Ingeniería Investigación y Tecnología
volumen XXV (número 4), octubre-diciembre 2024 1-13
ISSN 2594-0732 FI-UNAM artículo arbitrado
Información del artículo: Recibido: 30 de diciembre de 2022, aceptado: 4 de septiembre de 2024

https://doi.org/10.22201/fi.25940732e.2024.25.4.029

Ingeniería
Investigación y
Tecnología

Desarrollo y aplicación de una tarjeta embebida para el control de un sistema rueda-bola

Development and application of an embedded system to control a ball and wheel system

Soria-López Alberto
Centro de Investigación y de Estudios Avanzados
del I.P.N.
Correo: soria@cinvestav.mx
https://orcid.org/0000-0002-6310-9527

Ojeda-Misses Manuel Alejandro
Universidad Autónoma del Estado de Hidalgo
Instituto de Ciencias Básicas e Ingeniería
Área Académica de Computación y Electrónica
Correo: manuel_ojeda@uaeh.edu.mx
https://orcid.org/0000-0003-3963-5399

Resumen

Este articulo presenta el desarrollo de una tarjeta embebida para la aplicación de control de un sistema rueda-bola. En este trabajo se describen los elementos de la tarjeta embebida considerando desde la arquitectura, el hardware, el protocolo de comunicación y el software, obteniendo características como: bajo costo, confiable y robusta, además de tener una interfaz USB de alta velocidad y rendimiento en tiempo real. La tarjeta embebida permite la conexión de un motor de corriente directa permitiendo obtener un servomecanismo portátil fácil de interconectar con cualquier computadora y puede ser acoplado a sistemas robóticos abriendo nuevas posibilidades prácticas en aplicaciones control. La flexibilidad del servomecanismo integrado mediante la tarjeta embebida es, en este artículo, para la implementación y desarrollo de un sistema rueda-bola que permite validar mediante experimentos en tiempo real, considerando desde el modelado del sistema hasta la parte experimental.
Descriptores: Sistema embebido, control, instrumentación, sistema lineal, linealización, sistema rueda-bola.

Abstract

In this article he presents the description of the development of an embedded card for the application in the area of control and instrumentation. The platform is considered low-cost, reliable, and robust hardware, has a high-speed USB interface and hard real-time based performance. The platform is easy to interface with any computer and can be taken home by students, opening up many new possibilities for practical work in control courses. The embedded card used in the development of the platform is used for the implementation and development of a wheel-ball system through real-time experiments, considering from the modeling to the experimental results.
Keywords: Embedded system, control, instrumentation, linear system, linearization, system ball and wheel.

https://doi.org/10.22201/fi.25940732e.2024.25.4.029

Introducción

En la vida diaria el hardware y el software juegan un papel importante, debido a que permiten el desarrollo y la implementación de la tecnología que satisfacen las necesidades de los seres humanos. En muchas ocasiones solo se asocia la palabra hardware a una computadora que puede ejecutar diversas aplicaciones mediante el uso de software, sin embargo, la mayoría de la gente hace uso del hardware, denominado como el conjunto de elementos físicos o materiales en aparatos electrónicos. Por ejemplo, un automóvil puede tener varios recursos de hardware como los microprocesadores para controlar algunas funciones, tales como: sincronizar el motor, los frenos, la transmisión, el sistema de seguridad, el aire acondicionado, etcétera. Por lo tanto, un auto tiene integrado recursos de hardware y de software que permiten su funcionamiento mediante alguna tarjeta electrónica con un fin especifico, conocido como sistema embebido.

Sistemas embebidos

Los sistemas embebidos también son llamados empotrados o embarcados. Según Sánchez *et al.* (2015) existen diversas definiciones, entre ellas se encuentra un sistema integrado que se basa en un microprocesador y está construido para controlar algunas funciones (Heath, 2003). Noergaard (2013) menciona que a diferencia de una computadora, un sistema embebido no está diseñado para ser programado por el usuario final. Por otro lado, en Kleidermacher *et al.* (2012) a diferencia de computadora, un sistema embebido es un sistema integrado, es decir, un sistema computacional aplicado. En Abbott (2013) se define como un sistema electrónico que contiene al menos un microprocesador y un software para realizar alguna función integrada en la entidad más grande. Alva & Alcorta (2020) lo definen como un dispositivo que tiene una computadora dentro de él, pero el usuario de este dispositivo no necesariamente sabe que es una computadora. Finalmente, en Pont (2014) lo propone como un sistema electrónico con capacidad de cómputo diseñado para realizar una o varias funciones específicas. Las características principales de un sistema embebido según Alva *et al.* (2020) es que son específicos en hardware y con funcionalidad de software; es un sistema integrado y diseñado para realizar funciones específicas. Asimismo, están integrados por microcontroladores (Apaza, 2010), microprocesadores (Valdivieso & Solís, 2018), circuitos integrados digitales que contienen compuertas lógicas programables llamado FPGA por sus siglas en inglés *field programmable gate array* (Chan & Mourad, 2008), procesadores digitales de señales (Lapsley *et al.*, 1996), tarjetas embebidas de uso comercial, o sistemas embebidos construidos o incluso una combinación de ellos. Sin embargo, a pesar de tener aplicaciones con FPGA's (Hsu *et al.*, 2011; 2013) procesadores digitales de señales (Chih *et al,* 2009; Yung *et al.*, 2013), microcontroladores y microprocesadores (Hongjun & Byung, 2014; Hedjar & Bounkhel, 2014), entre otros. En algunos casos se recurre al desarrollo de sistemas embebidos mediante tarjetas electrónicas de fabricación propia, debido a que son diseñadas con base en requerimientos específicos para realizar prototipados rápidos con el fin de adecuarlas a un fin en común mediante un entorno de desarrollo integrado (IDE). En este caso, un IDE permite programar un microprocesador o un microprocesador para definir la aplicación del sistema embebido e incluso desarrollar una interfaz de usuario, que son aplicados en la electrónica, las comunicaciones, el control automático, la robótica, la mecatrónica, entre otras (Sánchez *et al.*, 2018). La principal ventaja es que el sistema embebido puede rediseñarse, reconfigurarse e incluso readaptarse para otras funciones en comparación con las tarjetas embebidas de algunas marcas como Arduino (Arduino, 2022), Rasberry (Rasberry, 2022), National Instruments (National Instruments, 2022), entre otras.

En este trabajo se consideran las características de los sistemas embebidos para la integración de un servomecanismo con fines educativos para implementar técnicas de control mediante la instrumentación (Bräunl, 2006; Flores *et al.*, 2016). El Control Automático (Control Automático, 2022) tiene como objetivo lograr la operación de procesos mediante la instrumentación según las especificaciones de funcionamiento, ello, con la menor supervisión humana posible a pesar de los errores o las perturbaciones. El área de control está presente en la industria, la robótica y gran variedad de sistemas que resuelven tareas específicas y problemas abiertos (Siegwart & Nourbakhsh, 2004), donde se busca controlar variables de posición, temperatura, presión y velocidad, entre otras, mediante computadoras, sensores, actuadores, instrumentación y sistemas embebidos (Dorf & Bishop, 2001; Bolton, 2001).

Seleccionar un sistema embebido para un objetivo específico no es una tarea sencilla, ya que de ese sistema dependerán las necesidades y requerimientos para la aplicación a desarrollar (Dorf & Bishop, 2001; Ogata, 2010). Por ende, en el desarrollo de una tarjeta embebida de elaboración propia se buscan técnicas de integración rápida. Para esto se recurre al uso de técnicas como hardware-in-the-loop (HIL) y prototipado de control rápido (Flores *et al.*, 2016) que ayudan a recortar el tiempo de diseño y reducir los costos de desarrollo, lo que permite validar al modelo, los algoritmos de control y

software sin comprometer el proceso en sí. De esta manera, las técnicas aplicadas requieren de la instrumentación mediante el uso de indicadores, transmisores, actuadores y transductores, los cuales ayudan a integrar un sistema embebido como el ejemplo presentado a continuación.

