) - a3\,Y(k-3)$$

Esta ecuación es la requerida para implementar la simulación del sistema en LabVIEW (ver figura 3), mediante constantes que representan las variables, multiplicadores para conectar los atrasos (Feedback Node) con las variables y el sumador que a su salida refleja a Y(k).

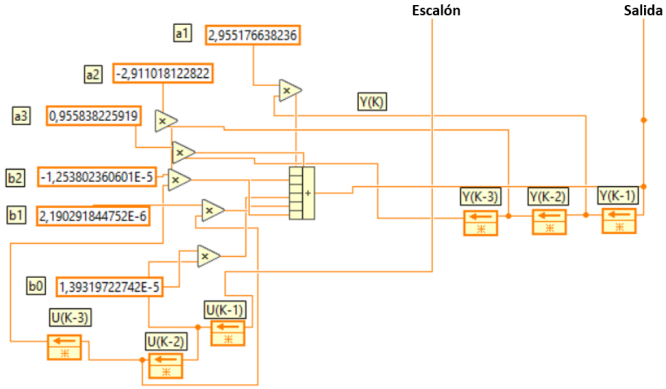

Figura 3. Simulación en LabVIEW.

Para la emulación del sistema, se realiza un código en el IDE de Arduino donde se establece la ecuación encontrada con sus respectivas variables, y la velocidad de transmisión de 115200 baudios, este es cargado a una primera ESP 32 que actúa como la planta del sistema.
La segunda ESP 32 es requerida para leer los datos(ADC) que arroja la planta, almacenarlos y mediante un puerto USB enviarlos (DAC) al programa. También puede utilizarse una tarjeta de adquisición como la National Instruments que cumple con las dos funciones al mismo tiempo.

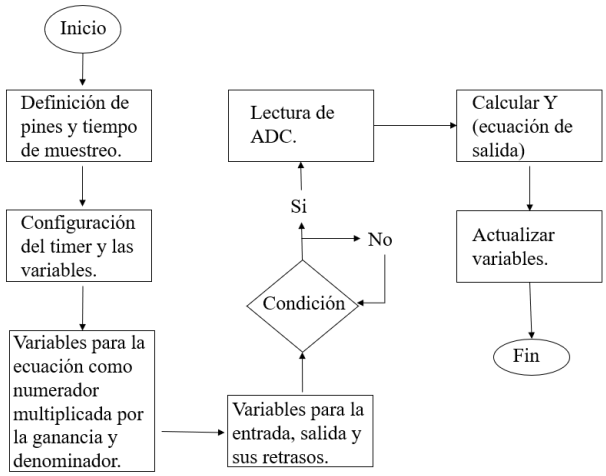

Figura 4. Diagrama de flujo del código que emula la planta.

Para que el programa reciba los datos enviados de la ESP 32 es necesario adicionar algunos bloques que permitan establecer la comunicación.

Como se puede observar en la figura 5, el recuadro número 1 adiciona el puerto con la velocidad ya establecida anteriormente, el cual es comunicado con el recuadro 2, el cual consiste en un bucle que se encarga de revisar si hay o no conexión. En caso de haberla, indica si es correcta o incorrecta, lo que conduce al recuadro 3, que envía un "flecha" en color verde para indicar que la conexión se establece con éxito, o una "x" roja de lo contrario. El recuadro 4 establece el ADC que lee los datos de la planta y luego el DAC para escribirlos.

Es importante resaltar que se asigna el valor máximo de voltaje de 3.2, para no llevarlo al límite de 3.3, valor máximo de voltaje de la ESP 32, al igual que el valor límite de 255 que hace referencia al PWM el cual inicia en 0.

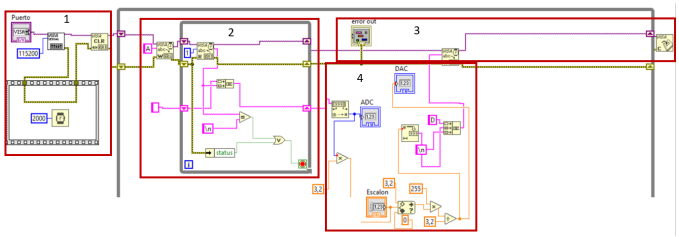

Figura 5. Ajustes a LabVIEW para establecer comunicación.

