LABORATORIO No 1: Emulación de un sistema de tercer orden.
(Noviembre de 2023)

Laura Valentina Buitrago Diaz
laurav.buitragod@uqvirtual.edu.co
Universidad del Quindío

Resumen

En este laboratorio se realizó un breve análisis de la automatización de la producción y repartición de la leche y un análisis detallado de la emulación de un sistema dinámico usando un sistema digital y una tarjeta para la adquisición de datos, con el propósito de tomar, observar y modelar en LabVIEW y Matlab los datos obtenidos de la emulación del sistema.

I. INTRODUCCIÓN

En el presente informe se expone en primer lugar la actividad realizada en clase la cual se pretende la automatización del procedimiento que se lleva a cabo para la recolección, producción y repartición de la leche, esto se logró mediante un análisis grupal, el cual se consideró cada etapa necesaria para obtener leche de buena calidad.

Para la primera etapa se tiene la alimentación de las vacas, para la cual se pretende tener el control de la dietas de las vacas para obtener una buena leche. En la segunda etapa que es la recolección de leche, se desea tener un control de la calidad de la leche que se está recogiendo de la vaca descartando así, la recolección de leche en mal estado. Para la tercera y cuarta etapa, la cual es la pasteurización y empaquetado de la leche se desea tener el control de la temperatura, nivel y sellado garantizado que el producto que salga a la venta sea apropiado para el consumo de las personas. Para la etapa de transporte se concluyó que es necesario el tema de la temperatura para evitar daños del producto al trasladarlo en diferentes comercializadoras y garantizar la calidad del producto.

En la práctica de laboratorio se presenta la emulación de un modelo de sistema dinámico que utiliza una función de transferencia de tercer orden, para lograr esto, se hizo uso del Arduino Uno. El objeto principal será representar tanto la señal de entrada como la señal de salida del sistema dinámico.

Se realizó la emulación del sistema dinámico y se utilizó el sistema digital para el circuito Arduino Uno, el cual, simula y controla el comportamiento del sistema a través de la función de transferencia. El modelo ESP32 se utiliza como dispositivo de adquisición para adquirir información del sistema para su uso, esta tarjeta se encargó de recopilar los datos obtenidos del comportamiento del sistema, los cuales fueron visualizados y analizados utilizando el software LabVIEW.

Gracias a estos instrumentos se permitió tener un análisis del desempeño del sistema dinámico emulado. Los datos adquiridos y su representación gráfica en los programas de LabVIEW y Simulink proporcionaron una vista del comportamiento del sistema, facilitando el análisis completo de sus servicios.

Al finalizar esta práctica se habrá analizado y comprendido sobre el proceso de emulación de un sistema dinámico de tercer orden utilizando la tarjeta Arduino uno, así como la adquisición y análisis de datos mediante el uso de la tarjeta ESP32.

II. MÉTODOS E INSTRUMENTOS

A. Instrumentos

Softwares:

LabVIEW, Matlab y Arduino.

Tabla 1. Lista de componentes.

Cantidad	Elemento	Referencia o valor
1	Microcontrolador	Arduino Uno
1	Microcontrolador	Esp32
1	Capacitor	$10\mu F$
1	Resistencia	$1k\Omega$

B. Métodos.

Para poder emular el sistema dinámico en uno digital fue necesario convertir la función de transferencia de tiempo continuo a tiempo discreto conocido como la transformada Z. Se implementó un script en Matlab para realizar dicha transformada con un tiempo de muestreo de 0.1 segundos y se graficó las dos funciones de transferencia con el fin de compararlas, verificando su comportamiento.

```
clear all
close all
clc

Nm = [0.002326, 0.003127];
Dm = [1, 0.448148, 0.066666, 0.003292];

Gs = tf(Nm, Dm)

format long
Gs_Z = c2d(Gs, 0.1)

figure("Name", "FT tiempo continuo")
step(Gs,90)

figure("Name", "FT tiempo discreto")
step(Gs_Z,90)
```

Figura 1. Script transformada de tiempo continuo a discreto.

Para la emulación del sistema se propuso una función de transferencia de tercer orden de la cual se obtuvo un función discreta.