Sistema embebido para un servomecanismo con desempeño en tiempo real para el control automático

Hoy en día, los trabajos de investigación y procesos industriales requieren de la instrumentación para llevar a cabo sistemas de control. Por tal motivo, los sistemas embebidos son usados para el desarrollo donde se considera la integración de transductores, actuadores, acondicionamiento de señal, circuitos para adquisición de datos y análisis de datos, etapa de control y de potencia, convertidores analógico-digital, software, entre otros. El sistema embebido presentado es usado para el desarrollo de un servomecanismo en control automático con características de bajo costo (Linux) y con desempeño en tiempo real (González *et al.*, 2019). El servomecanismo es diseñado e implementado con un microcontrolador PIC, con entradas emitidas desde la comunicación USB (bus universal en serie) y las salidas para el control de giro de un servomotor, la modulación de ancho de pulso (PWM) y el amplificador de corriente; además, tiene funcionalidad con programas como Matlab y Scilab (Scilab Enterprises, 2022) con una computadora en tiempo real.

El sistema es integrado y diseñado para realizar funciones específicas de la teoría del control, es decir, pueden desarrollarse diversas estrategias de control mediante el uso de diagramas de bloques, permitiendo tener una interfaz de usuario mediante los programas mencionados, los cuales facilitan la programación, configuración y control. Debido a que la mayoría de las computadoras actualmente tienen un puerto USB, esto permite el uso de dispositivos de adquisición de datos sin la necesidad de insertarlos en una ranura para placa base, es decir, el servomecanismo experimental usa el puerto USB de alta velocidad de 480 Mb/s que permite una frecuencia de muestreo de hasta 0.5 milisegundos. La Figura 1 muestra la arquitectura del sistema con tres elementos: Hardware (la placa del servomecanismo de control), el protocolo de comunicación (la interfaz de hardware para transcripción de datos del puerto paralelo mejorado (Enhanced Parallel Port, EPP) a USB) y el software dado en tiempo real con dos opciones, la primera con Linux mediante RTAI, Scilab/Xcos, controlador y QrtaiLab; y la segunda mediante Windows, QUARC y Matlab-Simulink.

Hardware

El servomecanismo es diseñado para implementar estrategias de control y está integrado mediante un motor de corriente directa (DC), un decodificador incremental óptico, la tarjeta embebida que contiene un microcontrolador con interfaz periférica (PIC), un microcontrolador de comunicación USB de alta velocidad, un aislador magnetorresistivo digital, un amplificador galvánico analógico y un puente H; y una fuente de alimentación. Dentro del PIC es implementado un amplificador de corriente con el puente H usando el lazo de corriente, donde las ganancias pueden ser ajustadas en tiempo real a través QrtaiLab (Holger, 2010) o Simulink (The Math Works, 2012), según sea el caso. En la Figura 2 se muestra la integración del sistema embebido, que en conjunto con los demás componentes integra un servo-

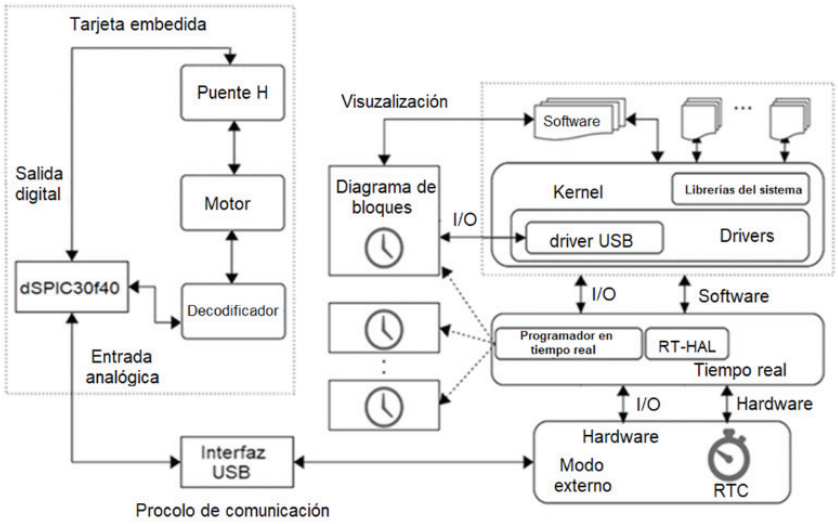

Figura 1. Arquitectura integrada del sistema embebido

https://doi.org/10.22201/fi.25940732e.2024.25.4.029

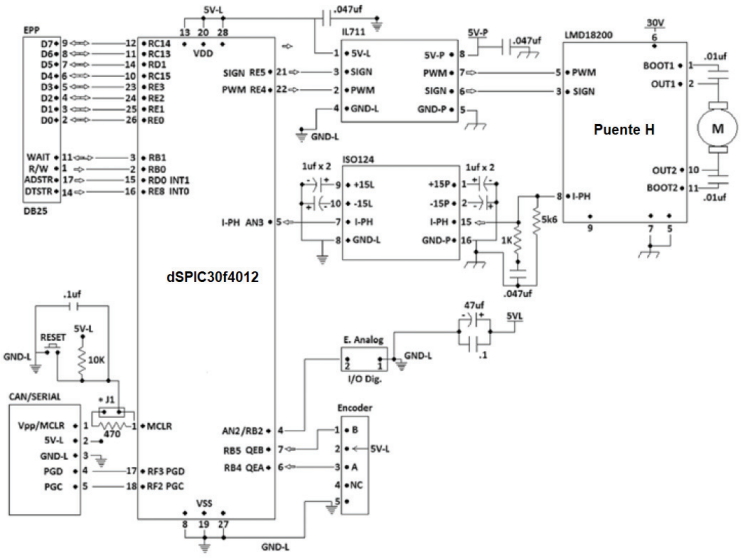

Figura 2. Diagrama esquemático del circuito electrónico del sistema embebido

mecanismo con fines educativos para el aprendizaje de control. El firmware es el programa dentro del dSPIC30f4012 y realiza cinco tareas: Inicializa los periféricos dSPIC30f4012 utilizados para controlar el motor DC, el convertidor AD para medir la corriente en el puente H, el módulo QEI para leer el decodificador óptico incremental y el módulo PWM para controlar el puente H; realiza el lazo de control de corriente en un controlador proporcional integral interno; actualiza la posición del decodificador en el vector de datos; actualiza el valor de salida PWM del vector de datos; y finalmente, transfiere los datos desde/hacia el vector de datos a través de INT1 e INT2 sin interrumpir las rutinas de servicio.

