# Diseño, construcción, programación y evaluación de un robot móvil omnidireccional

Design, construction, programming and evaluation of an omnidirectional mobile robot

**Daniel Ernesto Espitia Becerra**
Ing. Electrónico. Instructor
Mecatrónica
Grupo de Investigación, Innovación
y Conocimiento Aplicado de
Boyacá, GICAB
danielespitia@misena.edu.co

Servicio Nacional de Aprendizaje SENA
Regional Boyacá
Centro Industrial del
Mantenimiento y la Manufactura

## Resumen

Se presenta el proceso de diseño, construcción, programación y evaluación de un robot móvil omnidireccional. El principal objetivo de este trabajo es evaluar el funcionamiento del robot para tareas de navegación con odometría y seguimiento de pared mediante una comparación con el robot Robotino de FESTO. El robot diseñado utiliza tres motores DC con encoder, tres llantas omnidireccionales, controlador PID de motores y seis sensores ópticos de distancia. El control es implementado en Labview y las señales de los sensores y actuadores son adquiridas en un Arduino y transmitidas a través de bluetooth. Los resultados obtenidos muestran un comportamiento cercano al del robot Robotino de FESTO.

**Palabras clave:**
Robot, omnidireccional, Robotino, Labview, Arduino, Odometría

# Abstract

The process of design, construction, planning and evaluation of an omnidirectional mobile robot is presented. The main objective is to evaluate the robot operationin navigation tasks using odometry and wall following by comparison with FESTO Robotino® robot. The designed robot uses 3 DC motors with an encoder, 3 omnidirectional wheels, motors PID controllers and 6 optical distance sensors. The control is implemented in Labview and the signals from the sensors and actuators are acquired with an Arduino and transmitted via Bluetooth. The results show a behavior similar to that of the Robotino robot.


**Keywords:**
Robot, Omnidirectional, Robotino, Labview, Arduino, Odometry

## Introducción

A través de competencias como Worldskills se evidencia una gran diferencia entre regionales SENA con relación a los avances en robótica móvil: son pocas las que tienen un buen desempeño, principalmente por desconocimiento de la tecnología. Es común que el tiempo de desarrollo de un robot móvil sea muy grande, lo que hace los aprendices sólo logren desarrollar robots con funciones sencillas como robots seguidores de línea o robots sumo. Este artículo busca brindar una herramienta que facilite el proceso de creación de una plataforma de robótica móvil con altas prestaciones que puede ser adaptada a diversas aplicaciones.

En el documento se muestran los resultados parciales del proyecto de investigación, se espera adicionarle al robot nuevos elementos como un brazo manipulador, sensor laser para funciones de mapeo, implementación del Sistema Operativo de Robots ROS, desarrollo de material didáctico para enseñanza de robótica, entre otros.

## 1. Marco Teórico

### 1.1 Robot Omnidireccional

La principal característica de los robots omnidireccionales es que pueden moverse libremente en cualquier dirección sin necesidad de reorientarse, es decir tienen máxima maniobrabilidad en el plano.(Andaluz Ortiz, 2011).

Se pueden citar ejemplos de distintos tipos de robots omnidireccionales: de 3 llantas (Martínez & Sisto), de 4 llantas (Rojas & Förster, 2006), con llantas Mecanum (Salih, Rizon, Yaacob, Adom, & Mamat, 2006), con ruedas convencionales (Wada & Mori, 1996), entre otros.

## 1.2 Modelo Cinemático del Robot Omnidireccional

El robot diseñado consta de tres llantas omnidireccionales ubicadas a un ángulo de 120º entre una y otra. El modelo cinemático de este robot puede ser representado por la ecuación (1) (Kalmár-Nagy 2002).

$$\begin{pmatrix} V1 \\ V2 \\ V3 \end{pmatrix} = \begin{pmatrix} -\sin\Theta & -\cos\Theta & L \\ -\sin(\pi/3-\Theta) & -\cos(\pi/3-\Theta) & L \\ \sin(\pi/3+\Theta) & -\cos(\pi/3+\Theta) & L \end{pmatrix} \begin{pmatrix} Vx \\ Vy \\ W \end{pmatrix} \quad (1)$$