Es posible observar que se tiene un escalón, el cual es la entrada (U) de la simulación vista en la figura 3.

Por último, como se requiere obtener los datos del tiempo, la entrada y ambas señales, es necesario adicionar un reloj para que contabilice el tiempo de la obtención de datos, además de dos array (opcional) uno de ellos que contenga todas las variables, exceptuando el tiempo, para graficarlas en el programa y reconocer de inmediato si las señales coinciden, el segundo une el array anterior con el tiempo para guardar los datos en un archivo de texto al detener la simulación del botón "stop".

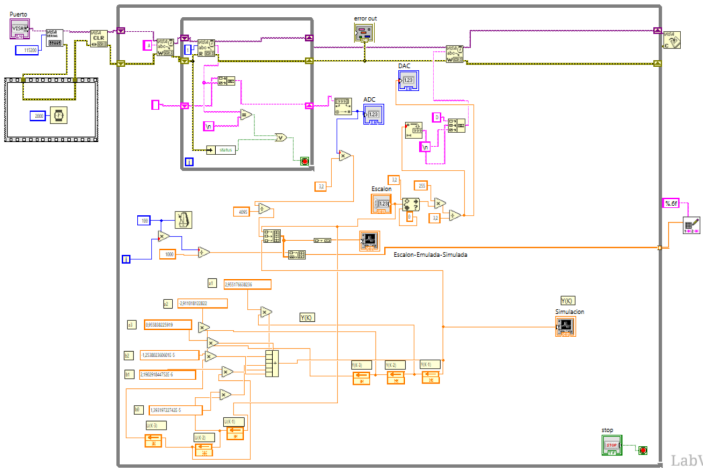

Figura 6. Simulación final vista desde el diagrama de bloques.

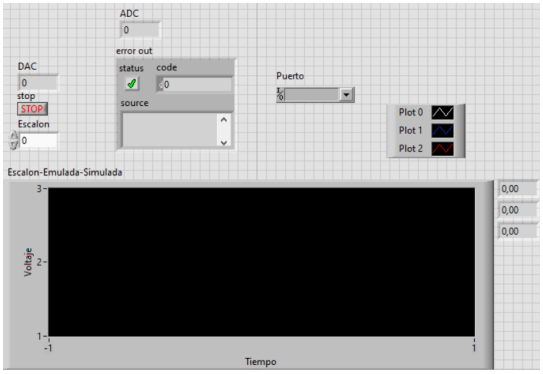

Figura 7. Simulación final vista desde el panel frontal.

Para asegurar que la respuesta emulada sea lo más similar posible a la simulada, se le adiciona una ganancia hallada de forma empírica, en el código cargado a las variables de estado que conforman el numerador en la función de transferencia discretizada.

Con los datos obtenidos, se hallan sus parámetros más relevantes, obtenidos de forma visual, para lo cual se graficaron las respuestas en MatLab ante 2 escalones.

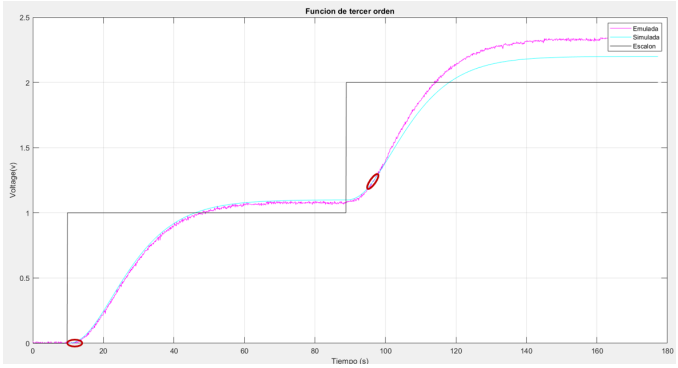

Figura 8. Puntos de equilibrio.

Los puntos de equilibrio representados en la figura 8, muestran los momentos en que ambas respuestas son idénticas, por lo cual se puede inferir que en esos momentos, la entrada y salida del sistema no presentan cambios entre sí.