El objetivo principal al seleccionar los componentes del sistema es tener la menor cantidad de componentes posible para obtener así el bajo costo (mediante software libre), además del alto rendimiento y la facilidad de reparación. Se puede observar que todos los circuitos integrados (IC) tienen una configuración doble en línea (Dual In Line, DIP) insertados en bases soldadas que se pueden cambiar con facilidad permitiendo la interconexión necesaria. El dSPIC30f4012 es seleccionado, ya que este PIC tiene módulos específicos para aplicaciones de control de motores, además de una interfaz de codificador en cuadratura (QEI) y una modulación de ancho de pulso (PWM). La frecuencia del PWM es de 20.317 KHz evitando ruidos audibles en el motor y está determinada por el microcontrolador. El convertidor analógico digital (ADC) cuenta con 10 bits y es capaz de medir el voltaje proporcional a la corriente generada en el puente H, que es usada como señal de realimentación para el amplificador de corriente. Los circuitos integrados digitales y analógicos de aislamiento, permiten proteger de la etapa del amplificador de potencia a la etapa de procesamiento digital. La interfaz USB USB2LPT utiliza el chip Cypress CY-68013 EZ-USB de alta velocidad a 480 Mb/s (Haftmann, 2022). El chip Cypress mediante el conector USB-LPT permite utilizar el servomecanismo con una interfaz USB. Por otro lado, el decodificador incremental óptico usado es el modelo US-Digital E-6 2500 que permite el conteo de 10000 pulsos por revolución. Finalmente, el motor corriente directa es de escobillas de doble eje modelo Dynetic System 509051 como se muestra en la Figura 3.

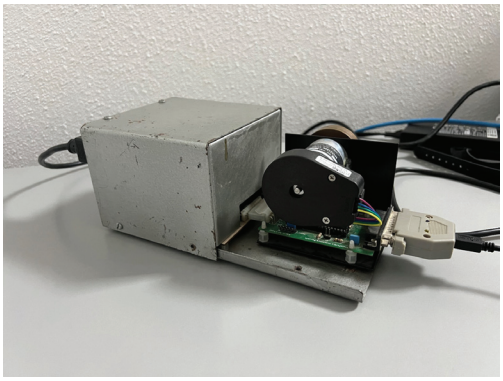

Figura 3. Servomecanismo integrado con la tarjeta embebida

Protocolo de comunicación

Para obtener la interfaz USB se recurre al uso de hardware descrito por la especificación EPP. La especificación EPP permite realizar ciclos de lectura/escritura de direcciones y datos, donde el protocolo de comunicación se realiza por el hardware en lugar del software. La computadora actúa como maestro y el dSPic30f4012 como esclavo. Por lo tanto, un ciclo de lectura transferi-

rá información del dSPIC30f4021 a la computadora. El ciclo de escritura envía la información desde la computadora al dSPIC30f4012. El firmware es el programa dentro del dSPIC30f4012 que realiza las siguientes tareas: Inicializa los periféricos dSPIC30f4012 utilizados para controlar el motor y el convertidor analógico-digital para medir la corriente en el puente H; el módulo QEI lee el decodificador óptico incremental y el módulo PWM impulsa el puente H, que realiza el lazo de corriente y actualiza la posición del decodificador, asimismo el valor de salida del PWM; finalmente, es capaz de transferir los datos desde y hacia el vector de datos a través de diversas rutinas de interrupción desde las terminales INT1 e INT2 del microcontrolador.

La computadora actuará como maestro y el dSPic30f4012 como esclavo. Por lo tanto, un ciclo de lectura transferirá información desde el dSPIC30f4021 a la PC. Un ciclo de escritura transferirá información desde la PC al dSPIC30f4012. La Figura 4 muestra la estructura del intercambio de datos básico donde la computadora realizará otros ciclos de direcciones y se utilizarán para sincronizar el intercambio de datos.

Software

El sistema embebido permite la conexión del servomecanismo con el uso de software libre, gratuito y de código abierto utilizando el sistema operativo Lubuntu 14.04, y RTAI 4.1 con kernel 3.81.6. RTAI, que es** una extensión en tiempo real para Linux. Consiste principalmente en una extensión para el kernel de Linux que introduce una capa de abstracción de hardware y una variedad de servicios para facilitar la programación. XCos es un modelador gráfico y simulador para sistemas dinámicos que permite la posibilidad de compilar dichos modelos en código ejecutable. Los bloques XCos para el servomecanismo se crean utilizando HART Toolbox Versión 3.23 (Holger, 2022) y la interfaz de usuario utiliza QRtaiLab 2.3 (Holger, 2010), basado en QT que es una plataforma de desarrollo para desarrollar interfaces gráficas.

La conexión desde Matlab-Simulink se basa en un núcleo en tiempo real y se requiere el uso de QUARC que permite la producción y creación rápida de prototipos. QUARC puede ser integrado con Simulink.

Asímismo, QUARC tiene una gama de características que permiten el rendimiento en tiempo real aceptando el ajuste de parámetros en línea directamente desde el diagrama en Simulink, también se pueden obtener resultados y gráficas de los experimentos en tiempo real y permite realizar varios experimentos simultáneamente, realiza la generación de código a partir de un solo diagrama en Simulink, entre otras (Quanser, 2022a; 2022b, 2022c); (Quanser Consulting, 2011). La arquitectura requiere los controladores la interfaz USB2LPT (versión 1.7) que es conectada al puerto USB para realizar la comunicación entre el sub sistema del kernel USB utilizando el chip FX2LP alcanzando una velocidad de transferencia máxima de 480 Mb/s (Haftmann, 2022).

La arquitectura de Linux agrupa los controladores de dispositivos en tres categorías: Controladores de bloque, carácter y red. La interfaz USB2LPT (versión 1.7) es un dispositivo de caracteres. Una vez anclado a un puerto USB que realiza la comunicación entre el subsistema del kernel USB y el estándar IEEE 1284–1994 (puerto paralelo) implementado mediante el chip FX2LP (Cypress Semiconductor) alcanzando una velocidad de transferencia máxima de 480 Mb/s.

La Figura 5 muestra el diagrama de bloques del controlador implementado para USB2LPT. Cuando el controlador se carga en la memoria, la función usb2lpt_init() lo registra en el subsistema del kernel USB. La función usb2lpt_probe() se ejecuta cuando el hardware USB2LPT está anclado a un puerto USB del ordenador. Con este se obtiene información sobre interfaces de hardware y puntos finales y se registra el dispositivo. Luego, la información se almacena en una estructura interna que representa el dispositivo. La función usb2lpt_open() abre la interfaz USB usb2lpt_read() y usb2lpt_write() contienen el código para ensamblar URB para enviar, recibir y copiar información hacia y desde

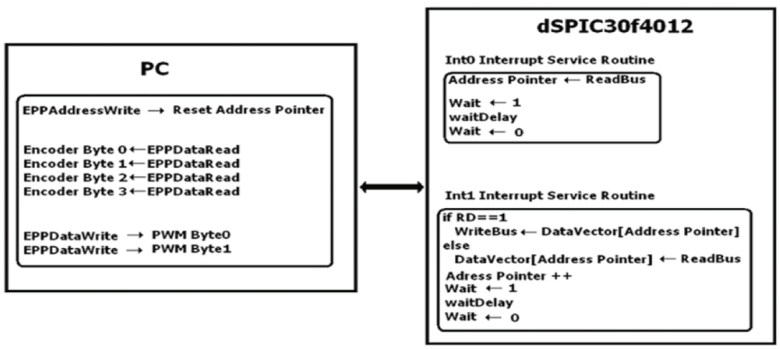

Figura 4. Diagrama de intercambio de datos básico entre la computadora y el microcontrolador dSPIC30f4012

https://doi.org/10.22201/fi.25940732e.2024.25.4.029

el programa de usuario. Estas funciones tienen sus punteros en la estructura file_operatives y forman la interfaz con el usuario. En ese sentido, el usuario ve al controlador como un archivo especial, al que se accede desde una aplicación utilizando primitivas del sistema como abrir(), cerrar(), leer() y escribir(). La función usb2lpt_disconnect() se llama cuando el dispositivo no está anclado al USB. Libera el registro del dispositivo realizado por usb2lpt_probe(). Por último, usb2lpt_cleanup() libera el registro del controlador en el subsistema del kernel USB realizado por usb2lpt_init() cuando el módulo se descarga del sistema.