Donde θ es el ángulo de rotación del robot en el sentido contrario de las manecillas del reloj respecto a un eje de coordenadas global en el piso. Asumiendo que el eje de coordenadas (x, y) se encuentra en el centro del robot (coordenadas locales), como se observa en la Figura 1, se puede decir que:

$$\Theta = 60 \quad (2)$$

Por lo que la ecuación (1) se reduce a

$$\begin{pmatrix} v1 \\ v2 \\ v3 \end{pmatrix} = \begin{pmatrix} -\sqrt{3}/2 & 1/2 & L \\ 0 & -1 & L \\ \sqrt{3}/2 & 1/2 & L \end{pmatrix} \begin{pmatrix} vx \\ vy \\ v \end{pmatrix} \quad (3)$$

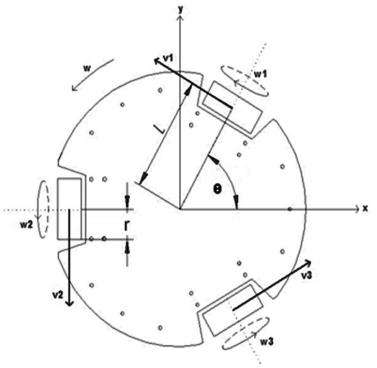



***Figura 1.*** Geometría del robot omnidireccional de tres llantas

Las ecuaciones de omniaccionamiento (4), (5), (6) permiten calcular las velocidades individuales de cada motor a partir de las velocidades deseadas de movimiento del robot V X, V Y y W.

$$w1 = \left[ -vx \cdot \frac{\sqrt{3}}{2} + \frac{1}{2} \cdot vy + (L \cdot \omega \cdot \frac{\pi}{180}) \right] \cdot k \quad (4)$$

$$w2 = \left[ -vx + (L \cdot \omega \cdot \frac{\pi}{180}) \right] \cdot K \quad (5)$$

$$w3 = \left[ -vx \cdot \frac{\sqrt{3}}{2} + \frac{1}{2} \cdot vy + (L \cdot \omega \cdot \frac{\pi}{180}) \right] \cdot K \quad (6)$$

k es la constante para convertir la velocidad de mm/s a rpm

$$k = \frac{60 \cdot n}{2\pi r} \quad (7)$$

Donde W1, W2 y W3 son las velocidades de los motores 1,2 y 3 (rpm), L es la distancia del centro del robot a una llanta (mm), N es el factor de reducción de la caja reductora, R es el radio de la llanta (mm), V X es la velocidad de desplazamiento del robot en el eje X (mm/s), V Y es la velocidad de desplazamiento del robot en el eje Y (mm/s), Wes la velocidad de giro del robot respecto al eje Z (rad/s), y mer, es el número de pulsos del encoder por vuelta.

Para que las velocidades w1, w2 y w3 puedan ser enviadas a los controladores de motores, deben ser primero convertidas a pulsos por segundo (PPS) por medio de las ecuaciones (8), (9) y (10).

$$w1_{pps} = w1 \cdot \frac{mer}{60} \quad (8)$$

$$w2_{pps} = w2 \cdot \frac{mer}{60} \quad (9)$$

$$w3_{pps} = w3 \cdot \frac{mer}{60} \quad (10)$$

## 1. 3 Odometría

La odometría es el método de navegación más usado para el posicionamiento de robots ya que tiene buena precisión a corto plazo, es económica y permite altas frecuencias de muestreo. (Borenstein, Everett, & Feng, 1996). Por medio de la odometría se puede calcular la posición (x,y, θ) del robot respecto a un punto de referencia en el piso (Figura 2).

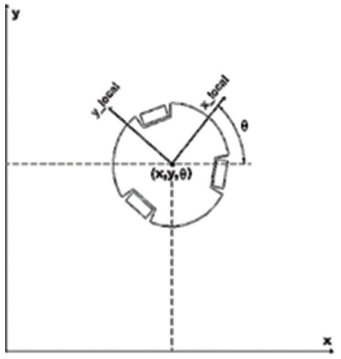



***Figura 2.*** Posición del robot respecto a un punto de referencia global

Para el cálculo de la odometría del robot es necesario calcular la velocidad real de los motores a partir de los pulsos de los encoders en un lapso de tiempo. En este caso puntual, la velocidad es calculada directamente por los controladores de motores, los cuales entregan la velocidad de cada motor en pulsos por segundo.

