Memorias del XX Congreso Latinoamericano de Control Automático (CLCA 2025)
13-17 de Octubre, 2025. Cancún, Quintana Roo, México

Implementation of master-observer schemes through data acquisition boards using Python ⋆

Tonalli Galicia *, **Luis N. Coria**, **Paul J. Campos**, **Diana Gamboa.**

* *Postgraduate Program in Engineering Sciences, Dynamics Systems and Control, Tecnológico Nacional de México/ IT Tijuana, Blvd. Alberto Limón Padilla s/n, Tijuana 22454, México (e-mail: tonalli.galicia@tectijuana.edu.mx, luis.coria@tectijuana.edu.mx, paul.campos@tectijuana.edu.mx, diana.gamboa@tectijuana.edu.mx)*

Abstract:
This work presents the implementation of ordinary differential equation systems and observer schemes using data acquisition boards programmed in Python. The proposed approach enables the emulation of autonomous linear and nonlinear systems of differential equations, generating electrical signals proportional to the system's dynamics. This methodology is also used to implement an observer, allowing the estimation of the master system to be visualized through electrical signals.

Keywords: Data acquisition, Python, Emulator, Observer.

1. INTRODUCCIÓN

La simulación así como la emulación de la dinámica de sistemas de ecuaciones diferenciales ordinarias (EDO) ha cobrado relevancia gracias a que permiten el desarrollo de esquemas de control, estimación y validación de modelos dinámicos, Muñoz-Sánchez et al. (2025). Este tipo de aplicaciones permite replicar de forma física y programable a través de diversos dispositivos como microcontroladores, Pérez-Cruz et al. (2020), el comportamiento de sistemas lineales y no lineales lo cual ha influido en diversos campos como son la biomedicina, Valle et al. (2021) , las comunicaciones y seguridad, Duran et al. (2018).

Por otra parte, el uso de tarjetas de adquisición de datos (DAQ's) se ve beneficiado de las aportaciones de software open source como lo son las herramientas que dota Python, Weber (2021), lo que facilita su implementación permitiendo un procesamiento y generación de señales electricas analógicas y/o digitales, favoreciendo su uso en distintos contextos.

Realizar la sincronización de las diversas operaciones que suponen la emulación de sistemas dinámicos donde es necesaria la lectura, cálculos y escritura de señales eléctricas, exigen el uso de operaciones en paralelo, Mattson et al. (2021) para así garantizar una reproducción de manera precisa.

⋆ Los autores agradecen al TecNM con los proyectos 22680.25-P y 23154.25-P. Este trabajo fue desarrollado en el marco de la "Red Internacional de Control y Cómputo Aplicados" soportada por el TecNM.

En este trabajo se presenta la implementación de sistemas autónomos de EDO y esquemas de observador utilizando tarjetas de adquisición de datos de National Instruments, programadas mediante Python con el uso de operaciones en paralelo. Si bien se han desarrollado aplicaciones que involucran estas tecnologías (Sanguino (2023), Jäger and Gümmer (2023)), la metodología propuesta permite definir fácilmente y emular la dinámica de sistemas autónomos lineales y no lineales haciendo uso de los canales de entrada y salida analógicos de las tarjetas DAQ's, mediante la generación y lectura de señales facilitando la interacción con dispositivos físicos así como la visualización directa de las dinámicas del sistema.

Además bajo el mismo principio se desarrolla un esquema maestro-esclavo, donde las variables reconstruidas son traducidas a señales eléctricas que pueden ser analizadas y validadas experimentalmente. Con la finalidad de hacer accesible dicho recurso, se diseña una interfaz visual que facilita la interacción con el usuario. Esta aproximación resulta útil en entornos de laboratorio para fines educativos, Cardona and Gutiérrez (2022), de verificación o como etapa previa al diseño de controladores en hardware, Olmo López (2025).

La estructura del trabajo es la siguiente: en la Sección 2 se presenta la metodología de diseño de las aplicaciones, así como la secuencia de las operaciones. La Sección 3 describe las características del hardware y software utilizados, junto con una prueba de fiabilidad basada en el análisis espectral de una señal caótica. Además, se presentan dos modelos de observadores no lineales tomados de la literatura para validar el uso de las aplicaciones desarrolladas. En la Sección 4 se muestran los resultados de la implementación de dichos sistemas, y

https://doi.org/10.58571/CNCA.AMCA.2025.065

finalmente, en la Sección 5, se discuten los resultados, se exponen las conclusiones y se plantea el trabajo a futuro.

