DESDE 2013
https://repository.uaeh.edu.mx/revistas/index.php/icbi/issue/archive
Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI

ISSN: 2007-6363

Publicación Semestral Pädi Vol. 10 No. Especial 6 (2022) 24-32

Aplicación con el robot Cozmo para el seguimiento de trayectorias como un sistema embebido
Application with Cozmo robot for trajectory tracking control as embedded system

M. A. Ojeda-Misses[a,*]

[a] Tecnológico de Estudios Superiores de Huixquilucan, Paraje El Río s/n, Col. La Magdalena Chichicaspa. C.P. 52773, Estado de México.

Resumen

Este artículo presenta una aplicación novedosa con el robot Cozmo para el control de seguimiento de trayectoria en lazo cerrado usando el Kinect v2 como sensor de realimentación mediante como un sistema embebido de control. La contribución de este artículo es llevar a cabo el control para el seguimiento de trayectoria en lazo cerrado considerando que el robot Cozmo no contiene algún sensor integrado en sus llantas. En este trabajo es presentada la instalación y la programación para el robot Cozmo, cuyo sistema embebido está disponible comercialmente para las aplicaciones, animaciones y movimientos en lazo abierto preestablecidos. La estrategia de control para el seguimiento de trayectorias es implementada mediante Matlab/Simulink, Python, Visual Studio y Kinect v2 utilizando un controlador no lineal. La propuesta de solución permite validar y comprobar que mediante el robot Cozmo pueden ser desarrollados diversos tipos de resultados experimentales, los cuales favorecen en gran medida la aplicación de las técnicas de control con realimentación en tiempo real.

Palabras Clave: Robot Cozmo, Seguimiento de trayectorias, control no lineal, sistema embebido.

Abstract

This article presents a novel application with the Cozmo robot for closed-loop trajectory tracking control using the Kinect v2 as a feedback sensor as an embedded control system. The contribution of this article is to carry out the control for closed-loop trajectory tracking considering that the Cozmo robot does not contain any sensor integrated in its wheels. In this work, the installation and programming for the Cozmo robot is presented, whose embedded system is commercially available for applications, animations, and pre-established open-loop movements. The control strategy is implemented for trajectory tracking using Matlab/Simulink, Python, Visual Studio and Kinect v2 using a nonlinear controller. The solution proposal allows validating and verifying that various types of experimental results can be developed through the Cozmo robot, which greatly favor the application of control techniques with real-time feedback.

Keywords: Cozmo robot, Trajectory tracking, non-linear control, embedded system.

1. Introducción

El robot Cozmo es una clase de robot con fines educativos e interactivo, desarrollado por la empresa Anki (Anki1, 2020) y diseñado para interactuar y jugar con una interfaz accesible y divertida. El robot dispone de una pantalla LCD de 128x64 pixeles como cara, una cámara VGA a 30 fotogramas por segundo (fps) con reconocimiento facial, dos motores que definen su dirección y un altavoz que le permite reproducir sonidos.

Gracias a los componentes, Cozmo es capaz de expresar ciertas emociones durante su uso (felicidad, enfado, tristeza, entre otros) lo que le otorga un uso más amigable que otros robots con un fin parecido. Este robot tiene la capacidad de llevar a cabo animaciones, decir textos o frases y diferentes dinámicas y movimientos mediante el control de sus ruedas. Cozmo expresa una amplia gama de emociones reales en respuesta a su entorno. Es un gran robot innovador lleno de tecnología de Anki (Anki1, 2020) que permite interactuar con el mundo que lo rodea y tomar decisiones basadas en su estado de ánimo.

Generalmente, este robot ha sido usado para el desarrollo de diversas aplicaciones en trabajos de investigación, entre los que se tienen: en (Lefkeli *et. al*, 2020) se busca medir la

*Autor para la correspondencia: manuel.o.m@huxquilucan.tecnm.mx
Correo electrónico: manuel.o.m@huxquilucan.tecnm.mx

Historial del manuscrito: recibido el 28/03/2022, última versión-revisada recibida el 01/07/2022, aceptado el 11/07/2022, publicado el 30/11/2022. **DOI:** https://doi.org/10.29057/icbi.v10iEspecial6.8873

influencia de interacción de Cozmo mediante el uso de juegos interactivos, que consistía en preguntas de trivia entre equipos de participantes humanos y el robot Cozmo. En (Pires *et. al*, 2018), se usa un marco de código abierto para la educación mediante aplicaciones que utilizan Cozmo. En (Taylor *et. al*, 2019) es usado Cozmo para realizar diversas dinámicas en un entorno de Lego a pequeña escala implementado mediante ROS, lo que facilita su uso y ajuste como plataforma robótica. En (Touretzky, 2017), se presenta Calypso, un lenguaje inspirado por Kodu Game Lab de Microsoft, pero diseñado para programar robots móviles reales en lugar de personajes en un mundo virtual. La implementación inicial de Calypso utiliza el robot Cozmo de Anki (Anki1, 2020), son algunas de las aplicaciones que se han desarrollado con Cozmo, sin embargo, en ninguno de los artículos mencionados se ha llevado a cabo la aplicación de Cozmo para el control de seguimiento de trayectorias como se presenta en este artículo.

En lo que respecta al control de seguimiento de trayectoria, se estudian a los robots móviles como sistemas no-holonómicos (Kolmanovsky *et al.*, 1995) que pueden desplazarse de forma autónoma en ambientes de trabajo y entornos (Siegwart *et al.*, 2004). Por tal motivo han sido utilizados en una amplia gama de aplicaciones, por ejemplo, para la realización de tareas, exploración, educación, entretenimiento, entre otras (Siliciano *et al.*, 2008). Estas aplicaciones han sido posibles, en parte, gracias a los avances teóricos en el diseño de controles mediante los cuales se han logrado resolver las siguientes problemáticas asociadas con los robots móviles, entre los que se encuentran: la regulación, el seguimiento de trayectorias, la evasión de obstáculos y la navegación y el mapeo simultaneo, entre otras aplicaciones. El control de seguimiento de trayectorias es un gran problema de control que ha sido estudiado, mediante el modelo cinemático y dinámico.