Para convertir la velocidad de los motores de pulsos por segundo (PPS) a revoluciones por minuto (rpm) se utilizan las ecuaciones (11), (12) y (13).

$$w1 = \frac{w1\,(pps) * 60}{mer} \quad (11)$$

$$w2 = \frac{w2\,(pps) * 60}{mer} \quad (12)$$

$$w3 = \frac{w3\,(pps) * 60}{mer} \quad (13)$$

A continuación se calcula las velocidades reales vx, vy y w del robot utilizando las ecuaciones de omniaccionamiento inverso (14), (15) y (16).

$$vx = \frac{1}{\sqrt{3} \cdot K} \cdot (w3 - w1) \quad (14)$$

$$vy = \frac{2}{3 \cdot K} \cdot \left( w1 + \left( \frac{w3 - w1}{2} \right) - w2 \right) \quad (15)$$

$$w = \left( vy + \frac{w2}{K} \right) \cdot \left( \frac{180}{\pi \cdot L} \right) \quad (16)$$

Ahora, se calcula la distancia recorrida en el último lapso de tiempo utilizando las ecuaciones (17), (18) y (19).

$$\Delta x_{local} = vx \cdot \Delta t \quad (17)$$

$$\Delta x_{local} = vy \cdot \Delta t \quad (18)$$

$$\Delta \Theta_{local} = w \cdot \Delta t \cdot \frac{\pi}{180} \quad (19)$$

El ángulo de giro del robot se puede calcular sumando el ángulo anterior al nuevo cambio de ángulo (20).

$$\Theta = \Theta + \Delta\Theta \ \frac{\pi}{180} \quad (20)$$

Las distancias en sistema de coordenadas local (respecto al centro del robot) deben ser transformadas a sistema de coordenadas global (respecto a un punto fijo en el piso) utilizando las ecuaciones (21) y (22).

$$\Delta\theta_{global} = \cos(\theta) \cdot \Delta t_{local} - \sin(\theta) \cdot \Delta y_{local} \quad (21)$$

$$\Delta\theta_{global} = \sin(\theta) \cdot \Delta t_{local} - \cos(\theta) \cdot \Delta y_{local} \quad (22)$$

Finalmente, se calcula la posición x y y del robot con las ecuaciones (23) y (24).

$$x = x + \Delta x_{global} \quad (23)$$

$$y = y + \Delta y_{global} \quad (24)$$

## 1.4 Control de Posición

El control de posición tiene como objetivo mover el robot dentro de un entorno hasta una posición deseada (x,y, θ). Para el control de posición del robot se utilizó un controlador proporcional (25),(26) y (27).

$$vx_{global} = Kp_x \cdot (x_{setpoint} - x) \quad (25)$$

$$vy_{global} = Kp_y \cdot (y_{setpoint} - y) \quad (26)$$

$$w = Kp_w \cdot (\theta_{setpoint} - \theta) \quad (27)$$

Donde x _setpoint es la posición deseada en el eje x, y_setpoint es la posición deseada en el eje y, θ_setpoint es el ángulo de giro deseado, Kpx es la constante proporcional para el movimiento en el eje x, Kpy es la constante proporcional para el movimiento en el eje y, Kpw es la constante proporcional para el giro del robot, y (x,y, θ) son las posiciones reales del robot entregadas por la odometría.

A continuación es necesario transformar del sistema de coordenadas global al sistema de coordenadas local con las ecuaciones (28) y (29).

$$vx = \cos(-\theta) \cdot vx_{global} - \sin(-\theta) \cdot vy_{global} \quad (28)$$

$$vy = \sin(-\theta) \cdot vx_{global} - \cos(-\theta) \cdot vy_{global} \quad (29)$$

Para que las velocidades vx, vy y w puedan ser aplicadas al robot se utilizan las ecuaciones de omniaccionamiento (4), (5), (6) para calcular las velocidades individuales de cada motor y luego se convierten a pulsos por segundo con las ecuaciones (8), (9), (10) para ser enviadas al controlador de motores.

### 1.5 Seguimiento de Pared

El robot cuenta con 6 sensores ópticos de distancia Sharp GP2D120 con una separación de 60 grados entre ellos, los cuales le permiten al robot conocer la distancia respecto a paredes y obstáculos del entorno.

El sensor GP2D120 tiene un rango de funcionamiento entre 4 y 30 cm, entrega una señal análoga de voltaje, pero como podemos apreciar en la Figura 3, su respuesta no es lineal.

Para el seguimiento de pared se implementó un control proporcional de distancia a la pared (30) y un control proporcional de giro respecto a la pared (31). El movimiento lateral es constante (32) y determina la velocidad de seguimiento de la pared.

$$vx = -Kp_3 \cdot (d_{setpoint} - S0) \quad (30)$$

$$w = Kp_4 \cdot (S0 - S1) \quad (31)$$

$$vy = \text{constante} \quad (32)$$

Donde Kp3 es la constante proporcional para el control de distancia a la pared, Kp4 es la constante proporcional para el control de giro respecto a la pared, d_setpoint es la distancia deseada del robot a la pared, S0 es la distancia detectada por el sensor óptico 0 (cm), y S1 es la distancia detectada por el sensor óptico 1 (cm).

