VÍNCULO DE UN BRAZO ROBÓTICO AL INTERNET DE LAS COSAS ROBÓTICAS

Mtro. Luis Alberto Flores Montaño
lfloresm1703@alumno.ipn.mx
Mtro. Rodrigo Vázquez López
rodrigo_em2@hotmail.com
Dr. Juan Carlos Herrera Lozada
jlozada@ipn.mx

Instituto Politécnico Nacional
Centro de Innovación y Desarrollo Tecnológico en Cómputo

Dr. Jacobo Sandoval Gutiérrez
j.sandoval@correo.ler.uam.mx

Universidad Autónoma Metropolitana Unidad Lerma
Departamento de Procesos Productivos

Boletín No. 82
1o. de enero de 2021

Resumen

La reciente expansión de tecnologías de la información y las comunicaciones ha traído una serie de cambios en el estilo de vida tanto social, económico e industrial. En el aspecto industrial se ha alcanzado la cuarta revolución industrial y generado la llegada de nuevas tecnologías como la conectividad entre dispositivos, y su registro de información a través de sensores, esto también es conocido como IoT y actualmente con la integración de robots en escenarios IoT son conocidos como el Internet de las cosas robóticas (en inglés IoRT).
Tomando en consideración lo anterior, en este documento se propone mostrar un caso de aplicación del IoRT. El sistema consta de un nodo maestro (computadora principal), la cual hace uso de MATLAB con un Middleware (ROS) y un brazo robótico de 6 grados de libertad que será el nodo esclavo (controlado por una Raspberry Pi). La información del proceso será almacenada en el nodo maestro, mientras la ejecución se realizará en el nodo esclavo. Las pruebas fueron aceptables y se pudo demostrar la viabilidad del control de múltiples robots con base en la conectividad del IoRT. Por último, la aplicación servirá de base para otros tipos de robots móviles, aéreos, entre otros.
Palabras Clave: Industria 4.0, Brazo Robótico, MATLAB, IoRT, ROS, Raspberry Pi.

Abstract

The recent expansion of information and communication technologies has brought a series of changes in the social, economic, and industrial lifestyle. In the industrial

aspect, the fourth industrial revolution has been reached and generated the arrival of new technologies such as connectivity between devices, and their registration of information through sensors, this is also known as IoT and currently with the integration of robots in IoT scenarios They are known as the Internet of robotic things (IoRT).
Taking the above into consideration, this document proposes to show a case of application of the IoRT. The system consists of a master node (main computer), which uses MATLAB with a Middleware (ROS) and a 6-degree-of-freedom robotic arm that will be the slave node (controlled by a Raspberry Pi). The process information will be stored in the master node, while the execution will be carried out in the slave node. The tests were acceptable and the feasibility of controlling multiple robots based on IoRT connectivity could be demonstrated. Finally, the application will serve as the basis for other types of mobile and aerial robots, among others.
Keywords: Industry 4.0, Robotic Arm, MATLAB, IoRT, ROS, Raspberry Pi .

I. Introducción

El desarrollo y la rápida expansión del internet se han convertido en nuevas e inevitables innovaciones; todo esto ha traído la cuarta revolución industrial. La industria 4.0 incluye dispositivos de mayor calidad y más rápidos para comunicarse entre sí y con las personas, con uso de sensores y sistemas de control, datos de proceso, y corrección de deficiencias y errores. Los sistemas ahora pueden comunicarse entre sí y comenzar a tomar decisiones sin necesidad de intervención humana (Muhuri,2019), esto también es conocido como el Internet de las Cosas (IoT). Las aplicaciones de IoT ya se están aprovechando en diversos dominios, como el campo de servicios médicos, el comercio minorista inteligente, el cliente servicio, casas inteligentes, vigilancia ambiental e industrial Internet. (Mazzara et al., 2019)(Mazzara et al., 2019)

El IoT y la robótica no pueden considerarse como dos dominios separados en estos días. El Internet de las cosas robóticas (en inglés IoRT) es un concepto que se ha introducido recientemente para describir la integración de las tecnologías de robótica en escenarios IoT, como consecuencia, estos dos campos de investigación comenzaron a interactuar y a vincular las comunidades de investigación. (Mazzara et al., 2019)