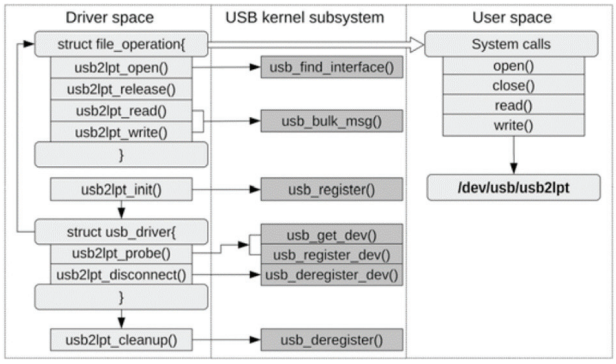

Figura 5. Arquitectura del controlador Linux USB2LPT

El controlador se implementa como un módulo del kernel de Linux. La compilación debe realizarse con la misma versión del compilador utilizada para el kernel RTAI. Incluir una licencia dual BSD/GPL evita que el kernel se contamine y evita advertencias en el registro del sistema. La carga y descarga del módulo se puede realizar con los programas insmod y rmmod. El estado de la carga se verifica con lsmod. Una vez cargado, el archivo especial del controlador sigue las mismas reglas de seguridad que los archivos normales e inicialmente sólo el usuario root ha concedido acceso. El acceso normal a usuarios y grupos se carga utilizando chmod 777 <ruta del archivo>. También se puede ejecutar la aplicación que utiliza el controlador como usuario root.

La interfaz empleada en Scilab para utilizar el controlador desarrollado anteriormente es un bloque de funciones XCos. El bloque representa una capa de abstracción compuesta en su base por la interfaz más compleja de la aplicación controladora en tiempo real para interactuar a bajo nivel con el hardware.

Una función escrita en C permite la gestión de la información transportada desde y hacia el servomecanismo, y una función de interfaz escrita en lenguaje Scilab en la parte superior encargada de lidiar con la apariencia y el comportamiento del bloque durante la simulación. El bloque resultante (Bloque 4 tipo lenguaje C) tiene seis entradas de señal, una entrada de activación y cuatro salidas de señal. Las entradas de señal permiten sintonizar, encender o apagar el bucle de corriente de bajo nivel PI y recibir la señal de referencia actual. La entrada de activación permite sincronizar el bloque con la simulación en tiempo real. Las salidas, por otro lado, corresponden a los conteos del codificador de cuadratura, la velocidad estimada usando un filtro digital de paso alto, la corriente medida y la señal PWM aplicada al motor de CD. El bloque de virtualización se puede simplificar en un super bloque para operar solo con las entradas y salidas de interés, por ejemplo, la señal de entrada de corriente de referencia (tau) y la señal de salida de velocidad (vel) como se muestra en la Figura 6.

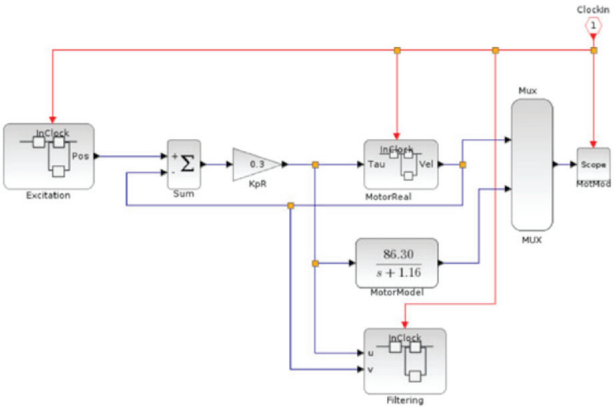

Figura 6. Modelo basado en diagrama de bloques en XCos

Aplicación con la tarjeta embebida para el control del sistema rueda-bola

El servomecanismo ha sido empleado por alumnos de Licenciatura y Maestría de las áreas de Control Automático, Electrónica, Mecatrónica, Industrial, entre otros. Donde la interfaz es a través de los programas Matlab y Scilab mediante los diagramas de bloques que representan una abstracción compuesta en su base por la interfaz de aplicación del controlador en tiempo real más compleja para interactuar a bajo nivel con el hardware; un programa computacional escrito en C que permite la gestión de la información transportada desde y hacia el servomecanismo, y una función de interfaz en Simulink y en Scilab. El bloque resultante tiene seis entradas de señal, una entrada de activación y cuatro salidas de señal. Las entradas de señal permiten que el lazo de corriente de bajo nivel permita sintonizar cualquier controlador desarrollado, además que encienda o apague y reciba la señal de referencia de corriente durante el uso de la plataforma. La entrada de activación permite sincronizar el bloque con la simulación en tiempo real. Las salidas, por otro lado, corresponden a los re-

cuentos del decodificador, la velocidad estimada utilizando un filtro, la corriente medida y la señal PWM del motor. El bloque que integra a estas partes puede ser insertado en un super bloque para operar al sistema con las entradas y salidas de interés, por ejemplo, la señal de entrada de corriente de referencia y la señal de salida de velocidad (Figura 7).

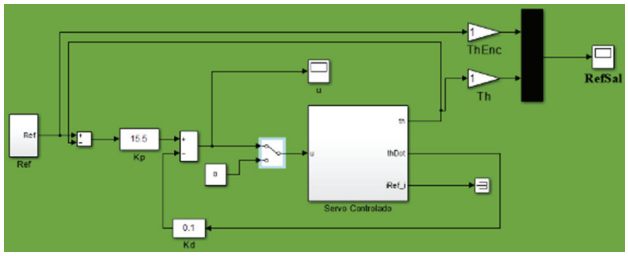

Figura 7. Diagrama de bloques para el control del servomecanismo con el sistema embebido

El servomecanismo ha ayudado a implementar y desarrollar diversos controladores como el proporcional (P), proporcional integral (PI), proporcional derivativo (PD), proporcional integral derivativo (PID), óptimos, robustos, basados en retardos, digitales, entre otros. De la misma forma, ha permitido el desarrollo de algunas estrategias de control y también con péndulos. Actualmente se desarrolla el control de un sistema rueda-bola para mantener en equilibrio la pelota mediante el procesamiento de imágenes en tiempo real como se muestra en la Figura 8.

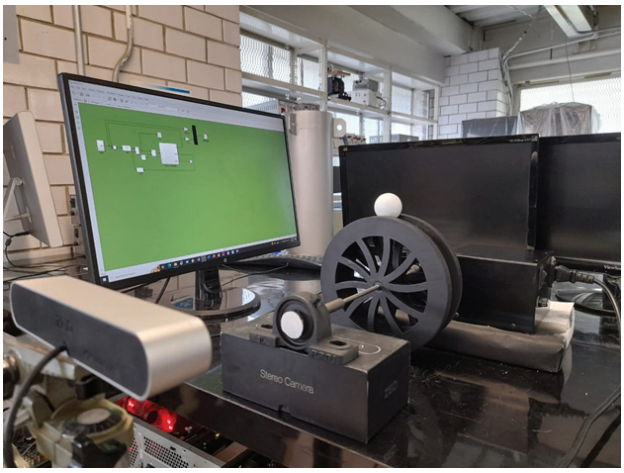

Figura 8. Aplicación con el sistema embebido para el sistema rueda-bola en tiempo real

El modelo del sistema rueda-bola (Ho *et al.*, 2009) es representado en la Figura 9 y se obtiene matemáticamente a partir de la formulación Euler-Lagrange. Asumiendo que la fricción en el sistema es suficiente para evitar que la pelota se deslice, la pelota siempre está en contacto con la rueda, la forma de la ecuación Euler-Lagrange es la siguiente:

$$\frac{d}{dt}\left[\frac{\partial L}{\partial \dot{q}}\right]-\frac{\partial L}{\partial q}=Q \tag{1}$$

Donde:

L = función Lagrangiana
Q = fuerzas generalizadas
q = coordenadas generalizadas

La función Lagrangiana se define como:

$$L=T-V \tag{2}$$

Donde T es la energía cinética y V es la energía potencial.
Para las coordenadas del sistema tenemos a q como:

$$q=\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix} \tag{3}$$

definiendo a θ_1 como el desplazamiento angular entre el eje y, y la línea que pasa por el centro de la bola y la rueda; y θ_2 es definida como desplazamiento angular de la rueda.