Se realizó en el IDE de Arduino para emular el sistema dinámico como se presenta en la figura 2.

```
#define ADC0pin A0
#define PWMPin 10

//tiempo de muestreo =64us*reload*lim_counter
byte reload = 125;
int lim_counter = 0.1*125;//1 segundo=64us*125*125
volatile int interruptCounter;

ISR(TIMER2_COMPA_vect)
{
  interruptCounter++;
  OCR2A=reload;
}
void setup() {
  Serial.begin(115200);
  cli();
  TCCR0B=0;
  OCR2A=reload;
  TCCR2A=(1<<CS22)|(1<<CS21)|(1<<CS20);
  TIMSK2=(1<<OCIE2A);
  sei();
  TCCR1B=TCCR1B & 0b11111001;
}
const float b0 = 1.197292430719E-5, b1 = 1.868607726887E-6, b2 = 1.078373477935E-5;
const float a1 = 2.95552102103, a2 = 2.911698789138, a3 = 0.9561745489626;
float u = 0, u_1 = 0, u_2 = 0, u_3 = 0;
float y = 0, y_1 = 0, y_2 = 0, y_3 = 0;
void loop() {
  if(interruptCounter>=lim_counter)
  {
    interruptCounter-=lim_counter;
    u=analogRead(ADC0pin)*5.0/1023.0;
    y = (b0*u_1) + (b1*u_2) - (b2*u_3) + (a1*y_1) - (a2*y_2) + (a3*y_3);
    u_3 = u_2;
    u_2 = u_1;
    u_1 = u;
    y_3 = y_2;
    y_2 = y_1;
    y_1 = y;
    if(y<0) y=0;
    if(y>5) y=5;
    analogWrite(PWMPin,y*255/5);

    Serial.print(u);
    Serial.print(",");
    Serial.println(y);
  }
}
```

Figura 2. Emulación desde el IDE de arduino.

Para el funcionamiento de este código se realiza las conexiones como se muestra en la figura 3, realizando un corto para observar y comprobar la respuesta de la función de transferencia de tercer orden.

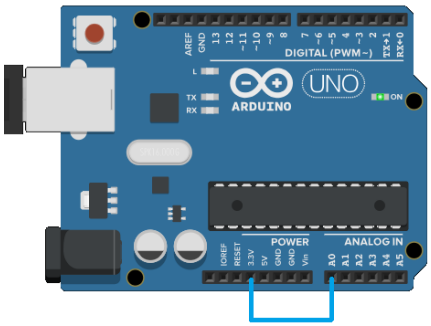

Figura 3. Conexiones del Arduino UNO.

Al obtener los datos de entrada y de salida del sistema, y al mostrarlos en el monitor serial de la IDE, se visualizó el comportamiento en el serial plotter. Posteriormente se comprueba el sistema por medio de la simulación desde el software LabVIEW.

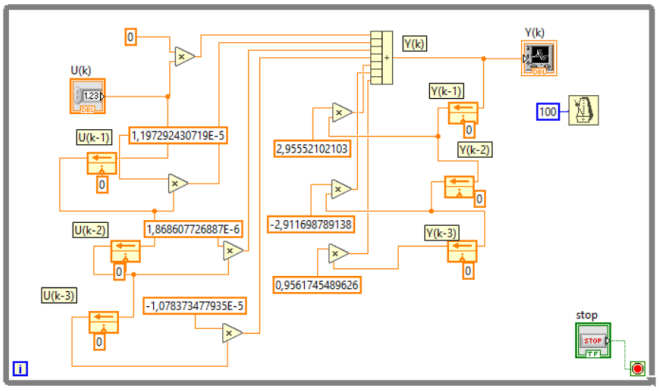

Figura 4. Diagrama de la simulación de la función de transferencia desde LabVIEW.

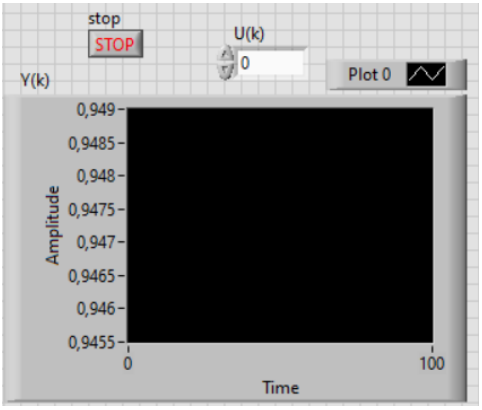

Figura 5. Panel de control.