Teniendo en cuenta el desarrollo de software para robots, este se está convirtiendo en un proceso cada vez más difícil, principalmente debido a la creciente complejidad de las tareas que deben realizar. Muchos robots están especializados para realizar tareas muy específicas, generando una variedad de hardware, software y programación específica. Como resultado, para superar estos problemas, nacieron diferentes entornos de trabajo para facilitar el desarrollo de robots. (Galli et al., 2017). Robot Operating System (ROS) es el líder entorno de desarrollo en robótica, y ofrece herramientas y bibliotecas para el desarrollo de sistemas robóticos. En años recientes ROS ha ganado una amplia difusión para la creación de sistemas robóticos, no solo en el laboratorio sino también en la industria. (Galli et al., 2017)

En este trabajo de investigación se pretende estudiar la viabilidad de esta herramienta, así como su utilidad. Varios algoritmos simples que ya se utilizan con ROS, pueden verificar la eficiencia de la comunicación aprovechando las herramientas de procesamiento de datos que proporciona MATLAB. El objetivo es realizar las mismas acciones que se realizan en un ROS entorno a través de MATLAB Robotics System Toolbox, que permitirá aprovechar la estructura del mensaje ROS en un entorno Windows sin usar Linux y una microcomputadora como es el caso de una Raspberry Pi. De esta manera es posible unir estas herramientas y ampliar la discusión sobre el desarrollo de este campo interdisciplinario que es la robótica y el IoRT. Adicionalmente, el documento proporciona una visión general, análisis y desafíos de posibles soluciones para Internet de Cosas robóticas, discutiendo los problemas de la arquitectura IoRT.

II. Materiales y métodos

Para el desarrollo de este proyecto, son necesarias varias herramientas. Se utilizan una PC y una microcomputadora Raspberry pi con distintos sistemas operativos. En la PC con Windows 10 se lleva a cabo el puente de comunicación con ROS, la y la ejecución de MATLAB.

Una arquitectura basada en ROS permite que diversos nodos y un máster se comuniquen entre sí por medio de mensajes. Los mensajes se pueden depositar en diversos tópicos a través una red TCP/IP. La figura 1 muestra el modelo de la arquitectura propuesta la cual se compone de la PC ejecutando

MATLAB (nodo maestro de ROS) y el nodo esclavo que se ejecuta dentro de la Raspberry Pi. El brazo robótico se conectó de forma física al puerto GPIO de la Raspberry para ejecutar las órdenes de control.

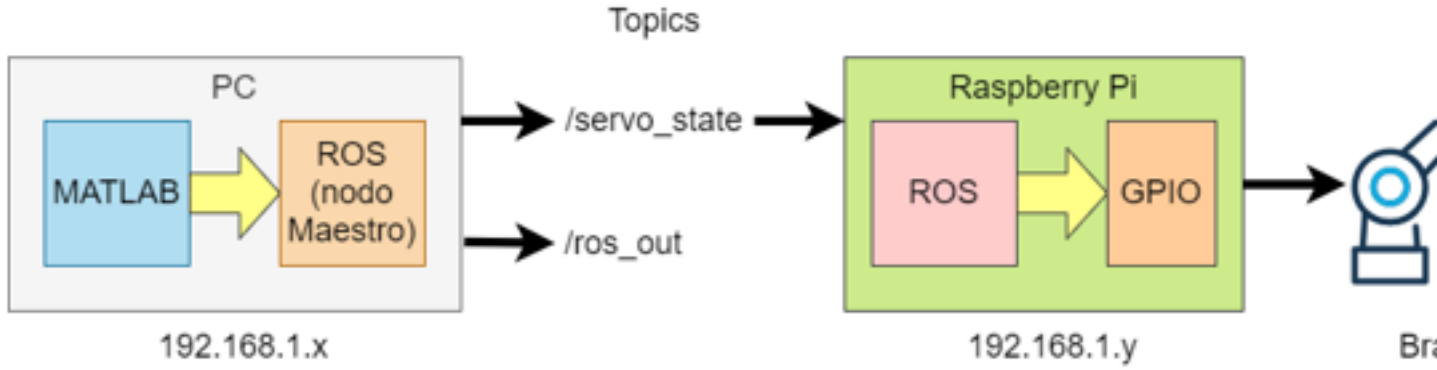

Figura 1. Arquitectura basada en ROS para el control del brazo.

Modelado

Utilizando las convenciones del análisis de la cinemática directa (’Siciliano & ’Khatib, 2016) se ha propuesto utilizar los parámetros de Denavit y Hartenberg (DH), cuyas bases han sido aprovechadas en diferentes problemáticas de la robótica. En todas sus formas, la convención requiere solo cuatro en lugar de seis parámetros para ubicar un marco de coordenadas en relación con otro. (Radavelli et al., 2012)

Los cuatro parámetros constan de dos parámetros de enlace, el enlace longitud ai y el de torsión de αi, y dos parámetros de unión, el desplazamiento de articulación di y el desplazamiento de ángulo i.