Q está dado por τ que se refiere al torque que ejerce la rueda.

$$Q=\begin{bmatrix}0\\ \tau\end{bmatrix} \tag{4}$$

La ecuación que expresa la energía cinética de la bola es la siguiente:

$$T_b=\frac{1}{2}m_b(r_r+r_b)^2\dot{\theta}_1^2+\frac{1}{2}I_b\dot{\theta}_3^2 \tag{5}$$

Donde:

m_b = masa de la bola
r_r = radio de la rueda
r_b = radio de la bola

https://doi.org/10.22201/fi.25940732e.2024.25.4.029

θ_3 = desplazamiento angular del centro de la bola respecto al eje vertical del mismo, y el momento de inercia I_b esta dado como:

$$I_b = \frac{2}{5} m_b r_b^2 \tag{6}$$

Mientras que la energía cinética de la rueda se representa como:

$$T_r = \frac{1}{2} I_r \dot{\theta}_2^2 \tag{7}$$

donde I_r es el momento de inercia de la rueda. Por lo tanto, obteniendo la sumatoria de las energías cinéticas se tiene:

$$T = T_r + T_b \tag{8}$$

$$= \frac{1}{2} I_r \dot{\theta}_2^2 + \frac{1}{2} m_b (r_r + r_b)^2 \dot{\theta}_1^2 + \frac{1}{2}\left(\frac{2}{5} m_b r_b^2\right)\dot{\theta}_3^2 \tag{9}$$

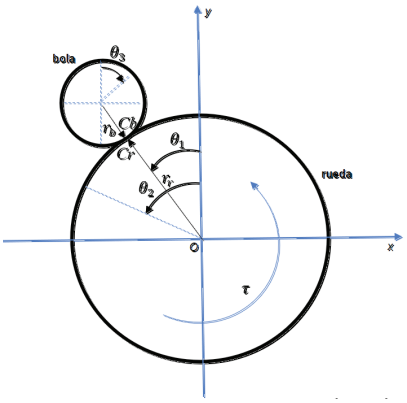

Figura 9. Representación gráfica en el plano del sistema rueda-bola

Por otro lado, debido a que θ_3 no es medible y como se está analizando respecto al contacto entre los puntos *Cb* y *Cr* se obtiene la siguiente condición de cómo se comporta el movimiento al girar la bola:

$$r_r \dot{\theta}_2 - (r_r + r_b)\dot{\theta}_1 = r_b \dot{\theta}_3 \tag{10}$$

sustituyendo θ_3 en la ecuación de la energía cinética tenemos como resultado lo siguiente:

$$T = \frac{1}{2} I_r \dot{\theta}_2^2 + \frac{1}{2} m_b (r_r + r_b)^2 \dot{\theta}_1^2 + \frac{1}{5} m_b (r_r \dot{\theta}_2 - r_r \dot{\theta}_1 - r_b \dot{\theta}_1)^2 \tag{11}$$

Así la energía potencial V es determinada como:

$$V = m_b g (r_r + r_b) \cos\theta_1 \tag{12}$$

Por lo que, el Lagrangiano es determinado como:

$$L = \frac{1}{2} I_r \dot{\theta}_2^2 + \frac{1}{2} m_b (r_r + r_b)^2 \dot{\theta}_1^2 + \frac{1}{5} m_b (r_r \dot{\theta}_2 - r_r \dot{\theta}_1 - r_b \dot{\theta}_1)^2$$
$$- m_b g (r_r + r_b) \cos\theta_1 \tag{13}$$

Obteniendo la derivada parcial respecto a θ_1 el Lagrangiano se tiene:

$$\frac{\partial L}{\partial \theta_1} = m_b g (r_r + r_b) \sin\theta_1 \tag{14}$$

$$\frac{\partial L}{\partial \dot{\theta}_1} = \left(\frac{7}{5} r_r^2 m_b + \frac{14}{5} r_r r_b m_b + \frac{7}{5} r_b^2 m_b\right)\dot{\theta}_1 + \left(-\frac{2}{5} r_r^2 m_b - \frac{1}{5} r_r r_b m_b\right)\dot{\theta}_2 \tag{15}$$

Expandiendo la ecuación 13 se tiene:

$$\left(\frac{7}{5} r_r + r_b m_b + \frac{7}{5} r_b^2 m_b\right)$$

$$L = \frac{1}{2} I_r \dot{\theta}_2^2 + \frac{1}{2} m_b r_r \dot{\theta}_1^2 + \frac{1}{2} m_b r_r r_b \dot{\theta}_1^2 + \frac{1}{2} m_b r_b^2 \dot{\theta}_1^2$$

$$+ \frac{1}{5} m_b (r_r^2 \dot{\theta}_2 - r_r^2 \dot{\theta}_1 \dot{\theta}_2 - r_r \dot{\theta}_1 r_b \dot{\theta}_2 - r_r^2 \dot{\theta}_1 \dot{\theta}_2 + r_r^2 \dot{\theta}_1^2 + r_r r_b \dot{\theta}_1^2 - r_r \dot{\theta}_1 r_b \dot{\theta}_2$$

$$+ r_r r_b \dot{\theta}_1^2 + r_b^2 \dot{\theta}_1^2) - m_b g (r_r + r_b) \cos\dot{\theta}_1$$

Y obteniendo la derivada:

$$\frac{\partial L}{\partial \dot{\theta}_1} = r_r^2 m_b \dot{\theta}_1 + \frac{2}{5} r_r^2 m_b \dot{\theta}_1 + 2 r_r r_b m_b \dot{\theta}_1 + \frac{2}{5} r_r r_b m_b \dot{\theta}_1 + \frac{2}{5} r_r r_b m_b \dot{\theta}_1$$

$$+ \frac{2}{5} r_b^2 m_b \dot{\theta}_1 + r_b^2 m_b \dot{\theta}_1 - \frac{1}{5} r_r^2 m_b \dot{\theta}_2 - \frac{1}{5} r_r r_b m_b \dot{\theta}_2 - \frac{1}{5} r_r^2 m_b \dot{\theta}_2$$

Agrupando términos se tiene:

$$\frac{\partial L}{\partial \dot{\theta}_1} = \frac{7}{5} r_r^2 \dot{\theta}_1 + \frac{14}{5} r_r r_b \dot{\theta}_1 + \frac{7}{5} r_b^2 \dot{\theta}_1 - \frac{2}{5} r_r^2 m_b \dot{\theta}_2 - \frac{1}{5} r_r r_b m_b \dot{\theta}_2$$

Así queda demostrada la ecuación 13, ahora obteniendo la derivada respecto al tiempo se tiene:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right)=\left(\frac{7}{5}r_r^2m_b+\frac{14}{5}r_r r_b m_b+\frac{7}{5}r_b^2m_b\right)\ddot{\theta}_1+\left(-\frac{2}{5}r_r^2m_b-\frac{2}{5}r_r r_b m_b\right)\ddot{\theta}_2 \tag{16}$$