Luego de comprobar por medio de la simulación del sistema, utilizando la tarjeta Arduino Uno, se utilizó la tarjeta ESP32 como adquisición de datos facilitando el manejo y la visualización de datos en un Software, en este caso se toman los datos obtenidos del comportamiento del sistema en el software de LabVIEW.

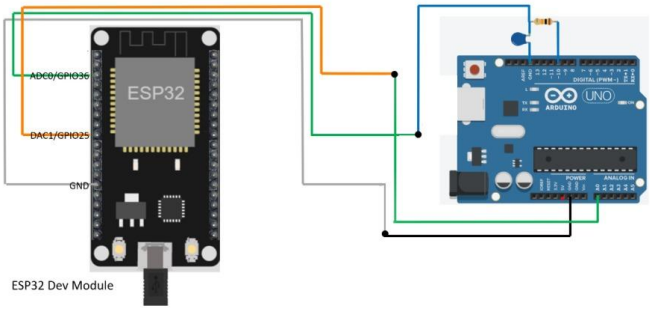

Figura 6. Conexiones del Arduino UNO a la tarjeta ESP32.

Se utilizó el software para procesar la señal emulada por la tarjeta Arduino Uno y recibir los datos a través de la comunicación serial con la tarjeta ESP32 en LabVIEW.

Se realizó la programación de la tarjeta de adquisición para el software por medio del IDE de Arduino utilizando las librerías requeridas para la tarjeta ESP32 como se muestra en la figura 7.

```
#include <dummy.h>
#define DAC1pin 25
#define ADC0pin 36
char seleccion;
int sensor = 0;
int control = 0;
void setup() {
  Serial.begin(115200);
}
void loop() {
  if (Serial.available() > 0)  {
    seleccion = Serial.read();
    if(seleccion=='A')
    {
      sensor = analogRead(ADC0pin);
      Serial.println(sensor);
    };
    if(seleccion=='D')
    {
      control = Serial.parseInt();
      dacWrite(DAC1pin,control);
    };
  };
}
```

Figura 7. Programa de la tarjeta de adquisición con simulink.

Para simular y emular el sistema desde LabVIEW se realizó el siguiente diagrama que permite observar el comportamiento del mismo.

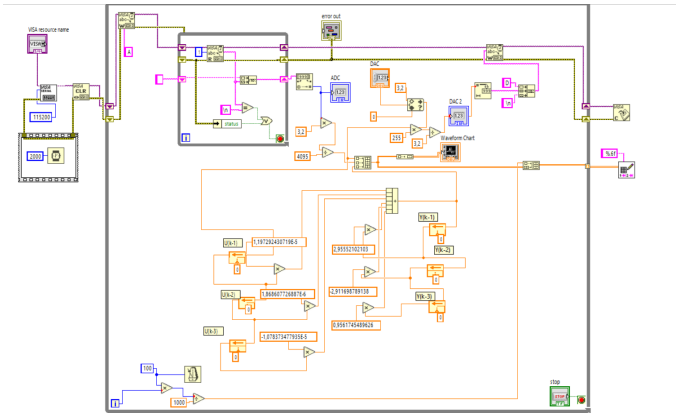

Figura 8. Diagrama desde LabVIEW.

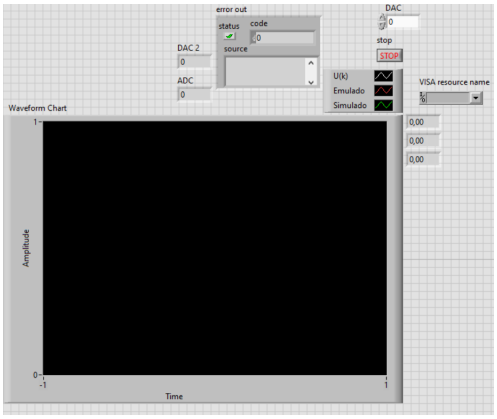

*Figura 9.*Panel de control.

Por último se graficaron los datos obtenidos en matlab de la señal emulada visualizando los datos obtenidos.