2. METODOLOGÍA

En este trabajo se presentan dos aplicaciones con interfaz gráfica desarrolladas en Python mediante la librería `tkinter`, las cuales permiten emular el esquema maestro-esclavo para sistemas de EDO's con el uso de tarjetas DAQ's. Para conseguir una correcta sincronización de los tiempos de cálculo y la lectura así como escritura de señales, se hace uso del paralelismo basado en hilos de la librería *threading*, Hattingh (2020). Las aplicaciones permiten al usuario definir de forma fácil e intuitiva el sistema a emular, habilitando para el caso de la aplicación que genera las señales del sistema maestro, dos canales de salida analógica que permiten enviar de manera independiente la información de dos de los estados de la dinámica. El desarrollo de la aplicación del observador parte de un diseño previo de un observador de estados que tenga la posibilidad de sincronizarse usando hasta dos salidas del maestro, por lo que partiendo del diseño del mismo se habilitan de forma adicional dos canales de entrada analógica, donde cada canal recibe la información asociada a la dinámica de un estado en concreto.

Con el objetivo de emular un sistema real se plantea la necesidad de generar los valores estimados de los estados a una tasa constante, esto implica definir de manera no variable el tamaño del paso en el método numérico a emplear. Se tiene que métodos como el de Dormand-Prince son de paso adaptativo, Ang and Amir Hamzah (2018), por lo que se opta por usar el método de Runge-Kutta de 4to orden (RK4) para la estimación de las salidas del sistema. Dado que se trabaja con operaciones en paralelo, se establece que el ciclo de reloj sea equivalente al tamaño del paso para el método numérico.

Como fase previa a la escritura de señales se agrega un módulo que permita hacer una simulación de la solución numérica del sistema, lo que permite definir parámetros adecuados de paso y de conversión *valor de estado-salida en voltaje*. Luego para el emulador del sistema maestro se definen condiciones iniciales, el tamaño del paso a emplear, cálculo automático de la razón de proporción a la que será generada la señal eléctrica en función del valor del estado del sistema, así como la relación inversa (para ingresarla en un observador), y a través de qué canal será enviado cada estado (limitado a dos estados/canales), ver Figura 2.1. Definidos los parámetros para la generación, se requiere efectuar en el maestro las operaciones siguientes:

- **Configuración inicial**:
  - Definir voltajes mínimos/máximos por canal.
  - Seleccionar estados a enviar (x_i para canal 1, x_j para canal 2).
- **Ciclo de generación**:
  1. Ciclo de reloj de duración h segundos.
  2. Cálculo de los valores de los estados de la dinámica por RK4.
  3. Conversión a voltaje
  4. Escritura DAQ:

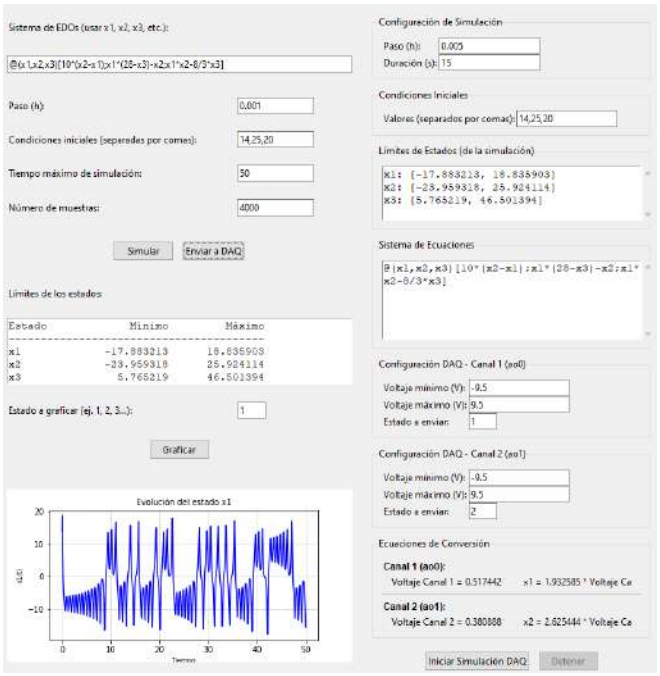

Figura 2.1. Interfaz de la aplicación para el sistema maestro. La simulación (panel izquierdo) permite definir parámetros para la generación de señales (panel derecho).