La obtención de este modelo es imprescindible para abordar la mayor parte de los problemas asociados al control y programación de los robots manipuladores, caminantes o humanoides. El origen del método DH permite establecer la relación entre dos barras rígidas consecutivas unidas por una articulación de un grado de libertad, mediante una matriz i-1A1 , hacia 4 parámetros (θ, d, a, α) asociados a 4 movimientos consecutivos (rotación y traslación en z, seguidos de traslación y rotación en x). Para que esto sea posible, es preciso asociar a cada eslabón del sistema de coordenadas posicionado según determinadas reglas. (Barrientos et al., 2012)

Sin embargo, es posible obtener directamente la matriz i-1A1 analizando cada eslabón que compone al brazo manipulador. Para obtener los parámetros α_i,a_i,d_i y θ_i se debe ubicar un sistema coordenado en cada elemento recordando que el vector de posición z debe ser normal al plano de movimiento del eslabón. Un ejemplo de colocación de sistemas coordenados se observa en la figura 2.

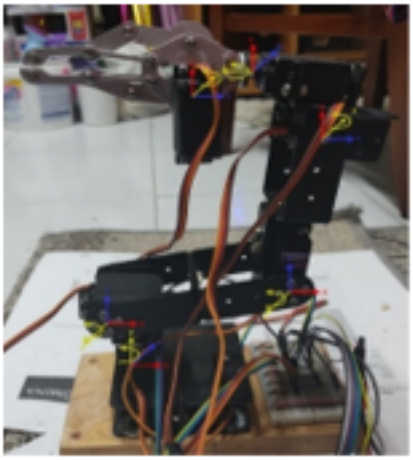

Figura 2. Ubicación de los ejes coordenados en cada unión.

Finalmente, se puede sintetizar el contenido de las matrices i-1A1 en una sola matriz DH con los parámetros α_i,a_i,d_i y θ_i de todos los eslabones que componen al manipulador con la finalidad de facilitar el cómputo del modelo. Los resultados de la matriz DH del brazo se observan en la Tabla 1.

Tabla. 1.
Matriz DH con los valores de cada encadenamiento del brazo.

i	α_i	a_i	d_i	θ_i
1	0	0	0	θ_1
2	$-\pi/2$	0	0	θ_2
3	0	10.5	0	θ_3
4	π	0	10	$-\pi/2$
5	$\pi/2$	0	0	θ_5
6	$-\pi/2$	0	0	θ_6

Caracterización de los servomotores

El brazo robótico está compuesto de 6 servomotores Tower Pro modelo MG996R. De acuerdo con la hoja de datos, dichos servomotores pueden funcionar con un voltaje de entre 4.8 y 6.6V en DC consumiendo aproximadamente un mínimo de 170 mA de corriente (sin carga) y un máximo de 1400 mA (cuando el motor aplica el máximo torque). La señal de control es una onda PWM de entre 1 y 2 ms con una frecuencia de 50Hz.

Para realizar el proceso de caracterización, es necesario manipular cada servomotor de forma individual con la finalidad de encontrar los valores de ancho de pulso PWM para obtener la posición deseada. En la figura 3 se puede observar el diagrama de conexión entre los servomotores y la Raspberry pi. Los servomotores se conectaron físicamente a los pines 11, 13 ,15, 16, 18 y 22 del puerto GPIO de la tarjeta y se utilizó una fuente de alimentación externa regulada de 5V y 5A con el propósito de cubrir la demanda energética de los 6 servomotores.

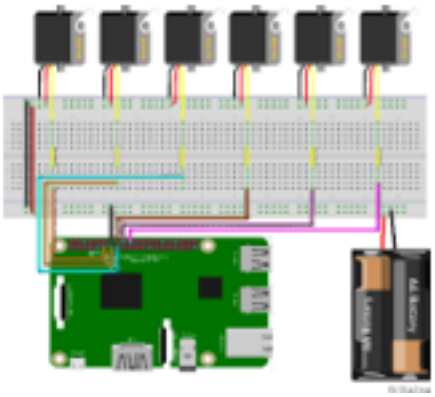

Figura 3. Diagrama de conexión entre los servomotores del brazo y la Raspberry Pi.