Algunos de los trabajos más relevantes relativos a este tema son: en (Silva *et al.*, 2019), se lleva a cabo el control de seguimiento de trayectoria para un robot móvil con ruedas considerando la dinámica relacionada con los actuadores y la etapa de potencia. En (Rossomando *et al.*, 2010), se lleva a cabo el control dinámico neuronal adaptativo para robots móviles en control de seguimiento de trayectoria. En (Sánchez *et al.*, 2018), es integrado el hardware para la implementación de un control de seguimiento en WMRs. Finalmente, en (Silva *et al.*, 2012), se presenta una revisión literaria sobre los robots móviles y diversas aplicaciones de control sistemas no-holónomicos mediante leyes de control diferenciables. Entre las que se menciona: el resultado experimental del control de un robot móvil tipo tráiler mediante un esquema de control jerarquizado mediante un controlador un control LQR y controladores PID. También se presenta un control de un robot móvil como un sistema de estructura variable, mediante un control externo y un control interno. Se presenta un control de planitud diferencial que satisface el modelo cinemático diferencial. Se presenta un controlador dinámico para la tarea de seguimiento de trayectoria. Se expone un control basado en modos deslizantes en combinación con planitud diferencial. Se presenta dentro de los tipos de controladores un control por medio del método de par calculado para el control de seguimiento de trayectoria a través de un control basado en modos deslizantes. Se menciona el modelo cinemático de un robot diferencial aplicado al diseño de un control mediante linealización entrada-salida. Se presenta un controlador difuso para el seguimiento de trayectorias de un tractor con cinco tráileres, mientras que para la evasión de obstáculos se diseña un controlador basado en un algoritmo genético, además, se enlista un compensador neuronal dinámico adaptable para robots móviles en el control de seguimiento de trayectorias. Finalmente, en (Ojeda *et al.*, 2021) se presenta el desarrollo de un robot móvil ludoeducativo aplicado para el aprendizaje de lenguas mediante una interfaz interactiva multimodal mediante un control de seguimiento de trayectorias.

En este artículo, se presenta el control de seguimiento de trayectorias para el robot Cozmo mediante el uso de Python, Visual Studio y Matlab-Simulink. Kinect v2 es usado como cámara fija que permite la adquisición de imágenes y mapear los movimientos de Cozmo para corregir la posición y la orientación del robot en tiempo real mediante un controlador no lineal para el seguimiento de trayectorias (Canudas *et al.*, 1996). Además, se presentan resultados experimentales que validan el uso del robot Cozmo para aplicaciones de control en lazo cerrado a comparación de su uso mediante la aplicación, donde pueden encontrarse juegos, trucos, y bloques para crear dinámicas. Por otro lado, ha sido usado mediante Python para el control de sus motores (en lazo abierto), animaciones, textos y scripts (Anki1, 2020).

El trabajo presentado está organizado de la siguiente manera: la sección 2 describe la plataforma del robot Cozmo. En la sección 3 se presenta el modelo cinemático y la estrategia propuesta para el seguimiento de trayectorias. Los resultados experimentales para el seguimiento de trayectorias se muestran en la sección 4, y finalmente, se presentan las conclusiones y los trabajos futuros en la sección 5.

2. Robot Cozmo

El robot Cozmo es un robot móvil capaz de realizar un animaciones y dinámicas definidas. En la Figura. 1 se muestra el robot conectado a un teléfono Android. El robot Cozmo tiene un procesador NXP Kinetis ARM Cortex M4, un procesador Expressif ESP8266, cuatro motores de corriente directa (uno para cada llanta, uno para la cabeza y uno para la grúa), pantalla, dos orugas de caucho como medio de locomoción y una batería de litio (Anki1, 2020). El teléfono inteligente se utiliza para ejecutar en la aplicación Cozmo para utilizar el SDK desarrollado en Python.

Figura. 1. Robot Cozmo conectado a una computadora y a un teléfono inteligente.

A) Configuración

Los programas esenciales para el uso de Cozmo son: la aplicación Cozmo, el sistema operativo Android, el lenguaje

de programación Python y al programa Android Debug Device (ADB). Primero, es necesario tener instalados en la computadora Python, Android Debug Device (ADB) y la aplicación en el teléfono Android "Cozmo".

Para llevar a cabo su funcionamiento es necesario el uso de la herramienta "Símbolo del sistema", también conocido como "CMD" o "Command Prompt". Para llevar a cabo el funcionamiento de Cozmo, es necesario instalar el SDK para Cozmo (Anki2, 2020).

Asimismo, el robot requiere de las librerías de ADB para el uso de Cozmo. Cuyas librerías deben ser instaladas desde el "Símbolo del sistema". Por otro lado, es necesario el uso de un teléfono inteligente en modo "Depuración de USB", para esto es necesario ajustarlo mediante: Ajustes >> Opciones de desarrollador >> Activado >> Depuración de USB.

Lo que permitirá llevar a cabo el control de animaciones y movimiento desde el SDK de Cozmo mediante Python. Ahora bajo el "Sistema del comandos" >>adb devices, es necesario configurar el teléfono Android para vincular la comunicación con el robot (por ejemplo: 52036612f4df93ed device como se muestra en la Figura. 2).

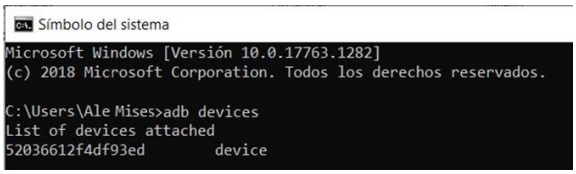

Figura. 2. Activación del teléfono inteligente como dispositivo para uso como desarrollador desde el símbolo del sistema en Windows.

Una vez realizado el paso anterior, es posible llevar a cabo el funcionamiento y la programación de Cozmo. Es decir, se abre la aplicación de Cozmo y se conecta a Cozmo mediante su base que viene integrada como se muestra la Figura. 3.

Figura. 3. Opción para conectar a Cozmo en el modo SDK desde el teléfono Android.

Esto permitirá conectar a Cozmo mediante el teléfono Android. En la pantalla aparecerá "Connecting to your Cozmo", es decir, conectando a tu robot Cozmo. Una vez que aparezca las tres indicaciones "✔" simbolizan que el robot ha sido conectado con éxito como se muestra en la Figura. 4.

Figura. 4. Indicaciones de que Cozmo ha sido conectado con éxito el modo SDK desde el teléfono Android.

Cabe mencionarse que las funciones de Cozmo jugar, trucos, programarlo mediante bloques (propios de la aplicación en Android), entre otros. Sin embargo, si se desean desarrollar aplicaciones específicas se pueden hacer mediante su programación mediante el kit de desarrollo de software (SDK), sin embargo, éste solo permite realizar movimiento predefinidos en lazo abierto. Esta función es activada mediante la opción "ENABLE SDK", es decir, está habilitado el SDK lo que permite llevar a cabo su programación mediante lenguaje en Python (véase Figura. 5).