Para la aplicación del observador se configuran los mismos parámetros que la aplicación previa, ver Figura 2.2. No obstante dado que en los observadores (por ejemplo de alta ganancia, Botero Castro (2009)), la estimaciones pueden encontrarse fuera de los rangos preestablecidos, se ha configurado la aplicación para que la escritura de voltaje tome el valor máximo o mínimo más cercanos al valor estimado, permitiendo que la dinámica del observador no se sature y tenga la posibilidad de sincronizarse. Las operaciones que se efectúan en el observador son:

- **Ciclo de lectura y generación**:
  1. Ciclo de reloj de duración h segundos.
  2. Lectura de entradas analógicas (DAQ): Primer paso en cada iteración. Lee las señales del maestro desde los canales $\text{ai}_0 \rightarrow \text{in}_1$ y $\text{ai}_1 \rightarrow \text{in}_2$ del dispositivo DAQ. Estos valores redefinen el sistema de ecuaciones diferenciales que ingresa el usuario.
  3. Cálculo de los valores de los estados de la dinámica por RK4.
  4. Escritura de salidas analógicas (DAQ) donde se mapea los estados seleccionados a voltajes. Los voltajes se saturan a ± 10 V si exceden el rango del DAQ.

3. DESARROLLO

Se enlistan las características del hardware utilizado para la implementación.

- Computadora - Maestro
  - Modelo: Dell OptiPlex 3020

 https://doi.org/10.58571/CNCA.AMCA.2025.065

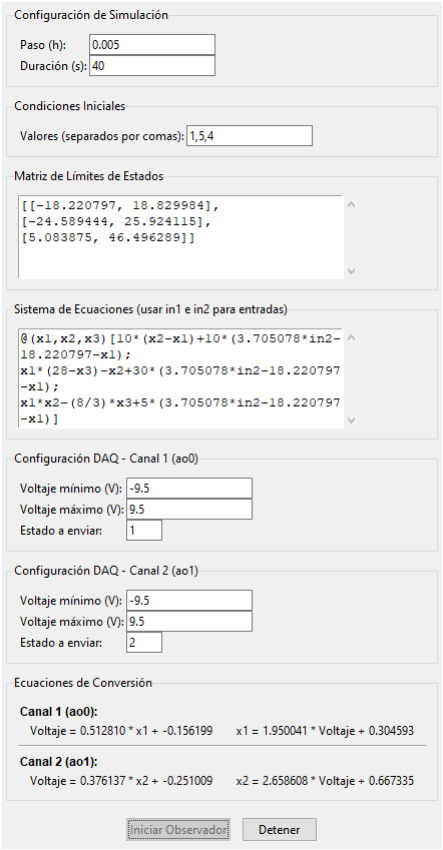

Figura 2.2. Interfaz de la aplicación para el observador. Se permite simular observadores con hasta dos entradas, las cuales deben de ser correctamente escaladas.

- Procesador: Intel Core i5-4590 @ 3.30GHz, 3301 MHz - 4 núcleos
- Memoria RAM: 12 GB DDR4
- Sistema Operativo: Windows 10 Pro

- Computadora - Observador
  - Modelo: Dell Precision T1700
  - Procesador: Intel Xeon E3-1246 @ 3.50GHz, 3501 MHz - 4 núcleos
  - Memoria RAM: 32 GB DDR4
  - Sistema Operativo: Windows 10 Pro
- **Software**
  - Python, utilizado como lenguaje de programación principal. El desarrollo se realizó en el entorno **Spyder**.
    - **Interfaz gráfica:** `tkinter`, `ttk`, `messagebox`, `scrolledtext`
    - **Adquisición de datos:** `nidaqmx`
    - **Procesamiento numérico:** `numpy`
    - **Control de ejecución y concurrencia:** `threading`, `time`
    - **Manejo de texto y expresiones regulares:** `re`, `ast`
  - **NI MAX 2024**, versión Q2, utilizado para la configuración, prueba y monitoreo de las tarjetas de adquisición de datos (DAQ).
- Tarjetas de adquisición de datos de National Instruments
  - DAQ USB-6211 - Maestro
    - Entradas analógicas: 16 entradas analógicas
    - Salidas analógicas: 2 salidas analógicas
    - Velocidad de muestreo: 250 kS/s.
  - DAQ SCB-68A - Observador
    - Tipo: Caja de terminales (Terminal Block) conectada por puerto PCI
    - Conector: Conector de 68 pines para conectar con dispositivos DAQ
    - Entradas Analógicas: 16 entradas analógicas
    - Salidas Analógicas: 2 salidas analógicas

3.1 Emulación de una señal caótica

Se utilizan señales caóticas por sus características de alta sensibilidad a las condiciones iniciales para así validar el esquema maestro-esclavo, emulando el sistema de Lorenz, Arévalo Soler (2023), posteriormente se analiza el espectro generado por el estado x_1.