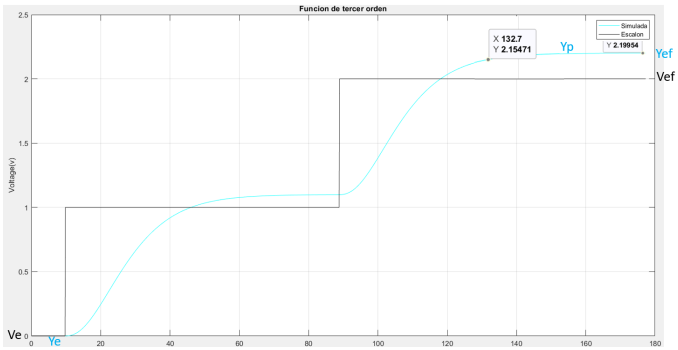

Figura 9. Análisis de la respuesta simulada.

Para hacer el análisis más sencillo se interpretan los resultados por separado, en este caso se inicia con la respuesta simulada, como se observa en la figura 9.

Para hallar la ganancia (G), se utiliza (3) mediante los puntos referenciados, obteniendo una ganancia de 1.099

$$G = \frac{2.19954-0}{2-0} \quad (3)$$

En el caso del sobre impulso (Mp), gráficamente puede concluirse que no lo presenta, sin embargo mediante (4) puede comprobarse.

$$Mp = \frac{Yp-Yef}{Yef-Ye} \quad (4)$$

Dado que Yp es igual que Yef el numerador se hace 0 y con esto el porcentaje de sobre impulso.

Finalmente para el tiempo de establecimiento (te) debe ser asignado un criterio, en este caso del 2%, donde al restarse y sumarse el valor final (Yef) con el criterio se establece un rango crítico y cuando la respuesta ingresa en él, se registra el tiempo. En este caso solo se toma en cuenta el valor restado, ya que la respuesta nunca supera el valor final.

Teniendo en cuenta lo mencionado anteriormente, el tiempo de establecimiento es de 122.7 segundos como se evidencia en (5), el cual toma el valor al entrar al criterio y lo resta con el tiempo en que la señal apenas es afectada por el escalón.

$$te = 132.7s - 10s \quad (5)$$

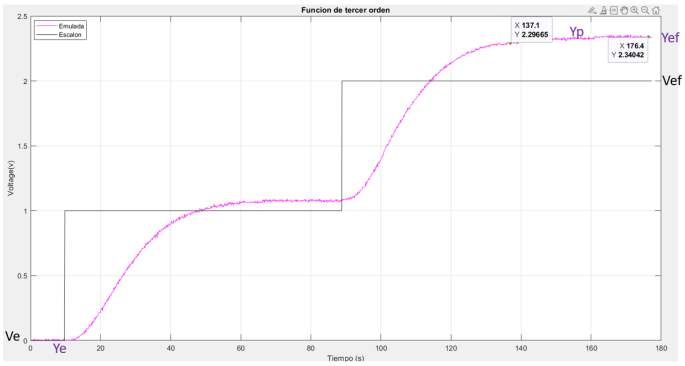

Figura 10. Análisis de la respuesta emulada.

Al igual que en el caso de la simulada, para la respuesta emulada los parámetros se hallan mediante (3), (4) y (5) con los datos y puntos establecidos en la figura 10. Teniendo una ganancia de 1.17021, un sobre impulso de 0% y un tiempo de establecimiento de 127.1 segundos.

Por último, se encuentra un modelo lineal para la respuesta emulada con los datos adquiridos desde el inicio de esta práctica, modelo hallado con ayuda de MatLab, mediante la herramienta system identification.

Lo se debe tener en cuenta para emplear esta herramienta es agregar los datos del escalón mejor respuesta, en este caso el primero, realizar la simulación en el dominio del tiempo, definir el tiempo de muestreo, el mismo que se utilizó al discretizar la función inicial y definir la cantidad de polos y ceros que se estima tiene la función (3 y 1 respectivamente). Lo que se espera es encontrar una función de transferencia similar a la proporcionada al inicio de la práctica.

```
  From input "u1" to output "y1":
          -0.0008592 s + 0.004857
  ---------------------------------------
  s^3 + 0.5713 s^2 + 0.08754 s + 0.004507
Name: tf1
Continuous-time identified transfer function.

Parameterization:
   Number of poles: 3   Number of zeros: 1
   Number of free coefficients: 5
```

Figura 11. Modelo lineal hallado.