Aplicando ahora las derivadas parciales respecto a θ_2 se tiene:

$$\frac{\partial L}{\partial \theta_2}=0 \tag{17}$$

De forma análoga como se obtuvo la ecuación 15 y 16, se obtiene la ecuación 18 y 19, presentadas a continuación:

$$\frac{\partial L}{\partial \dot{\theta}_2}=\left(-\frac{2}{5}r_r^2m_b-\frac{2}{5}r_r r_b m_b\right)\dot{\theta}_1+\left(I_r+\frac{2}{5}r_r^2m_b\right)\dot{\theta}_2 \tag{18}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_2}\right)=\left(-\frac{2}{5}r_r^2m_b-\frac{2}{5}r_r r_b m_b\right)\ddot{\theta}_1+\left(I_r+\frac{2}{5}r_r^2m_b\right)\ddot{\theta}_2 \tag{19}$$

Finalmente, las funciones dinámicas son las siguientes:

$$7(r_b+r_r)\,\ddot{\theta}_1-2r_r\,\ddot{\theta}_2-5g\sin\theta_1=0 \tag{20}$$

$$\left(-\frac{2m_b}{5}\right)(r_r^2+r_b r_r)\ddot{\theta}_1+\left(I_r+\frac{2}{5}r_r^2m_b\right)\ddot{\theta}_2=\tau \tag{21}$$

Definiendo para el sistema rueda-bola los estados:

$$x_1=\theta_1;\ x_2=\dot{\theta}_1;\ x_3=\theta_2;\ x_4=\dot{\theta}_2 \tag{22}$$

$$\dot{x}_1=x_2;\ \dot{x}_2=f_2;\ \dot{x}_3=x_4;\ \dot{x}_4=f_4 \tag{23}$$

De la ecuación 20 despejamos $\ddot{\theta}_2$:

$$\ddot{\theta}_2=\frac{7(r_b+r_r)\ddot{\theta}_1+5g\sin\theta_1}{2r_r} \tag{24}$$

De la ecuación 21 despejamos $\ddot{\theta}_1$:

$$\ddot{\theta}_1=\frac{5(\tau-(I_r+\frac{2}{5}r_r^2m_b)\ddot{\theta}_2)}{2m_b(r_r^2+r_b r_r)} \tag{25}$$

Sin embargo, el sistema dado por las ecuaciones (24) y (25) es no lineal, por lo tanto, linealizando el sistema (Nise, 2004) y asumiendo que θ_1 es pequeña, se sustituye la ecuación 24 en la ecuación 21 y se obtiene:

$$\left(-\frac{2m_b}{5}\right)(r_r^2+r_b r_r)\ddot{\theta}_1+\left(I_r+\frac{2}{5}r_r^2m_b\right)\left(\frac{7(r_b+r_r)\ddot{\theta}_1-5g\theta_1}{2r_r}\right)= \tag{26}$$

$$\dot{x}_2=\ddot{\theta}_1(\theta_1)=f_2(x_1)=\frac{10r_r\tau+(25l_r g+10gm_b r_r^2)\theta_1}{(r_b+r_r)(35l_r+m_b r_r^2)} \tag{27}$$

Similarmente, se reemplaza la ecuación 25 en la ecuación 20, también se asume que θ_1 es pequeña, se obtiene:

$$7(r_b+r_r)\left(\frac{5\left(\tau-\left(I_r+\frac{2}{5}r_r^2m_b\right)\ddot{\theta}_2\right)}{2m_b(r_r^2+r_b r_r)}\right)-2r_r\ddot{\theta}_2=5g\theta_1 \tag{28}$$

$$\dot{x}_4=\ddot{\theta}_2(\theta_1)=f_4(x_1)=\frac{7\tau+2gm_b r_r\,\theta_1}{7l_r+2m_b r_r^2} \tag{29}$$

Así es posible definir la matriz A para el sistema rueda-bola:

$$\boldsymbol{A}=\begin{pmatrix}0 & 1 & 0 & 0\\ \frac{(25I_r g+10gm_b r_r^2}{(r_b+r_r)(35I_r+m_b r_r^2)} & 0 & 0 & 0\\ 0 & 0 & 0 & 1\\ \frac{2gm_b r_r}{7I_r+2m_b r_r^2} & 0 & 0 & 0\end{pmatrix} \tag{30}$$

De forma análoga es definida la matriz de entrada B:

$$\boldsymbol{B}=\begin{pmatrix}0\\ \frac{10r_r}{(r_b+r_r)(35I_r+m_b r_r^2)}\\ 0\\ \frac{7}{7I_r+2m_b r_r^2}\end{pmatrix} \tag{31}$$

Para la matriz de salida C, se define de la siguiente manera:

https://doi.org/10.22201/fi.25940732e.2024.25.4.029

$$C = (1 \quad 0 \quad 0 \quad 0) \tag{32}$$

Se define de esta manera debido a que el único punto de interés para el control del sistema es el punto de contacto entre la bola y la rueda. Obteniendo finalmente el siguiente sistema en variables de estados:

$$\dot{\bar{x}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \frac{(25I_r g + 10 g m_b r_r^2)}{(r_b + r_r)(35I_r + m_b r_r^2)} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{2 g m_b r_r}{7I_r + 2 m_b r_r^2} & 0 & 0 & 0 \end{pmatrix} \tag{33}$$

$$\bar{x} + \begin{pmatrix} 0 \\ \frac{10 r_r}{(r_b + r_r)(35I_r + m_b r_r^2)} \\ 0 \\ \frac{7}{7I_r + 2 m_b r_r^2} \end{pmatrix} u$$

$$\bar{y} = (1 \quad 0 \quad 0 \quad 0)\bar{x} + [0]u \tag{34}$$

El controlador implementado es sintonizado mediante el método de Ackerman considerando los siguientes parámetros del sistema con base en la construcción física donde $m_b = 0.03\ kg$, $r_r = 0.3\ m$, $r_b = 0.03\ m$, $m_b = 0.285\ kg$ y $l_b = 0.256 \times 10^{-3}\ m^3$, aplicando el método de Ackerman en Matlab que permite asignar directamente los polos en $s_1 = -1$, $s_2 = -1$, $s_3 = -5$ y $s_4 = -10$, se obtiene el siguiente vector ganancias K = [7.4669 0.8192 0 0.0001] para el control de la bola sobre la rueda a nivel experimental.

Cabe mencionar que se emplea una cámara ZED como sensor de realimentación para el sistema rueda bola, donde por medio del procesamiento de imágenes es detectada en todo tiempo la bola. Mediante la segmentación en blanco y negro, además de estimar el área de la bola especifica de 3 cm de radio y estimando un área con base en el centroide de la bola, como se muestra en la Figura 10. Ahí se observa la bola en azul claro cuando es detectada, y el circulo situado en la chumacera sirve como marco de referencia para la medición del ángulo de la bola sobre la rueda. Finalmente, todo lo detectado por la cámara que no cumpla con el área o el perímetro determinado para la bola es marcado en color verde.

Figura 10. Adquisición de datos mediante el procesamiento de imágenes de la bola mediante la cámara ZED en tiempo real

Finalmente, en la Figura 11 se presenta la señal de salida del sistema rueda-bola que simboliza la posición angular de la bola para la asignación de polos mediante el método de Ackerman (negro), por otro lado, se muestra la señal de respuesta para el control de la bola mediante un controlador difuso. Como puede observarse el control lleva a cabo la tarea de mantener en equilibrio a la bola sobre la rueda, sin embargo, el controlador difuso muestra una respuesta más rápida.