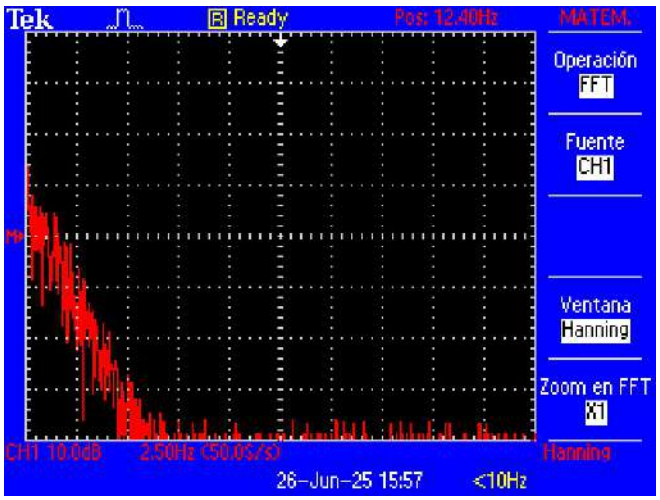

Figura 3.1. Espectro del estado x_1 de la generación del sistema de Lorenz.

El espectro de la señal en la Figura 3.1 muestra una concentración de energía en bajas frecuencias y una atenuación rápida, sin armónicos definidos. Esta distribución es típica de sistemas no periódicos Sauer et al. (1991), lo que sugiere un comportamiento caótico.

Para validar la funcionalidad de las aplicaciones, se implementan dos esquemas de observadores, los cuales permiten verificar su correcta sincronización.

3.2 Observador de orden reducido.

Se toma de la literatura un observador de orden reducido para un modelo de Diabetes Mellitus Tipo 1, dicho observador fue propuesto e implementado en FPGA en Gamboa et al. (2019). El modelo matemático en (1) y los valores de los parámetros son presentados en Lenbury et al. (2001) y describe la interacción entre tres variables de estado: insulina (x_1), glucosa (x_2) y células beta (x_3).

$$\begin{aligned}\dot{x}_1 &= x_3(r_1x_2 - r_2x_1 + c_1),\\ \dot{x}_2 &= \epsilon\left(\frac{r_3}{x_3} - r_4x_1 + c_2\right),\\ \dot{x}_3 &= r_5(x_2 - p)(q - x_3) + r_6x_3(q - x_3) - r_7x_3.\end{aligned} \tag{1}$$

 https://doi.org/10.58571/CNCA.AMCA.2025.065

El observador de orden reducido toma la salida x_3 y está descrito en (2). En la Figura 3.2 se muestra la convergencia del observador para x_1 y x_2.

$$\begin{aligned}\dot{z}_1 &= x_3(r_1 z_2 - r_2 z_1 + c_1),\\ \dot{z}_2 &= \epsilon\left(\frac{r_3}{x_3} - r_4 z_1 + c_2\right),\end{aligned} \tag{2}$$

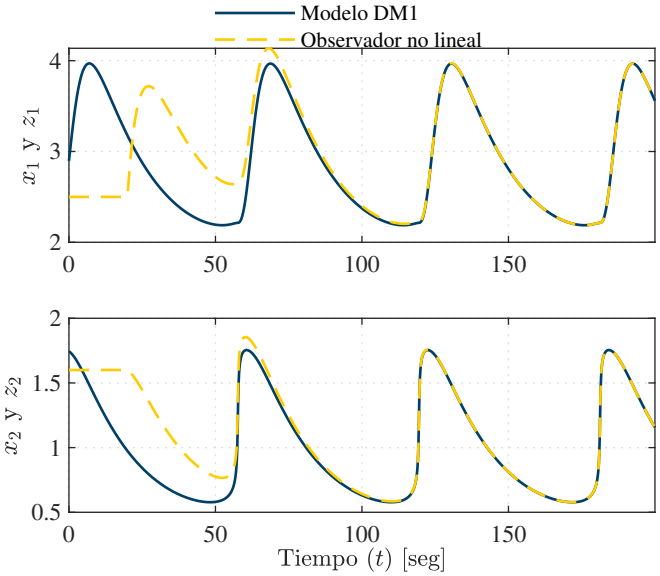

Figura 3.2. Simulación del sistema DMT1 y observador de orden reducido. Se usan las condiciones iniciales $X(0) = [2.897, 1.745, 0.9198]^T$ y $Z(0) = [2.5, 1.6]^T$. El observador se activa con un retraso de 25 segundos.