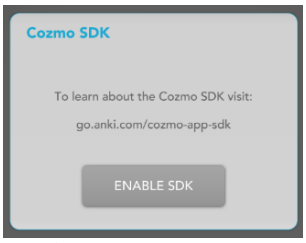

Figura. 5. Opción "ENABLE SDK" para Cozmo.

Finalmente, Cozmo estará habilitado como se muestra en la Figura. 6, que indica que Cozmo está conectado en modo SDK, y que es posible llevar a cabo su programación vía Python.

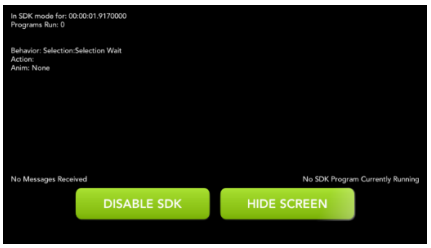

Figura. 6. Ventana desde el teléfono Android que indica que Cozmo ha sido conectado en el modo SDK.

B) Programación

Hasta el momento, se puede controlar y mover a Cozmo mediante programas en Python. La comunicación entre el SDK de Cozmo y Python es la llevada a cabo el puerto USB (Universal Serial BUS), es decir, es esencial para realizar el funcionamiento de Cozmo que permiten las siguientes funciones definidas. Las principales funciones definidas de Cozmo son:

1. Realizar una animación de Cozmo

```
<Accion Tipo="Animacion_Cozmo"
NombreAnimacion="Nombre_de_la_Animacion"  ></Accion>
```

Está función ejecuta la animación especificada. Algunas animaciones de Cozmo son:

2. Decir un texto con Cozmo

```
<Accion Tipo="Say_Text_Cozmo" Texto="Hola soy
Cozmo"  ></Accion> >
```

Mediante esta función Cozmo pronunciará el texto.

3. Accionar los velocidades de los motores de Cozmo

```
<Accion Tipo="Vels_Motores_Cozmo"
VelDer="VarVelDerecha" VelIzq="VarVelIzq" ></Accion>
```

Se establecen las velocidades de los motores de Cozmo. Se establece la velocidad de los motores a partir del valor que tengan las variables especificadas. Las variables son de tipo real.

4. Ejecutar un script en Python para Cozmo

```
    <Accion Tipo="Ejecuta_Script" Script="Ejemplo.py"
></Accion>
```

Mediante esta función se ejecuta un script Python que puede incluir llamadas al rutinas del SDK de Cozmo. En este ejemplo Cozmo realiza una animación (véase Figura. 7).

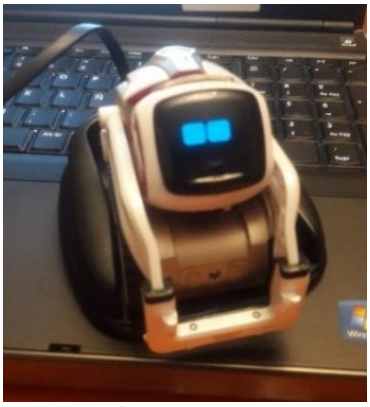

Figura. 7. Animación llevada a cabo mediante Cozmo.

El programa de Cozmo llevado a cabo en Python permite la conexión del robot mediante un IP y un puerto especifico. A continuación, se presenta dicho programa, donde el control de las velocidades de los motores es usado para el seguimiento de trayectorias.

```
import socket
import cozmo
import sys
import subprocess
#Este programa realiza la conexión de Cozmo mediante un IP y un puerto
def cozmo_program(robot: cozmo.robot.Robot):
    SayText=sys.argv[1]
def MainLoop(robot: cozmo.robot.Robot):
#Definición del IP y puerto
    UDP_IP = "127.0.0.1"
    UDP_PORT = 5005
 # Comunicación UDP
    sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
    sock.bind((UDP_IP, UDP_PORT))
    while True:
        Cmd,add = sock.recvfrom(1024)
 # Tamaño del buffer 1024 bytes
        CmdString= Cmd.decode('UTF-8')
        print ("received cmd:", CmdString
#Permite validar en todo momento si Cozmo está conectado
        if(CmdString=="Connected?"):
            sock.sendto(str.encode("CozmoOK"),addz
        if(CmdString=="DriveWheels")
 #Se realiza el control de velocidades de los motores
#Motor derecho
            Bytes = sock.recv(1024) # buffer size is 1024 bytes
            RWheelSpeed=int.from_bytes(Bytes)
            print ("Right Speed for Motor R:",RWheelSpeed)
#Motor izquierdo
            Bytes = sock.recv(1024) # buffer size is 1024 bytes
            LWheelSpeed=int.from_bytes(Bytes)
            print ("Left Speed for Motor L:", LWheelSpeed)
            robot.drive_wheels(RWheelSpeed, LWheelSpeed)
            sock.sendto(str.encode("End"),add)
            print("End Sent")
cozmo.robot.Robot.drive_off_charger_on_connect = False
cozmo.run_program(MainLoop)
```

El programa anterior permite el control de movimientos de Cozmo, que puede ser conectado desde Visual Studio mediante un programa .xaml y controlado desde Matlab-Simulink.

```
<Seguimiento_de_trayectorias
    <!—Se definen del variables-->
    <Variables No_Variables="2">
<Variable Nombre="Vel1"     Tipo="Real"    ></Variable>
<Variable Nombre="Vel2"     Tipo="Real" ></Variable>
    </Variables>
    <!—Definición de un estado-->
    <Estados No_Estados="1"
        <!--Menu-->
        <Estado  Nombre="Maneja_Cozmo">
            <Acciones No_Acciones="4">
    <!—Se imprime el estado en una interfaz-->
             <Accion Tipo="Imprime"
            Mensaje="Prueba con Cozmo"></Accion>
    <!—Se despliega una imagen del estado-->
             <Accion Tipo="Imagen"
             ArchivoImagen="Pruebaconcozmo.PNG">
     <!—Se activa la función para el mover los motores--
     >
             </Accion>
             <Accion Tipo="Vels_Motores_Cozmo"
             VelDer="Vel1" VelIzq="Vel2" ></Accion>
            </Acciones>
        </Estado>
    </Estados>
</Seguimiento_de_trayectorias>
```

Mientras que, la comunicación para el envío de datos entre Kinect v2 con Matlab/Simulink es llevada a cabo mediante un protocolo de comunicación TCP/IP. Este protocolo permite en el envío bidireccional de datos entre ambos programas que permite mapear los movimiento de Cozmo en tiempo real.