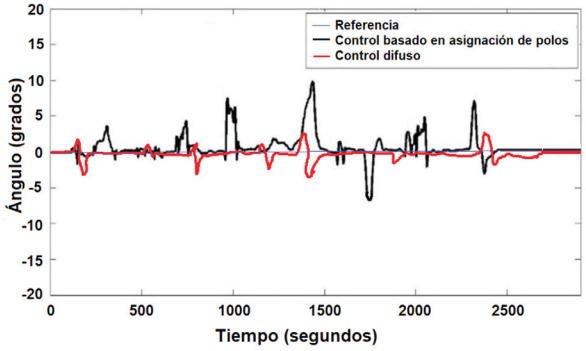

Figura 11. Señal de la posición angular de la bola sobre la rueda en tiempo real

Por otro lado, en las Figuras 12 y 13 se muestran las señales de error de ángulo y la señal de control del motor que lleva a cabo con éxito el sistema rueda-bola para el controlador basado en asignación de polos (negro) y el controlador difuso (rojo). De acuerdo con los resultados antes mostrados, el desempeño del servomecanismo desarrollado mediante la tarjeta embebida se considera que tiene un razonable rendimiento para propósitos de aplicaciones de control e instrumentación aplicada para la enseñanza y se puede obtener la misma versatilidad en cuanto a la implementación de ambos algoritmos de control.

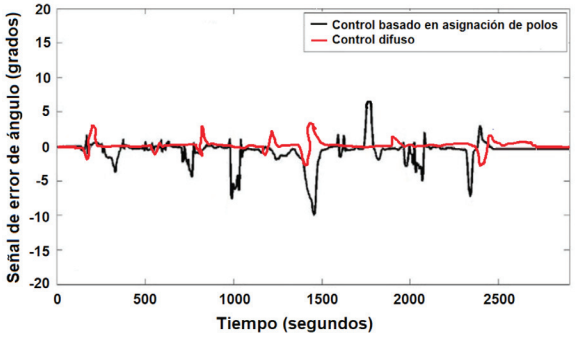

Figura 12. Señal de error de posición de la bola sobre la rueda en tiempo real

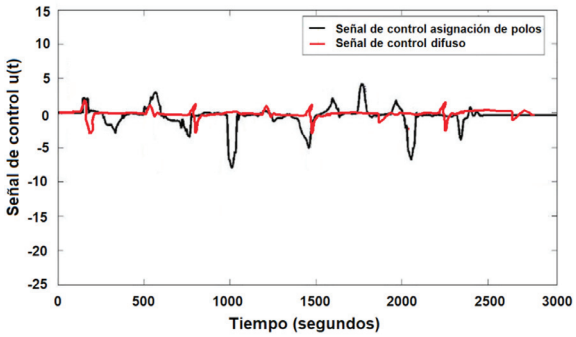

Figura 13. Señal de control del sistema rueda-bola en tiempo real

De acuerdo con los resultados, ambos controladores cumplen el objetivo de llevar a cabo el control de equilibrio de la bola mediante el servomecanismo que es integrado por la tarjeta embebida desarrollada. Respecto al tema de control, el controlador difuso (o de lógica difusa) y el controlador basado en asignación de polos tienen diferentes enfoques y características que los hacen más adecuados para distintas aplicaciones.

Algunas de las ventajas del controlador difuso frente a uno basado en asignación de polos se enfocan en el tema del manejo de la incertidumbre, es decir, el controlador difuso es especialmente útil en sistemas con incertidumbre, ya que no requiere un modelo matemático preciso. La lógica difusa puede manejar entradas ambiguas o imprecisas, lo que lo hace ideal para sistemas donde es difícil obtener un modelo exacto. En cuanto al control por asignación de polos, este controlador depende de un modelo matemático preciso y se basa en la ubicación de los polos del sistema para lograr el desempeño deseado. Esto puede ser limitante en sistemas con incertidumbre significativa.

Respecto al ámbito de la simplicidad y flexibilidad en el diseño, el controlador difuso ofrece una mayor flexibilidad, ya que el diseño de las reglas difusas puede basarse en el conocimiento experto o en la experiencia práctica sin necesidad de un modelo matemático complejo. Esto simplifica el diseño del controlador en sistemas complejos. En contraste, el controlador basado en asignación de polos requiere un conocimiento detallado del modelo del sistema, lo que puede complicar el diseño en sistemas complejos o en aquellos donde los parámetros pueden variar.

En cuanto a adaptabilidad el controlador difuso puede adaptarse mejor a cambios en el sistema o en su entorno, ya que las reglas difusas pueden modificarse o ajustarse más fácilmente. Es más adecuado para aplicaciones donde el sistema varía con el tiempo o donde las condiciones operativas no son constantes. En cambio, el controlador basado en asignación de polos, una vez diseñado, el controlador basado en asignación de polos no es tan flexible para adaptarse a cambios en el sistema sin rediseñar el controlador.

Finalmente, con base en el tema de robustez, el controlador difuso tiende a ser más robusto frente a variaciones y perturbaciones no modeladas debido a su capacidad para manejar la incertidumbre y trabajar con información parcial o inexacta. Por su parte, el controlador basado en asignación de polos puede ser menos robusto si el modelo del sistema no es exacto o si hay perturbaciones significativas, lo que podría requerir ajustes finos o un diseño más complejo para mantener la estabilidad. En resumen, la elección entre un controlador difuso y uno basado en asignación de polos dependerá del tipo de sistema que se desea controlar y de la naturaleza de la información disponible sobre el sistema.

Conclusiones

En este trabajo se presentó el desarrollo de un sistema embebido aplicado para el desarrollo de un servomecanismo de prototipaje rápido para experimentos de control del sistema rueda-bola mediante la instrumentación, la programación específica y el uso de Matlab. En el caso de software libre se reducen los costos asociados a las licencias. El uso de la tarjeta embebida permite el uso de QUARC usando la computadora como interfaz sin necesidad de una computadora externa integrada, al tiempo que garantiza un rendimiento en tiempo real, y la interfaz de programación gráfica de alto nivel basada en Simulink.

Ante los resultados obtenidos se puede decir que la tarjeta de adquision de datos permite lograr el objetivo de control del sistema rueda-bola mediante la arquitectura diseñada, el hardware, el protocolo de comunicación y el software presentado, obteniendo resultados

fiables a nivel experimental y en aspectos como bajo costo, confiable y robusto, además de poder conectarla mediante cualquier puerto USB obteniendo resultados en tiempo real.

Puede concluirse que la implementación y el uso de los sistemas embebidos de manera libre tienen la ventaja de ser diseñados e implementados con hardware y software específico, asimismo con funciones específicas. Está integrado por un microcontrolador, es de bajo costo, adaptable a otras funciones y aplicable para el desarrollo de prácticas y de experimentos en cursos de Control, Robótica, Mecatrónica, entre otros.

Finalmente, el servomecanismo permite validar y desarrollar prototipos aplicados al control, la instrumentación e incluso a la robótica, que permiten aplicar experimentos en tiempo real. Actualmente, se trabaja en el desarrollo de una tarjeta embebida con interfaz USB 3.0 y se implementan otros sistemas robóticos debido a la fácil adaptabilidad, flexibilidad y a que la tarjeta embebida está abierta para poder implementar otras aplicaciones.

Referencias

Abbott, D. (2013). *Linux for embedded and real-time applications*. 3rd ed., MIT, USA: Newnes.