3.3 Sincronización por observador de Thau

El sistema Lorenz-Stenflo, Stenflo and Shukla (2009), es una extensión del sistema de Lorenz útil para estudiar fenómenos caóticos y estructuras coherentes en sistemas geofísicos y plasmas. Su modelo matemático está dado por (3).

$$\begin{aligned}\dot{x}_1 &= -ax_1 + ax_2 + cx_4,\\ \dot{x}_2 &= rx_1 - x_2 - x_1x_3,\\ \dot{x}_3 &= -bx_3 + x_1x_2,\\ \dot{x}_4 &= -x_1 + ax_4.\end{aligned} \tag{3}$$

Dicho sistema exhibe caos para los valores de los parámetros $a = 2$, $b = 0.7$, $c = 1.5$ y $r = 26$. En Starkov et al. (2012) se presenta un observador de Thau que depende de las salidas x_1 y x_4 dado por el sistema en (4).

$$\begin{aligned}\dot{z}_1 &= az_2 - z_1(a + k_1) + cz_4 + k_1x_1,\\ \dot{z}_2 &= az_1 - x_1(a - r) - z_2 - z_1z_3,\\ \dot{z}_3 &= x_1z_2 - bz_3,\\ \dot{z}_4 &= cz_1 - z_4(a + k_8) + k_8x_4 - x_1(c + 1);\end{aligned} \tag{4}$$

donde los valores de las ganancias son $k_1 = 10$ y $k_8 = 100$. Dado que el observador requiere dos salidas del sistema y la tarjeta DAQ solo dispone de dos canales de salida analógica, la comparación experimental se limita al seguimiento de las variables x_1 y x_4 mediante sus respectivas estimaciones z_1 y z_4.

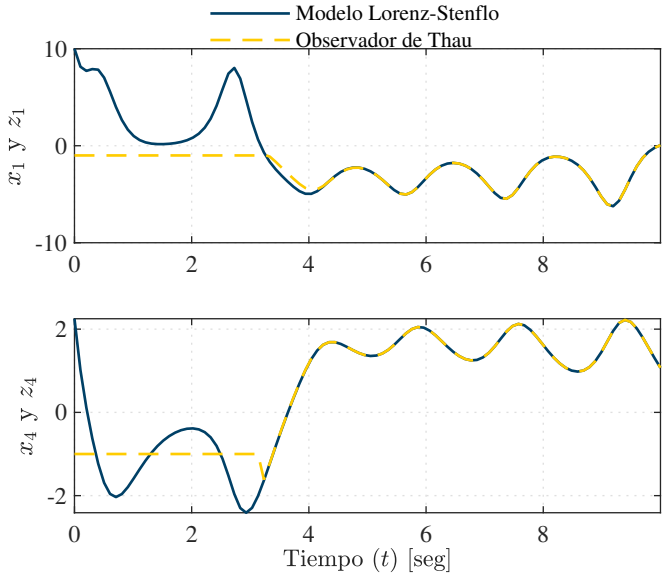

Figura 3.3. Simulación del sistema Lorenz-Stenflo y el observador de Thau. Se emplean las condiciones iniciales $X(0) = [10, -5, 20, 2.25]^T$ y $Z(0) = [-1, -1, 40, -1]^T$. El observador se activa con un retraso de 3.2 segundos

4. RESULTADOS

En este trabajo se muestran los resultados empleando un paso de $h = 0.005$, obteniendo soluciones similares con $h = 0.004$. No obstante una reducción a un valor menor de h implica un menor tiempo disponible para el cálculo de los valores a generar, lo que provoca, con el equipo empleado en este trabajo, la desincronización de los hilos teniendo como consecuencia una generación errónea de las señales eléctricas. Estos resultados muestran que existe un compromiso entre precisión numérica y estabilidad en la ejecución paralela, sugiriendo la necesidad de verificar experimentalmente el valor de h antes de cada implementación.

A continuación se discute el rendimiento de los observadores.

4.1 Observador de orden reducido

Se realiza la emulación del observador con las condiciones de la simulación descritas en la Figura 3.2. Los resultados se muestran en las Figuras 4.1 y 4.2. Se aprecia un comportamiento cercano al esperado en la simulación.