III. ANÁLISIS Y RESULTADOS.

Se parte de la función de transferencia de tercer orden en tiempo continuo.

$$G(s) = \frac{0.002326\,s + 0.003127}{s^3 + 0.448148\,s^2 + 0.066666\,s + 0.003292}$$

Al graficar la función de transferencia se obtuvo la figura 10.

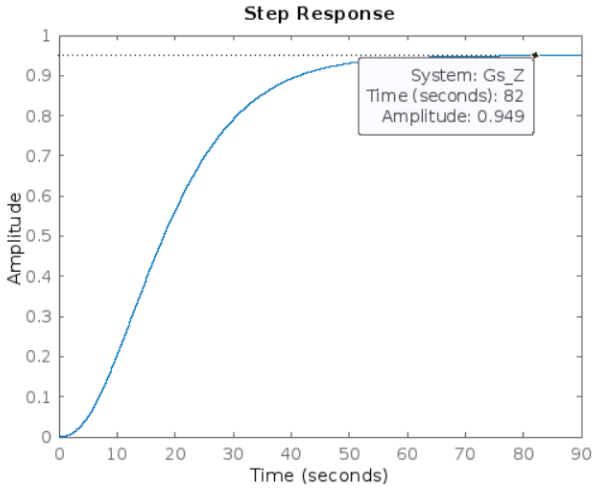

Figura 10. Función de transferencia ideal.

Teniendo la función de transferencia discreta $G(z)$ se realiza la transformada a ecuaciones de diferencia para realizar la emulación en la IDE de Arduino.

$$G(z) = \frac{y(z)}{u(z)} = \frac{b0+b1z+b2}{z^2+a1z+a2} \qquad (1)$$

De la ecuación (1) se procede a realizar el procedimiento necesario para obtener Y(k) obteniendo así:

$$Y(k) = b1u(k-1) + b2u(k-2) + b3u(k-3)$$

$$- a1y(k-1) - a2y(k-2) - a3y(k-3) \qquad (2)$$

Teniendo los siguientes parámetros se agregan según lo obtenido con la ecuación (2) a la simulación de LabVIEW y al IDE de Arduino.

$b0 = 1.197e-05 \qquad a1 = -2.956$
$b1 = 1.869e-06 \qquad a2 = 2.912$
$b2 = 1.08e-05 \qquad a3 = -0.9562$

Se observa que el tiempo de establecimiento es de 0.949 ya que la gráfica se estabiliza en este punto.

Para determinar el porcentaje de sobre impulso se observó la gráfica de la figura 10, utilizando la siguiente ecuación (3).

$$MP = \frac{Ymax - Yf}{Yf} \qquad (3)$$

$$\%MP = 0\%$$

Esto teniendo en cuenta que en la gráfica no se observa un sobre impulso por tanto el porcentaje es del 0%

Para hallar la ganancia se utiliza la ecuación (4).

$$K = \frac{Yef - Ye}{Uef - Ue} \qquad (4)$$

$$K = \frac{0.949 - 0}{1 - 0} = 0.949$$

En la figura 11 se observa el comportamiento del sistema desde el serial plotter del IDE de Arduino teniendo en cuenta que se tiene un escalón de 3.3v, por lo tanto la respuesta será la misma obtenida en matlab pero con un escalón mayor

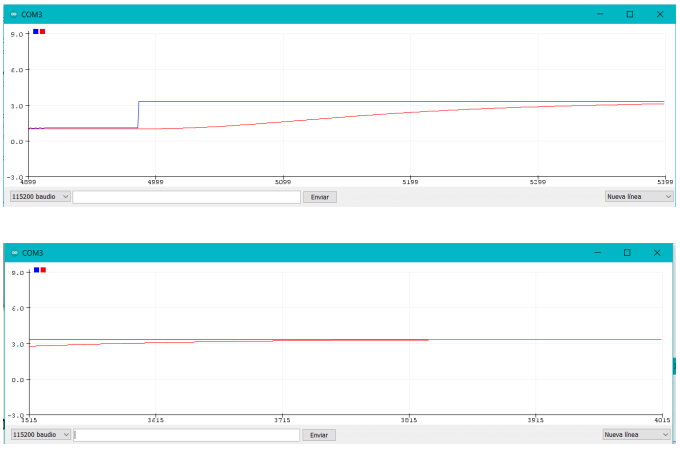

Figura 11. Emulación del sistema desde el IDE de Arduino.