Alva, J., & Alcorta, N. (2020). *Sistemas embebidos, guía metodológica para su desarrollo*. Perú: Fondo Editorial de la Universidad Privada Antenor Orrego.

Apaza-Condori, D. (2010). *Microcontroladores PIC. Fundamentos y aplicaciones. Un enfoque didáctico*. Arequipa: Universidad Autónoma San Francisco.

Arduino, (2022). Arduino Hardware. Recuperado en noviembre 2022 de https://www.arduino.cc/en/hardware

Bolton, W. (2001). *Ingeniería de Control.* 2ª ed. México: Alfaomega,

Bräunl, T. (2006). *Embedded robotics: Mobile robot design and applications with embedded systems*. New York: Springer-Verlag.

Canudas de Wit, C., Siciliano, B., & Bastin, G. (1996). *Theory of robot control.* N. Y.: Springer.

Chan, P., & Mourad, S. (2008). *Digital design using field programmable gate arrays*. USA: Pearson.

Chih-Yang, C., Tzuu-Hseng, S., Ying-Chieh, Y., & Cha-Cheng, C. (2009). Design and implementation of an adaptive sliding-mode dynamic controller for wheeled mobile robots. *Mechatronics,* 19(2), 156-166. http://dx.doi.org/10.1016/j.mechatronics.2008.09.004

Control Automático. (2022). Departamento de control automático, CINVESTAV. Recuperado en noviembre 2022 de https://www.ctrl.cinvestav.mx/

Dorf, R. C., & Bishop, R. H. (2001). *Modern control systems*. 9th ed. Upper Saddle River, NJ: Prentice Hall.

Flores, A., Sabin, D., Villoslada, Á., Blanco, D., & Moreno, L. (2016). Sistema avanzado de protipado rápido para control en la educación en ingeniería para grupos multidisciplinares. *Rev. Iberoamericana de Aut. e Inf. Ind.*, 13(3), 350-362. https://doi.org/10.1016/j.riai.2016.05.004

González-Vargas, A. M., Serna-Ramírez, J. M., Fory-Aguirre, C., Ojeda-Misses, A. *et al.* (2019). A low cost, free-software platform with hard real-time performance for control engineering education. *Comput Appl Eng Educ.*, 27:406-418. https://doi.org/10.1002/cae.22084

Haftmann, H. (2022). Converter From USB To Parallel 1.6. Recuperado en noviembre 2022. https://www-user.tu-chemnitz.de/~heha/basteln/PC/USB2LPT/ul-16.en.htm

Hedjar, R., & Bounkhel, M. (2014). Real-time obstacle avoidance for a swarm of autonomous mobile robots. *International Journal of Advanced Robotic Systems,* 11(67), 1-12. http://dx.doi.org/10.5772/58478

Heath, S. (2003). *Embedded systems design*. 2nd ed. Massachusetts, USA: Newnes.

Ho, M. T., Tu, Y. W., & Lin, H. S. (2009). Controlling a ball and wheel system using full-state-feedback linearization. Control Systems Magazine, IEEE, 29(5), 93-101. https://doi.org/10.1109/MCS.2009.934085

Holger, N. (2010). QRtaiLab. Recuperado en noviembre 2022 de https://qrtailab.sourceforge.net/

Holger, N. (2022). Hart Toolbox. Recuperado en noviembre 2022 de http://hart.sourceforge.net/

Hongjun, K., & Byung-Kook, K. (2014). Minimum-energy trajectory generation for cornering with a fixed heading for three-wheeled omnidirectional mobile robots. *Journal of Intelligent & Robotic Systems,* 75(2), 205-221. https://doi.org/10.1007/s10846-013-9855-1

Hsu-Chih, H. (2013). Intelligent motion control for four-wheeled holonomic mobile robots using FPGA-based artificial immune system algorithm. *Advances in Mechanical Engineering,* 5, 589510-589510. http://dx.doi.org/10.1155/2013/589510

Hsu-Chih, H., Ching-Chih, T., & Shui-Chun, L. (2011). Adaptive polar-space motion control for embedded omnidirectional mobile robots with parameter variations and uncertainties. *Journal of Intelligent & Robotic Systems,* 62(1), 81-102. https://doi.org/10.1007/s10846-010-9438-3

Kleidermacher, D., & Kleidermacher, M. (2012). *Embedded systems security: practical methods for safe and secure software and systems development*. Massachusetts, USA: Newnes.

Lapsley, P., Bier, J., Shoham, A., & Lee, E. (1996). *DSP processor fundamentals: architectures and features, Berkeley.* California: Berkeley Design Technology, Inc.

National Instruments. (2022). National Instruments Hardware. Recuperado en noviembre 2022 de https://www.ni.com/es-mx/shop.html#pinned-nav-section2

Nise, N. (2004). *Control Systems Engineering*. 4a ed. John Wiley & Sons, Inc.

Noergaard, T., (2013). *Embedded systems architecture: a comprehensive guide for engineers and programmers*. 2nd ed. Massachusetts, USA: Newnes.

Ogata, K. (2010). *Ingeniería de control moderna*. 5a ed. Madrid: Pearson.

Pont, J. M. (2014). *Patterns for time-triggered embedded systems.* U.K.: Pearson Education.

Quanser. (2022a). Quarc real-time control, rapid prototyping software for matlab/simulink. Recuperado en noviembre 2022 de https://docs.quanser.com/quarc/documentation/index.html

Quanser. (2022b) Q2-USB Data acquisition device. Recuperado en noviembre 2022 de https://www.quanser.com/products/q2-usb-data-acquisition-device/

Quanser. (2022c). Qnet 2.0 Dc motor board for Ni Elvis. Recuperado en noviembre 2022 de www.quanser.com/wp-content/uploads/2017/04/QNET-2.0-DC-Motor-Datasheet-v1.0.pdf

Quanser Consulting, (2011). QuaRC/versión 2.3.603.

Rasberry. (2022). Rasberry productos. Recuperado en noviembre 2022 de https://www.raspberrypi.com/products/

Sánchez, C. M., *et al.* (2018). An embedded hardware for implementation of a tracking control in WMRs. *IEEE Latin America Transactions*, 16(7), 1835-1842.

Sánchez, C., Silva, R., Cruz, M., Sosa, C., & Hernández, V. (2015). A review of embedded systems used in WMR for the trajectory tracking task. International Conference on Mechatronics, Electronics and Automotive Engineering, pp. 145-150.

Scilab Enterprises, (2022). Xcos/Features/Scilab/Home–Scilab. Recuperado en noviembre 2022 de https://www.scilab.org/software/xcos

Siegwart, R., & Nourbakhsh, I. R. (2004). *Introduction to autonomous mobile robots,* MIT Press. The Math Works. (2012). Matlab-Simulink/versión 8.0 (R2012b), Natick, MA, USA.

Valdivieso, C., & Solís, R. (2018). Microprocesadores fundamentos y aplicaciones. Diseño embebido con simulaciones interactivas, Proyecto LATIn, Ecuador.

Yung-Hsiang, C., Tzuu-Hseng, S. L., & Yung-Yue, C. (2013). A practical trajectory tracking approach for autonomous mobile robots: Nonlinear adaptive *H*2 design. *Transactions of The Canadian Society for Mechanical Engineering*, 37(3), 385-394. https://doi.org/10.1139/TCSME-2013-0028

Cómo citar:

Soria-López, A., & Ojeda-Misses, M. A. (2024). Desarrollo y aplicación de una tarjeta embebida para el control un sistema rueda-bola. *Ingeniería Investigación y Tecnología,* 25 (04), 1-13. https://doi.org/10.22201/fi.25940732e.2024.25.4.029