4.2 Sincronización por observador de Thau

Dada la limitación en la cantidad de canales de salida por parte del hardware empleado, este trabajo se limita a presentar la sincronización de los estados x_1 con z_1, y x_4 con z_4 en las Figuras 4.3 y 4.4 respectivamente. Se emplean las mismas condiciones definidas para la simulación (Figura 3.3).

Se aprecia un comportamiento semejante al mostrado en la simulación lo que permite dar confiabilidad en las señales generadas.

 https://doi.org/10.58571/CNCA.AMCA.2025.065

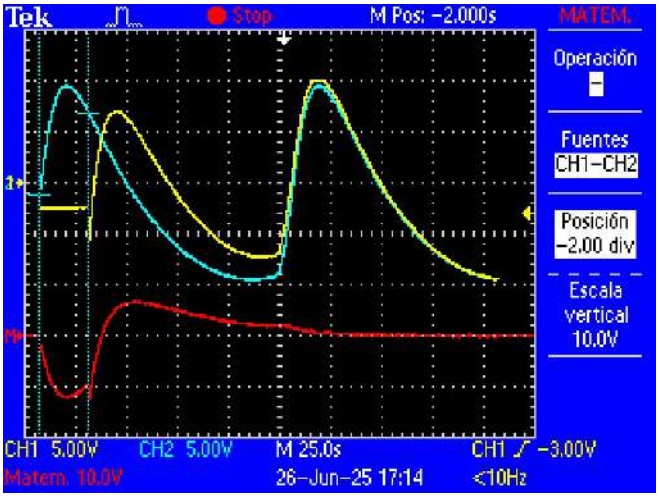

Figura 4.1. Resultado de la implementación del observador para el sistema de Diabetes Mellitus Tipo 1, para la insulina (x_1). El observador se activa 25 segundos después del maestro. La línea roja representa el error entre las señales x_1 y z_1.

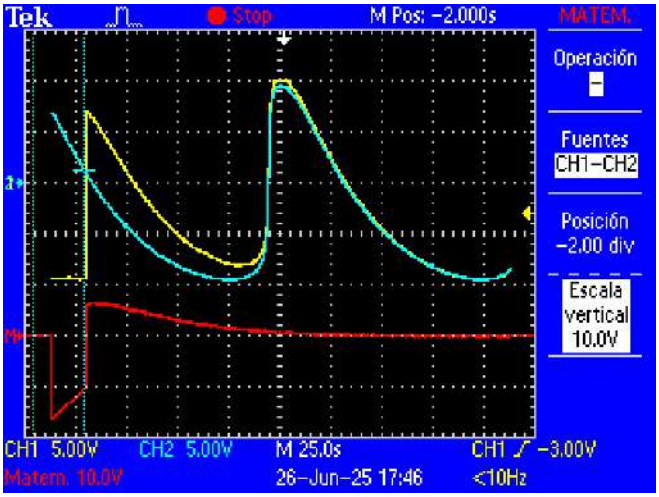

Figura 4.2. Resultado de la implementación del observador para el sistema de Diabetes Mellitus Tipo 1, para la glucosa (x_2). El observador se activa 25 segundos después del maestro. La línea roja representa el error entre las señales x_1 y z_1.

5. CONCLUSIONES

En este trabajo se presentó una metodología práctica para la emulación de sistemas dinámicos descritos por ecuaciones diferenciales ordinarias, mediante el uso de tarjetas DAQ y programación en Python con ejecución en paralelo. Se desarrollaron dos aplicaciones con interfaz gráfica que permiten definir de forma intuitiva los sistemas a emular, así como observar sus dinámicas a través de señales analógicas.

A diferencia de otras implementaciones reportadas en la literatura, Alami et al. (2021), la propuesta aquí presentada combina la flexibilidad de una interfaz gráfica intuitiva con la ejecución paralela en Python para interactuar directamente con hardware de adquisición de

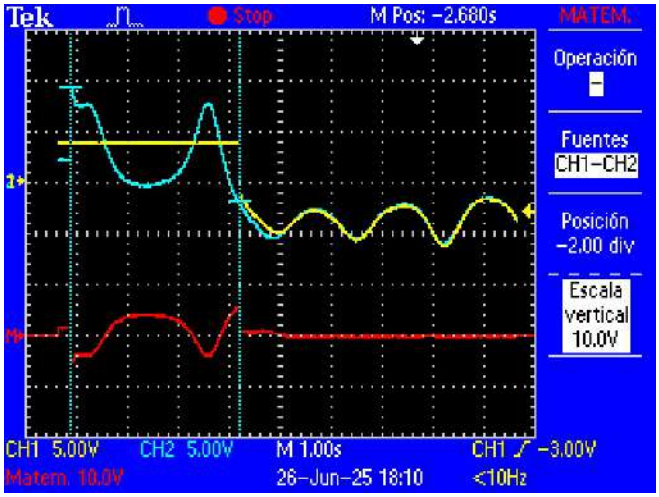

Figura 4.3. Resultado de la implementación del observador para el sistema Lorenz-Stenflo, para el estado x_1 y z_1. El error se muestra en la línea de color rojo.