System identification también ofrece el cálculo de estimación entre los datos y el modelo hallado, en este caso fue del 98.46% como se muestra en la figura 12.

```
RESULT
Termination condition: Near (local) minimum, (norm(g) < tol)..
Number of iterations: 3, Number of function evaluations: 7

Status: Estimated using TFEST
Fit to estimation data: 98.46%, FPE: 4.27767e-05
```

Figura 12. Porcentaje de estimación.

Al final, es exportado el modelo lineal al Workspace de MatLab donde es graficado mediante el comando "plot" como se observa en la figura 13.

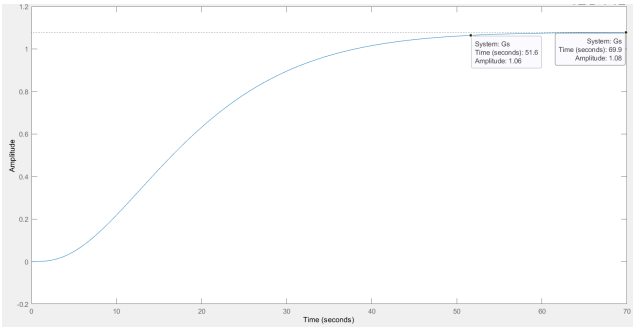

Figura 13. Respuesta del modelo lineal.

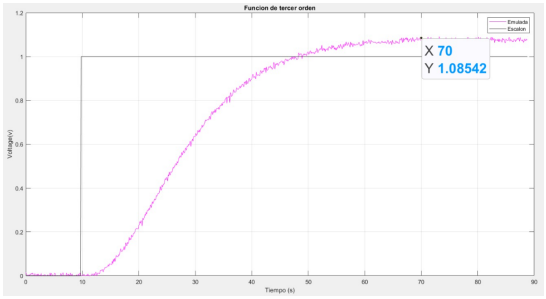

Figura 14. Respuesta emulada ante 1 escalón.

Entre la figura 13 y 14 se encuentran similitudes muy cercanas de manera visual en el tiempo de establecimiento y la amplitud final como se especifica en la tabla 2, por lo cual, puede concluirse que el modelo lineal es correcto aunque no presente una gran similitud en sus coeficientes al comparar la figura 11 con la función proporcionada al inicio.

Tabla 2. Comparación entre la respuesta emulada y la del modelo lineal.

	Emulada	Modelo lineal	%E
te (s)	70	69.9	1.14
A (v)	1.085	1.08	0.46

Retomando las tres respuestas obtenidas, estas son enfrentadas ante un mismo escalón y graficadas al tiempo (ver figura 15), donde permite establecer una comparación más eficiente.

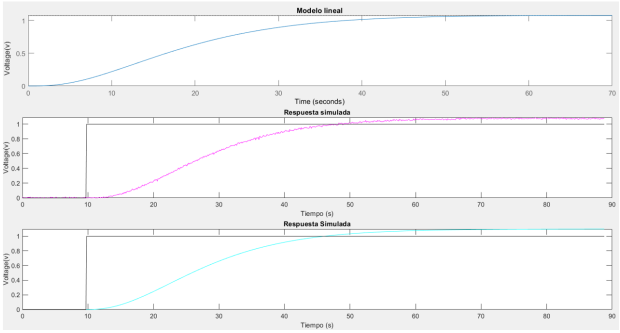

Figura 15. Representación de las 3 respuestas obtenidas.

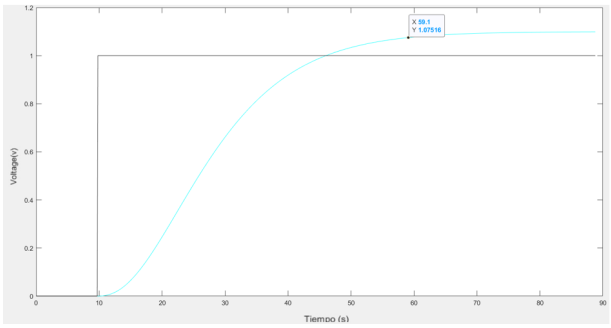

Figura 16. Respuesta simulada ante un escalón.