Por medio del software se comprueba la función de transferencia en donde se estabiliza en 0.95 como se observa en la figura.

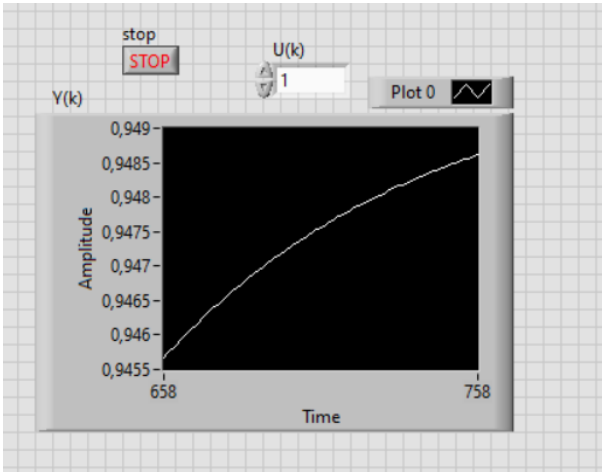

Figura 12. Emulación del sistema desde LabVIEW.

En las figura 13 se visualizan las respuesta obtenida del sistema desde el programa LabVIEW.

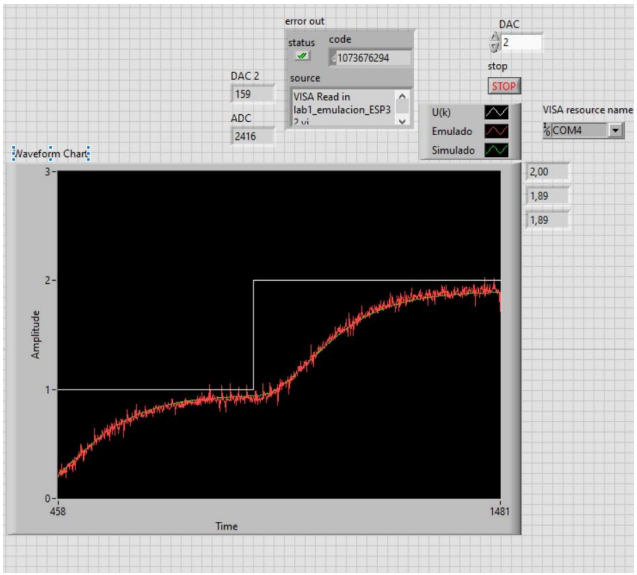

Figura 13. Emulación del sistema desde LabVIEW.

Ya tomados los datos de la respuesta de la emulación del sistema se observan desde matlab las gráficas obtenidas en la figura 14, comprobando los datos calculados y visualizando los puntos de equilibrio.

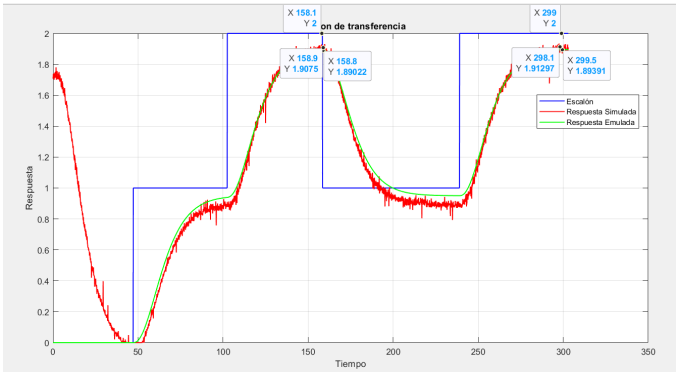

*Figura 14.*Puntos de equilibrio.

En la figura 14 se observan los puntos de equilibrio del sistema emulado y simulado, lo que representa que tanto la entrada como la salida del sistema tienen el mismo comportamiento.

Tabla 2.Puntos de equilibrio.