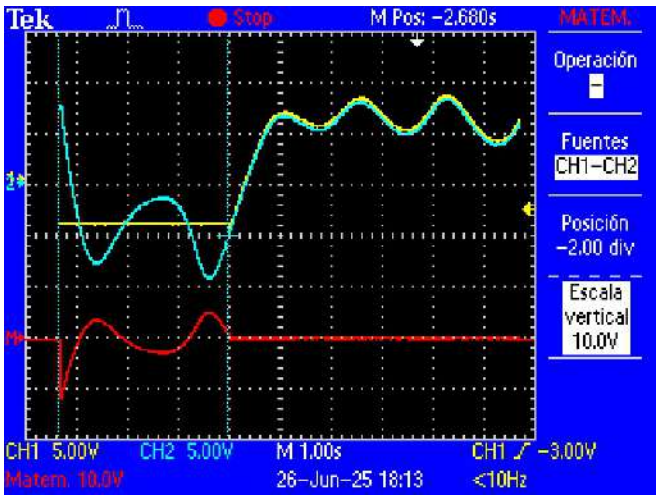

Figura 4.4. Resultado de la implementación del observador para el sistema Lorenz-Stenflo, para el estado x_4 y z_4. El error se muestra en la línea de color rojo.

datos. Este enfoque facilita la validación experimental de esquemas maestro-observador en tiempo real sin necesidad de equipos especializados de alto costo, lo cual constituye una contribución novedosa frente a soluciones previamente orientadas únicamente a simulación o a hardware dedicado como FPGA.

Uno de los aspectos críticos identificados en el desarrollo fue la elección adecuada del valor del paso h en el método numérico. Dicho valor impacta tanto en la precisión de la solución como en la sincronización con el hardware, dado que el tamaño del paso define el ciclo de reloj que regula el cálculo, la escritura y la lectura de señales. Por lo anterior, debe elegirse cuidadosamente para lograr un equilibrio entre precisión numérica y capacidad de procesamiento. Dado que el tiempo de cálculo variará en función del sistema de EDO's no se puede definir un valor óptimo, no obstante se puede concluir en la necesidad de verificar en

 https://doi.org/10.58571/CNCA.AMCA.2025.065

simulación un valor que proporcione resultados óptimos así como mejorar el hardware para mayor capacidad.

Por otra parte, se identificó una limitación importante impuesta por el hardware: la cantidad de canales analógicos disponibles. En particular, la tarjeta DAQ utilizada cuenta con únicamente dos salidas analógicas, lo cual restringe la cantidad de estados que pueden transmitirse. Una consecuencia directa de la restricción en el número de canales analógicos es la imposibilidad de representar simultáneamente todos los estados del sistema. Una alternativa a corto plazo es el uso de técnicas de multiplexación, que permitirían transmitir varias señales a través de un mismo canal. Sin embargo, ello implica una reducción en el ancho de banda efectivo de las señales reconstruidas y puede introducir retardos que afectarían la sincronización eficaz del observador. Este aspecto representa un reto para aplicaciones en las que la fidelidad temporal es crítica, aunque sigue siendo viable en escenarios de validación preliminar. Se propone lo anterior como trabajo a futuro.

En conjunto, esta propuesta representa una recurso para aplicaciones didácticas, validación de modelos y desarrollo de controladores, integrando software libre con hardware especializado. Por último, se comparte el recurso mediante repositorio GitHub para su exploración.

REFERENCIAS

Alami, H.E., Bossoufi, B., Motahhir, S., Alkhammash, E.H., Masud, M., Karim, M., Taoussi, M., Bouderbala, M., Lamnadi, M., and El Mahfoud, M. (2021). Fpga in the loop implementation for observer sliding mode control of dfig-generators for wind turbines. *Electronics*, 11(1), 116.

Ang, T.K. and Amir Hamzah, N.S. (2018). *Solving ordinary differential equations by the Dormand Prince method*. Penerbit UTHM.

Arévalo Soler, P. (2023). Sistema de lorenz i criptosistemes caòtics.