El procesamiento de datos de Kinect es llevado a cabo desde Visual Studio en un programa en C#, donde se realiza la segmentación y la medición de objetos en el plano (x,y). Estos datos son enviados en tiempo real a Matlab/Simulink y permiten llevar a cabo el control visual en lazo cerrado. Que, a su vez, emiten acciones de control para los motores de Cozmo y son enviados a Visual Studio al programa llevado a cabo en el archivo XAML.

Como fue presentado, el robot Cozmo cuenta con una serie de componentes y puede realizar movimientos con sus motores en lazo abierto, sin embargo, el robot no tiene integrado algún sensor o sistema que le permita medir sus desplazamientos a grandes distancias ni que le permita saber si el robot ha resbalado.

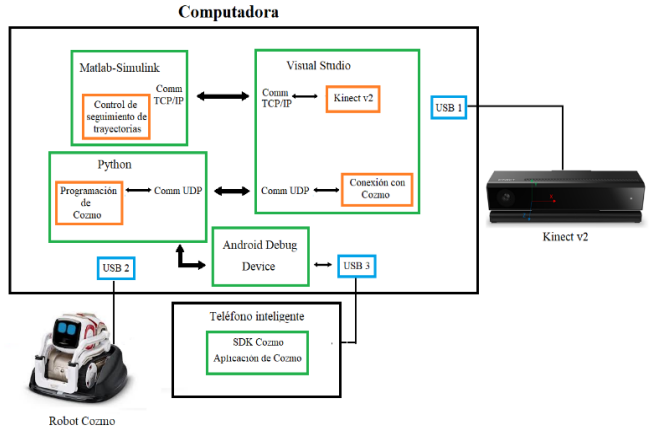

Figura. 8. Diagrama esquemático del sistema para el control de seguimiento de trayectoria de Cozmo.

Por lo tanto, es posible llevar a cabo la integración del sistema para el control de seguimiento de trayectorias en lazo cerrado, representada en la Figura. 8. El cual se basa en una dinámica que permite controlar al robot de manera automática mediante Matlab-Simulink (se lleva a cabo el control de seguimiento de trayectorias) que está conectado mediante una

comunicación TCP/IP con Visual Studio. En Visual Studio se lleva a cabo la programación de Kinect v2 que permite adquirir datos de posición y de orientación de Cozmo (útiles para llevar a cabo el control en lazo cerrado mediante el procesamiento de imágenes) y la programación para la conexión de Cozmo con Python mediante una comunicación UDP. Cabe mencionarse que en Python se lleva a cabo la programación de Cozmo desde el SDK de Cozmo para ejecutar las acciones del robot. Finalmente, en el puerto USB 1 se conecta físicamente el Kinect v2, en el puerto USB 2 está conectado el robot Cozmo, por último, en el puerto USB 3 se conecta el teléfono inteligente.

Cabe mencionarse que para el control de seguimiento de trayectorias se usa la función de accionamiento de los motores de Cozmo. Las cuales son activadas con base en la posición actual adquirida con Kinect v2, y son enviados los datos de accionamiento de Matlab-Simulink a Visual Studio, aquí se lleva a cabo el establecimiento de las velocidades a partir del valor que tengan las variables especificadas. Las variables son de tipo real en Python. Para lograr esto es necesario presentar el modelo cinemático y la estrategia de control usada para el control de seguimiento de trayectorias para Cozmo, presentado a continuación.

3. Modelo cinemático y estrategia de control

El robot Cozmo es modelado como un robot diferencial (2,0), cuya representación es mostrado en la Figura. 9.

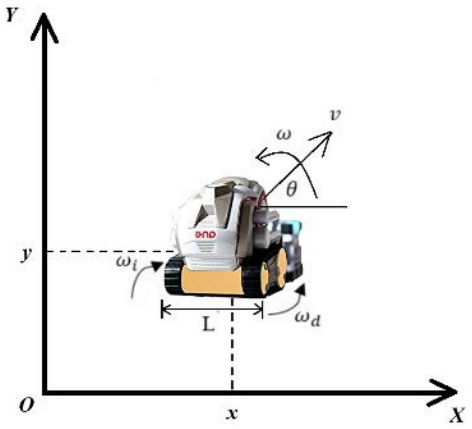

Figura. 9. Modelo cinemático para Cozmo en el plano.

La cinemática de movimiento del robot es representada como:

$$\dot{x}(t) = v(t)\,cos(\theta)$$
$$\dot{y}(t) = v(t)\,sin(\theta)$$
$$\dot{\theta}(t) = \omega(t) \quad (1)$$

donde $(\dot{x}, \dot{y})$ son las velocidades con respecto a la posición en el en el plano, el vector de posición (x, y, θ) del robot es medido con base en el marco de referencia de Kinect v2, $v$representa la velocidad lineal yω representa la velocidad angular del robot. Las velocidades lineal y angular son definidas como:

$$v(t) = \frac{r(\omega_d + \omega_i)}{2}$$
$$\omega(t) = \frac{r(\omega_d - \omega_i)}{L} \quad (2)$$

donde $\omega_d$representa la velocidad angular de la llanta derecha, ω_i representa la velocidad angular de la llanta izquierda, $r$es determina el radio de las llantas y Les definida como la distancia entre las llantas.

Para que el robot realice movimientos planeados es necesario resolver el problema de seguimiento de trayectorias de acuerdo con el control de vehículos autónomos estudiado por (Canudas *et al.*, 1996). Este problema puede formularse y solucionarse mediante estrategias de control que permitan estabilizar el robot móvil en un punto de operación, o simplemente hacer que el robot siga de forma una trayectoria de referencia. Para cualquier trayectoria deseada se emplea el siguiente modelo:

$$\dot{x}_{ref} = v_{ref}\,cos(\theta_{ref})$$
$$\dot{y}_{ref} = v_{ref}\,sin(\theta_{ref})$$
$$\dot{\theta}_{ref} = \omega_{ref} \quad (3)$$

donde se considera que $v_{ref}, \omega_{ref} > 0$y sus respectivas derivadas son acotadas, y se supone que:

$$\lim_{t\to\infty} v_{ref}(t) \neq 0$$
$$\lim_{t\to\infty} \omega_{ref}(t) \neq 0 \quad (4)$$