S. Simulado	S.Emulado	Escalón
1.907	1.89	2

Posteriormente se obtuvo el modelo lineal del sistema a partir de los datos tomados con ayuda del script que se muestra en la figura 15.

```
num = load ("modelo2.m")
% [num, m, raw] = dlmread("lab1datos.m");
Tiempo = num(:,1);
Vi = num(:,2);
emulada = num(:,3);
% simulada = num(:,4);
step(tf2)
```

*Figura 15.*Script desde matlab.

```
RESULT
Termination condition: Near (local) minimum, (norm(g) < tol)..
Number of iterations: 12, Number of function evaluations: 26

Status: Estimated using TFEST
Fit to estimation data: 95.62%, FPE: 0.000227531
```

*Figura 16.*Estimación del sistema.

Se obtuvo la función de transferencia que se muestra en la figura 17, observando así el modelo lineal en la figura 18.

```
  From input "u1" to output "y1":
          -0.01635 s + 0.006663
  ---------------------------------------
  s^3 + 0.7935 s^2 + 0.134 s + 0.007509
Name: tf2
Continuous-time identified transfer function.

Parameterization:
   Number of poles: 3   Number of zeros: 1
```

*Figura 17.*Función de transferencia.

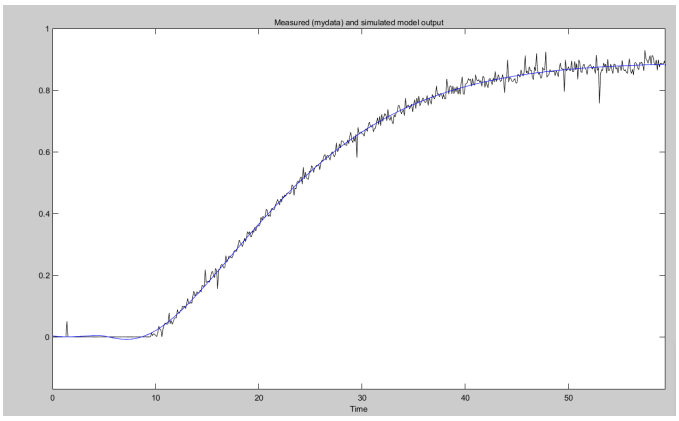

Figura 18. Modelo lineal.

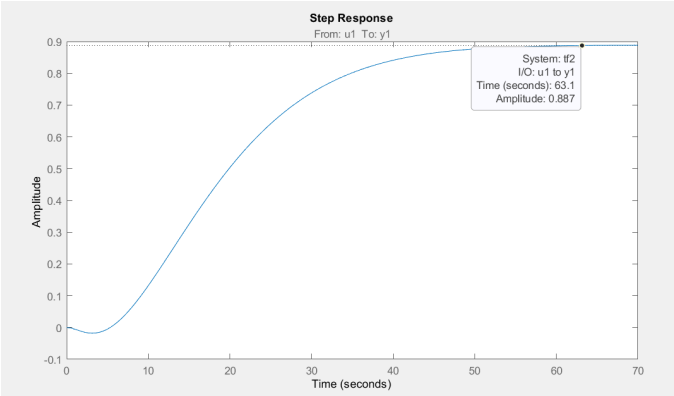

*Figura 19.*Datos del modelo lineal.

Tabla 3.Comparación de las gráficas obtenidas .

Datos obtenidos	Simulado LabVIEW	Emulado	Modelo Lineal
T. Establecimiento	0.949	0.95	0.887
Ganancia	0.949	0.95	0.887
% Sobreimpulso	0	0	0

IV. Conclusiones

La emulación exitosa del sistema dinámico utilizando la placa ESP32 dio como resultado una representación precisa de las señales de entrada y salida del sistema, que formaron la base para el análisis del comportamiento del sistema que desempeñó un papel clave en la evaluación y verificación de los diseños de los sistemas.

Esta fase de modelado es importante para identificar y resolver problemas de diseño. Además, la emulación permite mejorar el diseño hasta obtener un mejor resultado, lo que conduce a la creación de sistemas más fiables, seguros y eficientes.

Para integrar estos dispositivos y herramientas, se proporcionó una imagen detallada y completa del rendimiento del sistema emulado, lo que resultó en una evaluación detallada de su rendimiento.

V. Referencias

[1]. Cangrejo, Daniel (2023) Figura 6.Conexiones del Arduino UNO a la tarjeta ESP32.
[3] Buitrago, Laura (2023) Figura 3. Conexiones del Arduino UNO.