Botero Castro, Héctor Antonio, Á.Z.H.D. (2009). Una revisión de los métodos más frecuentes para la estimación del estado en procesos químicos. *Dyna*, 76(158), 135–146.

Boukezzoula, R., Galichet, S., and Foulloy, L. (2004). Observer-based fuzzy adaptive control for a class of nonlinear systems: Real-time implementation for a robot wrist. *IEEE Transactions on Control Systems Technology*, 12(3), 340–351.

Cardona, J. and Gutiérrez, J.G.H. (2022). Emulación de sistemas como una metodología para la enseñanza de estrategias de control. *Encuentro Internacional de Educación en Ingeniería*.

Duran, J.G.V., Villalobos, R.R., de los Ríos, L.N.C., and Ortega, M.d.J.G. (2018). Diseño de sistemas hipercaóticos para implementación en dispositivos lógicos programables enfocado a aplicaciones de seguridad. *Pistas Educativas*, 39(128).

Gamboa, D., Coria, L.N., Cárdenas Valdez, J.R., Ramírez Villalobos, R., and Valle Trujillo, P.A. (2019). Implementación en hardware de un observador no lineal para un modelo matemático de diabetes mellitus tipo 1 (dm1). *Computación y Sistemas*, 23(4), 1475–1486.

Hattingh, C. (2020). *Using Asyncio in Python: understanding Python's asynchronous programming features*. .º'Reilly Media, Inc.".

Jäger, D. and Gümmer, V. (2023). Pythondaq – a python based measurement data acquisition and processing software. *Journal of Physics: Conference Series*, 2511(1), 012016. doi:10.1088/1742-6596/2511/1/012016. URL https://dx.doi.org/10.1088/1742-6596/2511/1/012016.

Kapitaniak, T., Mohammadi, S.A., Mekhilef, S., Alsaadi, F.E., Hayat, T., and Pham, V.T. (2018). A new chaotic system with stable equilibrium: Entropy analysis, parameter estimation, and circuit design. *Entropy*, 20(9), 670.

Lenbury, Y., Ruktamatakul, S., and Amornsamarnkul, S. (2001). Modeling insulin kinetics: responses to a single oral glucose administration or ambulatory-fed conditions. *Biosystems*, 59(1), 15–25.

Mattson, T.G., Anderson, T.A., and Georgakoudis, G. (2021). Pyomp: Multithreaded parallel programming in python. *Computing in Science & Engineering*, 23(6), 77–80.

Muñoz-Sánchez, J.J., Ramos-León, G., Núñez-Pilligua, W., Cuenca-Álava, L., and Palacios-Castro, E. (2025). Análisis, mediante simulación, del comportamiento eléctrico y mecánico de un generador sincrónico trifásico bajo carga. *Revista Científica FINIBUS-Ingeniería, Industria y Arquitectura*, 8(15), 112–120.

Olmo López, S. (2025). *Diseño e implementación de un sistema para la simulación/emulación de sondas de temperatura y de presión para testear maquina industrial HVAC*. B.S. thesis, Universitat Politècnica de Catalunya.

Pérez-Cruz, J.H., Allende Peña, J.M., Nwachioma, C., Rubio, J.d.J., Pacheco, J., Meda-Campaña, J.A., Ávila-González, D., Guevara Galindo, O., Adrian Romero, I., and Belmonte Jiménez, S.I. (2020). A luenberger-like observer for multistable kapitaniak chaotic system. *Complexity*, 2020(1), 9531431.

Sanguino, J.L.M. (2023). Software basado en python para el control de un horno simulado.

Sauer, T., Yorke, J.A., and Casdagli, M. (1991). Embedology. *Journal of statistical Physics*, 65, 579–616.

Starkov, K.E., Coria, L.N., and Aguilar, L.T. (2012). On synchronization of chaotic systems based on the thau observer design. *Communications in Nonlinear Science and Numerical Simulation*, 17(1), 17–25.

Stenflo, L. and Shukla, P. (2009). Nonlinear acoustic–gravity waves. *Journal of plasma physics*, 75(6), 841–847.

Valle, P.A., Coria, L.N., Plata, C., and Salazar, Y. (2021). Car-t cell therapy for the treatment of all: eradication conditions and in silico experimentation. *Hemato*, 2(3), 441–462.

Weber, S. (2021). Pymodaq: An open-source python-based software for modular data acquisition. *Review of Scientific Instruments*, 92(4).

 https://doi.org/10.58571/CNCA.AMCA.2025.065