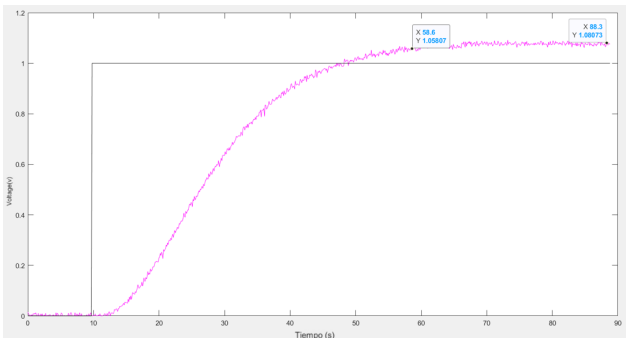

Figura 17. Respuesta emulada ante un escalón.

Tabla 3. Comparación entre las 3 respuestas

	Simulada	Emulada	Modelo lineal
te (s)	49.1	48.6	51.6
G (v)	1.099	1.08073	1.08
Mp	0	0	0

Para hallar el tiempo de establecimiento de la respuesta simulada ante un escalón, se empleó el mismo criterio del 2% mediante (5) y de igual manera para la respuesta emulada.

Para hallar la ganancia, teniendo un escalón de 1, se registra el valor final de la respuesta, en el caso de la simulada es la misma que se obtuvo anteriormente, ya que aumenta en igual proporción ante cada escalón, caso contrario para la emulada,

por lo cual se toma el dato desde la figura 17. Ambos datos son registrados en la tabla 3.

De lo anterior, se obtienen 3 distintos valores para la ganancia, con una media de 1.086 y una desviación estándar de $\pm$ 0.011 y para el tiempo de establecimiento, con una media de 49.76 y una desviación estándar de $\pm$ 1.61, con estos valores se evidencia la poca presencia de error entre las respuestas, donde además prevalece el porcentaje de sobre impulso en 0, lo que comprueba el mismo comportamiento en las 3 respuestas.

IV. CONCLUSIONES

- Es posible identificar el comportamiento de un sistema conociendo únicamente su modelo matemático, a partir del cual se establecen arreglos matemáticos que conllevan a la simulación del mismo.
- Simular el comportamiento de un sistema requiere someterlo a diferentes cambios de entrada, con dichos cambios se obtiene su modelo de respuesta.
- Además de conocer el comportamiento ideal, es posible implementar un sistema que emule la planta, someterla a diferentes cambios de entrada y conocer una respuesta más cercana a la realidad.
- Establecer diferentes tipos de respuesta ante un mismo sistema genera mayor precisión sobre los datos que pueden ser esperados en el sistema real.

V. REFERENCIAS

[1] MathWorks.(s.f). Matemáticas. Gráficas, Programación. MATLAB - El lenguaje del cálculo técnico (mathworks.com)

[2] ni.(s.f). ¿Qué es LabVIEW? ¿Qué es LabVIEW? Programación gráfica para pruebas y medidas - NI

VI. ANEXOS

- Código de MatLab para graficar las 3 respuestas al tiempo.

```
Gs=tf([-0.000859229519904
0.004856925659326],[1      0.571251563030952
0.087543393998215 0.004507036192907]);

DATOSS = load("datoLineal.txt");

datosenx = DATOSS(:,1);

datoseny = DATOSS(:,3);

datoSimulada = DATOSS(:,4);

datoEscalon = DATOSS(:,2);

figure(1);

subplot(3,1,1)

step(Gs);

title('Modelo lineal');

subplot(3,1,2);

plot(datosenx,  datoseny,  'm',  datosenx,
datoEscalon, 'k');

title('Respuesta simulada');

xlabel('Tiempo (s)');

ylabel('Voltage(v)');

subplot(3,1,3);

plot(datosenx, datoSimulada, 'c', datosenx,
datoEscalon, 'k');

title('Respuesta Simulada');

xlabel('Tiempo (s)');

ylabel('Voltage(v)');
```