Esto implica que se desea seguir una trayectoria, ya que en el caso en que las velocidades fueran nulas, el problema consistiría en alcanzar una postura de referencia fija. El problema para el seguimiento de trayectorias se basa en diseñar una estrategia de control $(u_1 \quad u_2)^T = (p, p_r, v_{ref}, \omega_{ref})$ tal que:

$$\lim_{t\to\infty}[p(t) - p_{ref}(t)] = 0. \quad (5)$$

donde $p(t)$ es la posición del robot móvil medida por el vector (x, y, θ), mientras que,$p_{ref}(t)$ es la posición de referencia del robot móvil dada por el vector $(x_{ref}, y_{ref}, \theta_{ref})$. El problema de seguimiento de trayectorias definido involucra ecuaciones de error que describen la evolución temporal de la diferencia entre $p(t) - p_{ref}(t)$, por lo que, puede emplearse el siguiente cambio de coordenadas:

$$\begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} cos(\theta) & sin(\theta) & 0 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x - x_{ref} \\ y - y_{ref} \\ \theta - \theta_{ref} \end{pmatrix} \quad (6)$$

donde e_1, e_2 son los errores de posición en el marco de referencia del robot móvil y $e_3$es el error en orientación, mientras que, los errores en posición son $e_x = x - x_{ref}$ y $e_y = y - y_{ref}$, y el error en orientación es definido como $e_\theta = \theta - \theta_{ref}$, respectivamente. Las ecuaciones de error de seguimiento asociadas se obtienen derivando (6) con respecto al tiempo, es decir, realizando un cambio de variable según (Canudas *et al.*, 1996) se tiene:

$$\dot{e}(t) = \begin{pmatrix} 0 & \omega_{ref}(t) & 0 \\ -\omega_{ref}(t) & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} e + \begin{pmatrix} 0 \\ sin\, e_3 \\ 0 \end{pmatrix} v_{ref}(t) + \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \quad (7)$$

Linealizando a la ecuación (7) sobre los puntos de equilibrio$e = 0$y $u = 0$, se obtiene el siguiente sistema lineal variable en el tiempo:

$$\dot{e}(t) = \begin{pmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{pmatrix} = \begin{pmatrix} 0 & \omega_{ref}(t) & 0 \\ -\omega_{ref}(t) & 0 & v_{ref}(t) \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \quad (8)$$

El controlador no lineal empleado para el seguimiento de trayectoria es definido como:

$$u_1 = -k_1 e_1$$

$$u_2 = -k_2 v_{ref} \frac{sin(e_3)}{e_3} e_2 - k_3 e_3 \quad (9)$$

donde k_2 es una constante positiva yk_1, k_3son funciones estrictamente positivas en $\mathbb{R} \times \mathbb{R} - (0,0)$. Dichas funciones son definidas como:

$$k_1 = k_3 = 2\zeta \sqrt{\sigma v_{ref}^2 + \omega_{ref}^2} \quad (10)$$

con $k_2 = \sigma$ y ζ ganancias reales y positivas.

Las estrategias de control son aplicadas a las velocidades angulares de cada motor del robot. Ahora, es primordial comprobar la estabilidad del sistema mediante la estrategia de control. Para esto, se considera la dinámica del error y un punto de equilibrio estable para el sistema dado por $e = 0$. Considerando la siguiente función candidata de Lyapunov:

$$V(e) = \frac{k_2}{2}(e_1^2 + e_2^2) + \frac{e_3^2}{2} \quad (11)$$

derivando (11) con respecto al tiempo t a lo largo de las trayectorias del sistema se tiene:

$$\dot{V}(e) = k_2 e_1 \dot{e}_1 + k_2 e_2 \dot{e}_2 + e_3 \dot{e}_3 \quad (12)$$

De la ecuación 8 pueden obtenerse $\dot{e}_1 = u_1 + \omega_{ref} e_2, \dot{e}_2 = v_{ref}\, sin(e_3) - \omega_{ref} e_1, \dot{e}_3 = u_2$, por lo tanto:

$$\dot{V}(e) = k_2 e_1 (u_1 + \omega_{ref} e_2) + k_2 e_2 (v_{ref}\, sin(e_3) - \omega_{ref} e_1) + e_3 u_2 \quad (13)$$

por lo tanto, sustituyendo la estrategia de control dada por la ecuación (9), simplificando y realizando las operaciones correspondientes se tiene:

$$\dot{V}(e) = k_2 e_1 (-k_1 e_1) + k_2 e_2 v_{ref}\, sin(e_3) + e_3 \left(-k_2 v_{ref} \frac{sin(e_3)}{e_3} e_2 - k_3 e_3\right) \quad (14)$$

$$\dot{V}(e) = -k_1 k_2 {e_1}^2 - k_3 {e_3}^2 \quad (15)$$

donde se debe cumplir que $k_1, k_2, k_3 > 0$ para que $\dot{V}$sea negativa definida y garantice que el sistema sea asintóticamente estable.

4. Resultados experimentales

En esta sección se presentan los resultados experimentales aplicados con Cozmo usando dos trayectorias, primero una trayectoria senoidal en el apartado A), mientras que, en el apartado B), se presenta una trayectoria para una lemniscata. El robot móvil usado en los experimentos está integrado mediante dos motores de corriente directa, controlados mediante Python.

Se emplea el Kinect v2 con configuración de cámara fija que permite monitorear la posición y la orientación del robot, cuyas variables son realimentadas y es capaz de llevar a cabo el control de seguimiento de trayectoria mediante la programación llevada a cabo en Visual Studio y envía los datos en tiempo real a Matlab2012b-Simulink mediante QuaRC Quanser.

El sistema integrado para el control de seguimiento de trayectorias es mostrado en la Figura. 10, este contiene: 1) el Kinect v2, 2) la computadora personal Dell Mobile Precision M4700, 3) los programas en la computadora: Matlab-Simulink-QuaRC, Visual Studio, Python y Android Debug Device, 4) el teléfono inteligente, 5) la aplicación de Cozmo y 6) el robot Cozmo.