# 2. Materiales y Métodos

## 2.1 Selección de Componentes

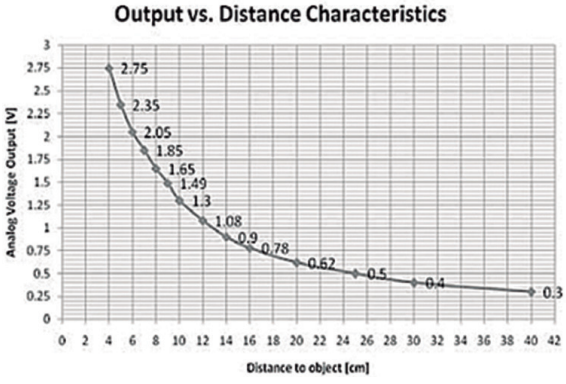
Para la selección de elementos del robot se realizó un análisis del estado del arte comparando robots comerciales existentes, lo cual permitió elegir el modelo omnidireccional de tres ruedas debido a su facilidad para desplazarse en cualquier dirección y a que no requiere de suspensión. Se eligió 500mm/s como la velocidad máxima deseada de desplazamiento del robot.

### 2.1.1 Motores

Se seleccionaron los motores Pololu 37D 150 rpm teniendo en cuenta que tienen la velocidad y el torque requerido para el robot, además cuentan con un encoder de cuadratura de 64 pulsos por vuelta necesario para realizar el control de velocidad del motor.

### 2.1.2 Llantas

Se optó por las llantas omnidireccionales VEX IQ de 200mm de perímetro, la razón, los rodillos proporcionan muy buen




Fuente: (Lung, Sabou, Orha, & Buchman, 2010)

***Figura 3.*** Curva de voltaje de salida vs distancia del GP2D120

agarre evitando deslizamiento. Además, su radio es de solo 31.8mm permite obtener la velocidad máxima deseada para el robot, en combinación con los motores elegidos.

### 2.1.3 Controlador de Motores

Se seleccionó el controlador de motores Roboclaw 2x5A debido a que permiten manejar dos motores a una corriente máxima de 5A, tienen entradas para encoders de cuadratura, realizan control PID de velocidad y posición, y tienen protecciones contra batería baja y sobre corriente.

El roboclaw realiza control PID de velocidad del motor, donde la variable a controlar son los pulsos por segundo provenientes del encoder de cuadratura del motor. El setpoint y las constantes proporcional, integral y derivativa son enviadas por comunicación serial desde el Arduino al roboclaw.

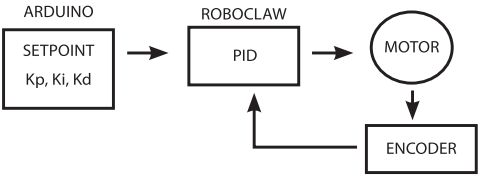



***Figura 4.*** Diagrama de bloques del sistema de control de velocidad

El controlador se sintonizó manualmente siguiendo el procedimiento de calibración del Manual de Usuario del Roboclaw que indica:

**1.** Determinar el valor de pulsos por segundo del motor a máxima velocidad QPPS.

**2.** Colocar las constantes integral y derivativa en cero e iniciar a incrementar la constante proporcional hasta obtener un valor cercano al setpoint. Si se producen vibraciones en el motor, establecer la constante proporcional a 2/3 de ese valor.

**3.** Incrementar el valor de la constante integral hasta obtener una buena respuesta sin vibraciones.

**4.** Usar la constante derivativa solamente si no es posible obtener un buen control con el controlador PI.

### 2.1.4 Controlador

Se eligió Arduino Mega 2550 para realizar las funciones de comunicación con el controlador de motores, lectura de los sensores de distancia, posibilidad de conectar nuevos sensores como giroscopio, sensor óptico de línea, entre otros. Adicionalmente se seleccionó el módulo Mega Sensor Shield para facilitar la conexión de sensores y actuadores al Arduino.

### 2.1.5 Sensores

Se seleccionaron los sensores ópticos infrarrojos Sharp GP2D120 para detección de paredes y obstáculos entre 4 y 30cm.