La microcomputadora Raspberry Pi puede generar ondas PWM en las salidas del puerto GPIO vía software. Dicha posibilidad permitió que se programara un script en Python con el cual se manipuló el ciclo de trabajo de la onda PWM. El experimento consistió en aumentar o disminuir en el ciclo de trabajo en intervalos de 0.5 % hasta obtener los valores correspondientes a las posiciones máxima y mínima a los que se puede mover cada servomotor. Los resultados obtenidos se presentan en la Tabla 2.

Tabla 2.
Valores experimentales obtenidos de ciclo de trabajo (en porcentaje).

Servomotor	Rango de movimiento		Posición Home
	Mínimo %	Máximo %	
Base	2	12	7
Brazo 1	2	12	2
Brazo 2	2	12	3
Brazo 3	2.5	9.5	7
Base gripper	2	12	7
Gripper	4	8	8

Las tablas anteriores tienen como una consideración general el espacio de trabajo previamente caracterizado. Como ejemplo, se puede observar el mecanismo del gripper y el cuarto encadenamiento (servomotor etiquetado como Brazo 3). Para conocer el rango aproximado de movimiento en grados se utilizó la siguiente relación:

$$g = (x-2)\frac{180}{10} = 18(x-2)$$

Donde g representa el valor en grados y x el valor de ciclo de trabajo de la onda PWM en porcentaje. En la Tabla 3 se puede observar los valores de rango de movimiento en grados.

Tabla 3.
Rangos de movimiento en grados para el espacio de trabajo.

Servomotor	Rango de movimiento		Posición Home
	Mínimo °	Máximo °	
Base	0	180	90
Brazo 1	0	180	0
Brazo 2	0	180	18
Brazo 3	9	135	90
Base gripper	0	180	90
Gripper	36	108	108

Nodo maestro

MATLAB es una plataforma que está optimizada para resolver problemas de ingeniería y científicos. Los gráficos integrados facilitan mucho la visualización de los datos y la obtención de información a partir de ellos. Hay una gran diversidad de librerías preinstaladas que permiten trabajar inmediatamente con algoritmos para su dominio o ver ejemplos de aplicaciones que se quieran realizar. (Gómez & Barca, 2017)

Para realizar la comunicación con ROS, hay que instalar los Add-On de ROS y de soporte de MATLAB y Simulink en Raspberry Pi. Una vez hecho esto ya se puede establecer la comunicación con la Raspberry Pi y posteriormente con un dispositivo robótico. Con ella se puede diseñar y desarrollar algoritmos para la representación de mapas, planear rutas entre otros. System Toolbox contiene interfaces entre MATLAB y ROS, la cual hace posible realizar la comunicación remota con la Raspberry Pi y a su vez con el brazo.

Para iniciar el nodo maestro de ROS en MATLAB se teclea el comando:

» rosinit(' NodeHost ' ,< Dirección IP >)

Donde < Dirección IP > se sustituye por la dirección IP de la PC que ejecuta MATLAB. Para verificar la lista de nodos y tópicos disponibles se utilizan los comandos:

» rosnode list » rostopic list

Para terminar la ejecución de ROS se utiliza el comando:

» rosshutdown

Nodo esclavo

La instalación de los Add-On de Raspberry en MATLAB generará una imagen del sistema operativo Raspberry OS con los paquetes necesarios para comunicarse con MATLAB. Una vez instalada la imagen del SO en la Raspberry pi, es necesario ingresar a la terminal y ejecutar los siguientes comandos para completar la instalación de ROS:

$ cd ~/catkin_ws/ $ catkin_make $ echo "source ~/catkin_ws/devel/setup.bash"» ~/.bashrc

Una buena práctica es crear paquetes para mantener organizado el código fuente de las aplicaciones que generemos. Para crear un paquete, es necesario ejecutar los siguientes comandos dentro del directorio catkin_ws.

$ cd src/ $ catkin_create_pkg rospy std_msgs $ cd .. $ catkin_make

Es importante aclarar que es el nombre que tendrá el paquete mientras que rospy y std_msgs son paquetes adicionales y necesarios para el funcionamiento del script. Una vez creado el paquete, los siguientes comandos sirven para crear un directorio dónde almacenar los scripts y un archivo vacío para colocar el código.