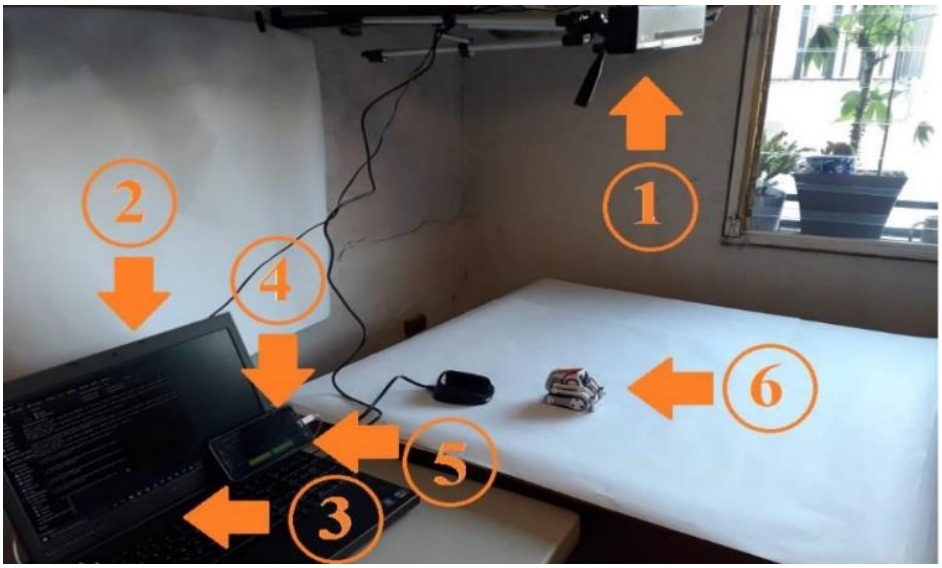

Figura. 10. Sistema integrado para el control de seguimiento de trayectorias integrado con 1) Kinect v2, 2) computadora personal Dell Mobile Precision M4700, 3) programas en la computadora: Matlab-Simulink-QuaRC, Visual Studio y Python, 4) teléfono inteligente, 5) aplicación de Cozmo y 6) robot Cozmo.

En el campo de la robótica, el Kinect v2 se ha utilizado principalmente en asociación con robots articulares y con robots móviles para realizar aplicaciones e interfaces. La mayoría de las aplicaciones han sido implementadas para los robots móviles, mientras que las interfaces han sido desarrolladas solo para robots articulares. Entre las aplicaciones más notables están el sistema articular de validación de dispositivos RGBD para medir el equilibrio de pacientes en terapia (Ayed *et al.*, 2017); la aplicación para la captura automática de posturas corporales (Muñoz *et al*., 2018) y el robot móvil compacto bajo ROS para la navegación y la localización en ambientes no estructurados (Araújo *et al.*, 2014).

El Kinect v2 es usado mediante las herramientas del kit de desarrollo de software de Kinect v2, que permiten mediante la *Camera space* (Microsoft, 2021), llamado el espacio de trabajo de la cámara.

El espacio empleado es de 1.5m x 1.5m adquirido con la cámara mediante un sistema de coordenadas 3D utilizado por Kinect. El sistema de coordenadas se define el origen (x = 0, y

= 0, z = 0) está ubicado en el centro del sensor IR en Kinect, cuya relación es 1 unidad = 1 metro.

Mediante las funciones basadas en matrices:

"*MapDepthPointsToCameraSpace*",
"*GetDepthFrameToCameraSpaceTable*"

es posible procesar las imágenes adquiridas mediante Kinect (Microsoft, 2021).

Una vez obtenido un píxel de profundidad, es posible que desee conocer el valor de infrarrojos en el espacio (x,y) con base en el marco de referencia de Kinect. Esta operación es fácil porque la profundidad y los infrarrojos es adquirida mediante Kinect, capaz de muestrear la misma fila / columna en la imagen infrarroja en tiempo real (Microsoft, 2021).

El comando usado para asignar un marco del espacio de color al espacio de la cámara es *MapColorFrameToCameraSpace* (Microsoft, 2021), cuya sintaxis en C# es declarada mediante:

```
public void MapColorFrameToCameraSpace
(Array<UInt16>[] depthFrameData, out
Array<CameraSpacePoint>[]cameraSpacePoints
```

La implementación del procesamiento de imágenes con Kinect es llevada a cabo en una computadora personal, la cual integra los componentes materiales y de software, donde se estima el centroide del rectángulo de color rojo y se despliega su posición en el plano (x,y). El sistema está integrado mediante un entorno Matlab2012b-Simulink-QuaRC y Visual Studio (C#, XAML) en conjunto con el Kinect Studio Software Development Kit (SDK v2). El procesador ejecuta el programa de la conexión con Cozmo que lo enlaza con Python y con el SDK del robot.

Para el control de seguimiento de trayectorias se implementó el controlador no lineal presentado e implementado mediante diagramas a bloques. La comunicación entre –que envía la señal de control de los motores– Visual y Python se hace a través de un puerto UDP, proporcionado por el puerto USB correspondiente.

En la Figura. 11 se muestra imagen adquirida con Kinect v2, donde adquiere las imágenes con Kinect para la trayectoria senoidal y en forma de lemniscata, presentadas a continuación.

A) Trayectoria senoidal

Para la trayectoria senoidal se usa la siguiente ecuación paramétrica:

$$x_{ref}(t) = v_o t$$

$$y_{ref}(t) = A_o \, sin(x_{ref}) \qquad (16)$$

donde los parámetros de la función paramétrica son determinados mediante la amplitud $A_o = 0.5\text{m}$ y la velocidad$v_o = 0.08\text{m/s}$ en el intervalo $t \in [0{,}78]$. Además, se consideran la siguiente posición y orientación inicial para el robot Cozmo $x_o = 0$ m, $y_o = 0.1\text{m}$, $\theta_o = \pi/4$ rad.

Por otro lado, se usan las velocidades de referencia $v_{ref} = 0.1\text{m/s}$, $\omega_{ref} = 0.1\text{rad/s}$, $k_2 = 1.1$y $\varsigma = 0.1$, por lo que, las ganancias son $k_1 = k_3 = 0.028$, que garantizan que $\dot{V}$sea negativa definida y el sistema sea asintóticamente estable.

El resultado para la trayectoria senoidal es mostrado en la Figura. 12, donde son comparadas las trayectorias de referencia y real del robot Cozmo.

Figura. 11. Imagen adquirida con Kinect v2 con Cozmo realizando el seguimiento de trayectorias.