$ cd ~/catkin_ws/src/nombre_paquete $ mkdir scripts $ cd scripts/ $ touch nombre_archivo.py $ chmod +x nombre_archivo.py

El código del script se presenta en el Anexo 1. La función del script es iniciar un nodo en ROS y suscribirse a un tópico para recibir mensajes. Los mensajes recibidos indican la nueva posición a la cual se deben mover los servomotores. Es importante aclarar que para ejecutar el script se deben utilizan los siguientes comandos:

$ export ROS_MASTER_URI=http://:11311 $ rosrun nombre_paquete prueba.py

Donde es donde se debe colocar la dirección IP de la computadora que ejecuta el nodo maestro (MATLAB).

III.Resultados

Para manipular la posición del brazo, el nodo maestro debe enviar uno o más mensajes al tópico al cual se encuentra suscrito el nodo esclavo. Para llevar a cabo dicha acción, se ejecuta el siguiente código en MATLAB:

b = rospublisher('/servo_state', 'std_msgs/String'); msg = rosmessage(b); msg.Data = '7.0, 2.0, 3.0, 7.0, 7.0, 7.8'; send(b,msg);

En dicho código se observa que '/servo_state' es el nombre del tópico al cual se enviará el mensaje (debe coincidir con el tópico al cual se suscribió el nodo esclavo). La cadena almacenada en msg.Data es la que contiene las nuevas posiciones que tendrán los servomotores.

IV. Conclusiones

Este documento es el preámbulo en el contexto del desarrollo, de dispositivos robóticos conectados a una red y comunicados con otros nodos el cual es conocido como el internet de las cosas robóticas (IoRT); se ha podido implementar un nodo maestro haciendo uso de MATLAB, y comunicarlo a través de un middleware llamado ROS, para lograr hacer la comunicación entre el este y una Raspberry Pi, la cual es encargada a su vez de transmitir el movimiento de un brazo robótico a través de programación en Python 3, obteniendo datos desde el nodo maestro.

Cabe mencionar que este tipo de comunicación entre el nodo maestro y el nodo esclavo es vía remota, haciendo uso de ROS y un router que ayuda a la comunicación entre dispositivos, gracias a esto, es posible comunicar otros nodos a través de estos medios y así mandar información de mando desde otros lugares remotos, por lo que es posible generar un sistema de comunicaciones o en otras palabras un red de diversos nodos; y como se mencionó en un inicio es un prefacio al internet de las cosas robóticos; para una futura aplicación en este tipo de sistemas es importante tomar en cuenta la comunicación segura entre ellos, por lo que es muy posible que se vean las opciones de mandar información de manera más segura haciendo uso de criptografía ligera, diseñada especialmente para dispositivos de arquitectura computacional sencilla.

Referencias

Barrientos, A., Á lvarez, M., Hernández, J. D., del Cerro, J., & Rossi, C. (**2012**). *Modelado de cadenas cinematicas mediante matrices de desplazamiento. Una alternativa al metodo de denavit-hartenberg.RIAI - Revista Iberoamericana de Automatica e Informatica Industrial*9(4), 371–382. https://doi.org/10.1016/j.riai.2012.09.004

Galli, M., Barber, R., Garrido, S., & Moreno, L. (**2017**). *Path planning using Matlab-ROS integration applied to mobile robots.2017 IEEE International Conference on Autonomous Robot Systems and Competitions*ICARSC 2017, 98–103. https://doi.org/10.1109/ICARSC.2017.7964059

Gómez, J., & Barca, R. (**2017**). *Integración del brazo robot IRB120 en entorno ROS- Matlab*

Mazzara, M., Zhuchkov, N., Afanasyev, I., Chakraborty, S., Maksatbek, A., Yesildirek, A., Kassab, M., & Distefano, S. (**2019**). *Towards the Internet of Robotic Things: Analysis, Architecture, Components and*

Challenges Hikester-The Event Management Application View project Towards the Internet of Robotic Things: Analysis, Architecture, Components and Challenges. https://www.researchgate.net/publication/334360894

Radavelli, L., Simoni, R., de Pieri, R., Martins, D., Cardona, A., Kohan, P. H., Quinteros, R. D., & Storti, M. A. (**2012**). *Mecánica Computacional*Vol XXXI. http://www.posmec.ufsc.brhttp://www.joinville.ufsc.br/http://www.das.ufsc.br/http://www.mtm.ufsc.br://www.amcaon

'Siciliano, B., & 'Khatib, O. (**2016**). *Springer Handbook of Robotics (B. Siciliano & O. Khatib, Eds.).*Springer International Publishing. https://doi.org/10.1007/978-3-319-32552-1