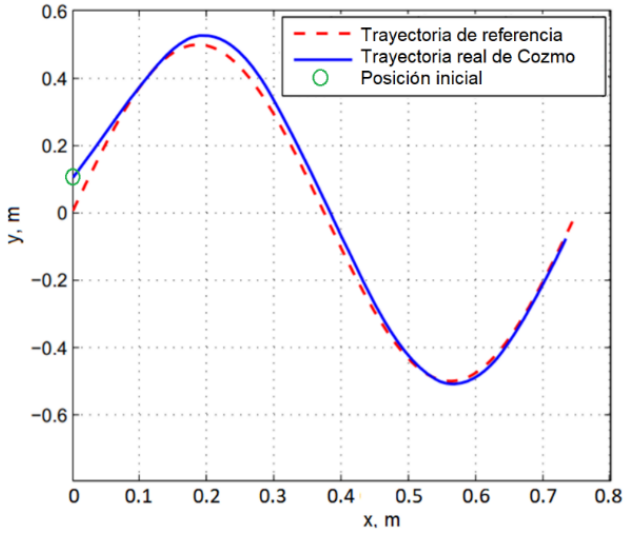

Figura. 12. Resultado experimental obtenido para la trayectoria senoidal

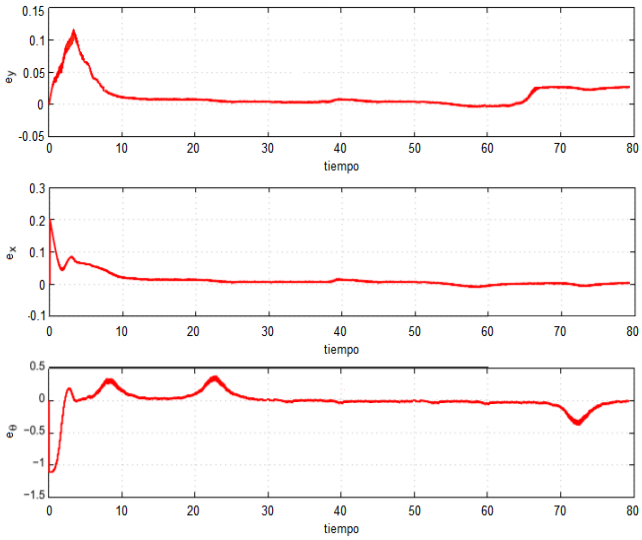

Figura. 13. Señales de error e_x, e_y, e_θpara la trayectoria senoidal.

Asimismo, en la Figura. 13 son presentadas las señales de los errores de posición e_x, e_y y de orientación e_θ para los parámetros y las ganancias mencionadas.

B) Trayectoria lemniscata

Para la trayectoria en forma de una lemniscata se usa la siguiente ecuación paramétrica:

$$x_{ref}(t) = r_o\left(-cos\left(\frac{2\pi}{p_o}t\right)\right)$$
$$y_{ref}(t) = r_o\, sin\left(\frac{4\pi}{p_o}t\right) \quad (17)$$

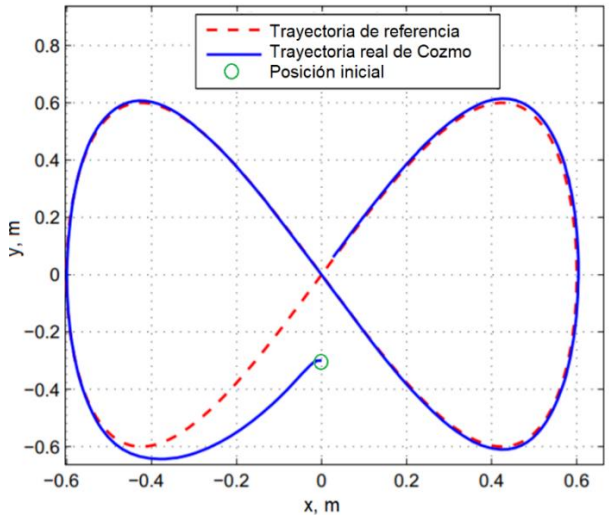

Figura. 14. Resultado experimental obtenido para la trayectoria de la lemniscata.

Los parámetros de la función paramétrica de la lemniscata son determinados mediante la amplitud $r_o = 0.6$m y la frecuencia $p_o = 60\,s^{-1}$en un intervalo de tiempo $t \in [0,60]$. Además, se consideran la siguiente posición y orientación inicial para el robot Cozmo $x_o = 0$m, $y_o = -0.3$m, $\theta_o = \pi$ rad. Por otro lado, se usan las velocidades de referencia $v_{ref} = 0.2$m/s, $\omega_{ref} = 0.2$rad/s,$k_2 = 1.1$y $\varsigma = 0.1$, por lo que, las ganancias son $k_1 = k_3 = 0.057$, que garantizan que $\dot{V}$sea negativa definida y el sistema sea asintóticamente estable.

La trayectoria en forma de lemniscata es mostrada en la Figura. 14, donde son presentadas las trayectorias de referencia y real del robot Cozmo. Asimismo, en la Figura. 15 son presentados los errores de posición y de orientación, respectivamente. Donde puede verse que la propuesta de control presentada presenta resultados viables, factibles y con buen rendimiento. Finalmente, los parámetros físicos del robot Cozmo usados son un radio (r), $r = 0.0125$m y una distancia entre las llantas (L), $L = 0.045$m, respectivamente.

5. Conclusiones

Este artículo presenta una aplicación novedosa mediante el robot Cozmo para el control de seguimiento de trayectorias que brinda experiencia práctica en usando un sistema de control integrados. El proyecto se estructura en torno al robot Cozmo, cuyo sistema integrado está diseñado para aplicaciones definidas y mediante el control de motores en lazo abierto. Por ende, cabe señala que la propuesta del artículo es innovadora ya que usa un diseño de control en lazo cerrado integrado por Matlab/Simulink, Python, Visual Studio y el Kinect v2 como sensor de realimentación. La estrategia de control propuesta fomenta el uso de Cozmo bajo una técnica de control con realimentación en tiempo real.

La estrategia de control no lineal presentada para el seguimiento de trayectorias presentó un desempeño aceptable al ser aplicada para realizar implementaciones prácticas con Cozmo, incluso con un robot de fábrica, que en muchos casos los recursos y la programación es limitada. Además, se logró implementar un controlador donde Kinect v2 funciona como sensor de realimentación y es capaz de adquirir y monitorear la posición y la orientación del robot bajo un modelo cinemático. En suma, se brinda un análisis de estabilidad que permite corroborar la estabilidad del sistema.

El control de seguimiento de trayectorias fue llevado a cabo en espacios cerrados, donde los niveles de iluminación no afectan el reconocimiento ni la adquisición de los datos. Finalmente, esta aplicación permite aprovechar a Cozmo bajo diversos esquemas de control de localización y mapeo, y puede ser aplicado para aplicaciones sociales, educativas, guías, lúdicas, entre otras.

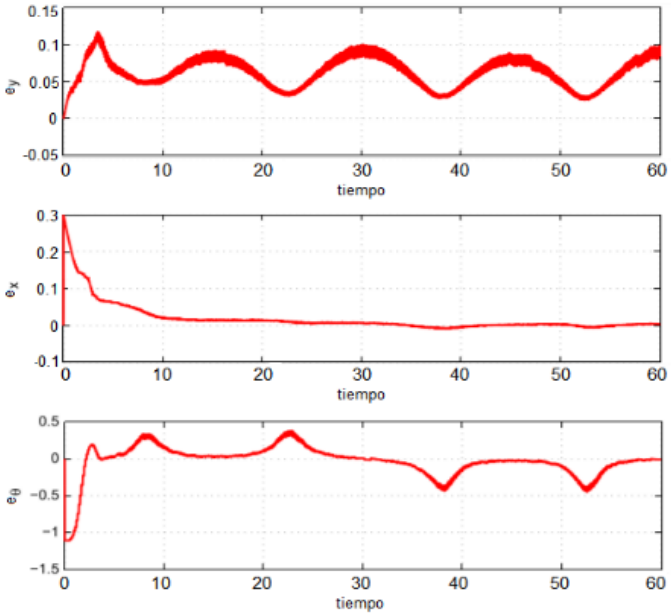

Figura. 15. Señales de error e_x, e_y, e_θpara la trayectoria de la lemniscata.

Agradecimientos

Este artículo ha sido posible gracias al apoyo de la Dirección General de Asuntos del Personal Académico de la Universidad Nacional Autónoma de México (proyecto PAPIIT IT400119), así como al Departamento de Control Automático del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, Unidad Zacatenco.

Referencias

Anki Inc. (1), “Meet Cozmo”, Cozmo. [En línea] https://www.anki.com/en-us/cozmo, Accessed: 28-08-2020.

Anki Inc. (2), “Instalación de Cozmo”, Cozmo. http://cozmosdk.anki.com/docs/, Accessed: 28-08-2020.

Araújo, A., Portugal, M. S., Couceiro, J., Sales, J. & Rocha, R. P. (2014). Desarrollo de un robot móvil compacto integrado en el middleware ROS. Revista Iberoamericana de Automática e Informática Industrial, 11(3), 315-326. https://doi.org/10.1016/j. riai.2014.02.009

Ayed, I., Moyà, B., Martínez, P., Varona, J., Ghazel, A. & Jaume, A. (2017). Validación de dispositivos RGBD para medir terapéuticamente el

equilibrio: el test de alcance funcional con Microsoft Kinect. Revista Iberoamericana de Automática e Informática Industrial, 14(1), 115-120 https://doi.org/10.1016/j.riai.2016.07.007

Canudas de Wit, C., Siliciano, B., Bastin, G., Theory of Robot Control, Springer, 1996.

Kolmanovsky, I., and McClamroch, N. H., "Developments in nonholonomic control problems," IEEE Control Systems Magazine, vol. 15, no. 6, pp. 20–36, 1995.

Lefkeli, D., Ozbay, Y., Gurhan-Canli, Z., Eskenazi, T., "Competing with or Against Cozmo, the Robot: Influence of Interaction Context and Outcome on Mind Perception,", International Journal of Social Robotics, Springer, 2020. DOI: https:10.1007/s12369-020-00668-3

Microsoft. "Coordinate mapping", Camera space, https://docs.microsoft.com/en-us/previous-versions/windows/kinect/dn785530(v=ieb.10) Accessed: 20-10-2019.

Muñoz, R., Barcelos, T., Villaroel, R., Guíñez, R. & Merino, E. (2018). Body posture visualizer to support multimodal learning analytics. IEEE Latin America Transactions, 2706-2715, https://doi.org/10.1109/TLA.2018.8795111

Ojeda-Misses M. A., Silva-Ochoa H. & Soria-López A. (2021). Ludibot: Interfaz humano-robot móvil para el aprendizaje lúdico de idiomas. Ingeniería Investigación y Tecnología, 22 (03), 1-10. https://doi.org/10.22201/fi.25940732e.2021.22.3.021

Pires Kusumota, V. L., Vidal Aroca, R. and Martins, F. N., "An Open Source Framework for Educational Applications Using Cozmo Mobile Robot," 2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE), Joao Pessoa, 2018, pp. 569-576, DOI: 10.1109/LARS/SBR/WRE.2018.00104.

Rossomando, F. G., Soria, C., and Carelli, R., "Adaptive Neural Dynamic Compensator for Mobile Robots in Trajectory tracking control," in *IEEE Latin America Transactions*, vol. 9, no. 5, pp. 593-602, 2011, DOI: 10.1109/TLA.2011.6030965.

Sánchez, C. M., et al., "An Embedded Hardware for Implementation of a Tracking Control in WMRs," in IEEE Latin America Transactions, vol. 16, no. 7, pp. 1835-1842, July 2018, DOI: 10.1109/TLA.2018.8447346.

Siciliano, B., and Khatib, O., Handbook of Robotics. Heidelberg, Berlin, Germany: Springer-Verlag, pp. 1065–1298, 2008.

Siegwart, R., and Nourbakhsh, I. R., Introduction to Autonomous Mobile Robots. Cambridge, MA, USA: MIT Press, 2004.

Silva Ortigoza, R., Aranda, M. M., Ortigoza, G. S., Guzmán, V. M. H., Vilchis, M. A. M., González, G. S., Lozada, J. C. H. and. Carbajal, M. O., "Wheeled mobile robots: A review," IEEE Latin America Transactions, vol. 10, no. 6, pp. 2209-–2217, 2012. DOI: 10.1109/TLA.2012.6418124

Silva Ortigoza, R., Garcia Sanchez, J. R., Hernandez Guzman, V. M., Marquez Sanchez, M., and Marcelino Aranda, M., "Trajectory Tracking Control for a Differential Drive Wheeled Mobile Robot Considering the Dynamics Related to the Actuators and Power Stage," in *IEEE Latin America Transactions*, vol. 14, no. 2, pp. 657-664, Feb. 2016, DOI: 10.1109/TLA.2016.7437207.

Taylor, H. M., Dondrup, C. and Lohan, K. S. "Introducing a Scalable and Modular Control Framework for Low-cost Monocular Robots in Hazardous Environments," 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 2019, pp. 6421-6426, DOI: 10.1109/IROS40897.2019.8967521.

Touretzky, D. S., "Computational thinking and mental models: From kodu to calypso," 2017 IEEE Blocks and Beyond Workshop (B&B), Raleigh, NC, 2017, pp. 71-78, doi: 10.1109/BLOCKS.2017.8120416.