### 2.1.6 Comunicación

Para la comunicación inalámbrica con el robot se optó por la tecnología bluetooth. Se eligió el módulo Bluesnirf silver por su facilidad de conexión e integración, además permite realizar comunicaciones hasta de 18m de distancia lo cual es suficiente para nuestra aplicación ya que tiene como objeto comunicar el arduino a un computador portátil con Labview de manera inalámbrica.

### 2.1.7 Baterías

Se seleccionó un pack de baterías recargables de iones de litio de 14.8V 2200mAh.

## 2.2 Diseño Mecánico

Para el diseño mecánico del robot se utilizó el software de diseño asistido por computador Solidedge. Se modelaron los componentes seleccionados y se realizó el diseño estructural del robot.

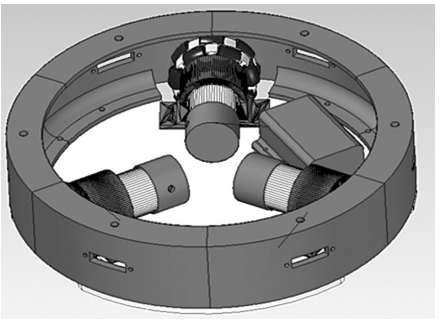


***Figura 5.*** Modelo 3D del robot omnidireccional

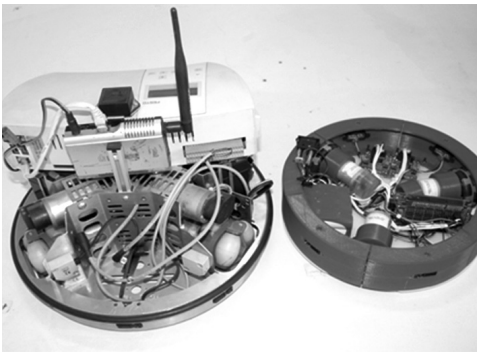


***Figura 6.*** Robotino y Robot Omnidireccional construidos

### 2.3 Construcción

Para la construcción del robot se utilizó una máquina de corte laser para partir las láminas superior e inferior de robot en material acrílico.

Se utilizó una impresora 3D para fabricar las piezas laterales, los soportes de los motores, las tapas de los encoder, el soporte de la batería, utilizando plástico ABS.

### 2.4 Programación

Para la programación del robot se eligió el software Labview de National Instruments. Se utilizó la librería LIFA (Labview Inter face for Arduino) para comunicar inalámbricamente el Labview con el Arduino. Se programaron funciones personalizadas en el Arduino en lenguaje c++ para el manejo del controlador de motores roboclaw desde labview.

Se implementaron los algoritmos para navegación utilizando odometría, posicionamiento y seguimiento de paredes en el software Labview.

### 2.4.1 Omniaccionamiento

En la Figura 7 se aprecia la programación del algoritmo de omni - accionamiento para el robot.


***Figura 7.*** Algoritmo de omniaccionamiento implementado en Labview

### 2.4.2 Omniaccionamiento Inverso

En la Figura 8 se muestra la programación del algoritmo de omni-accionamiento inverso para el robot.

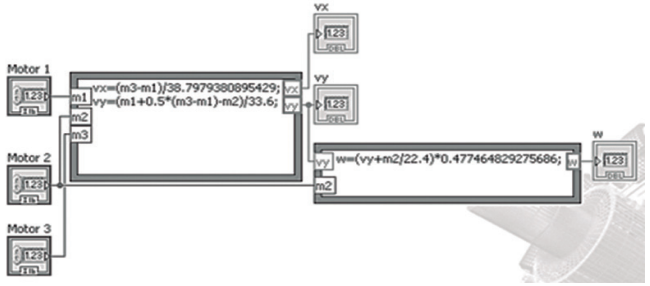



***Figura 8.*** Algoritmo de omniaccionamiento inverso implementado en Labview

### 2.4.3 Odometría

En la Figura 9 se evidencia la programación de la odometría para el robot. El flujo de señales para el cálculo de la odometría se muestra en la Figura 10.

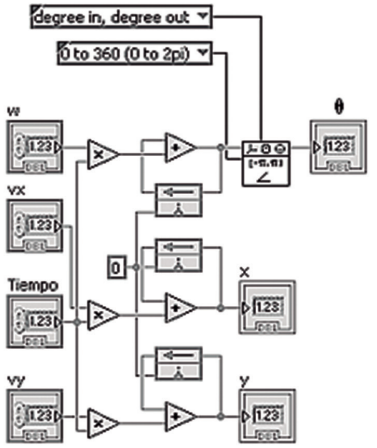



***Figura 9.*** Algoritmo de odometría implementado en Labview

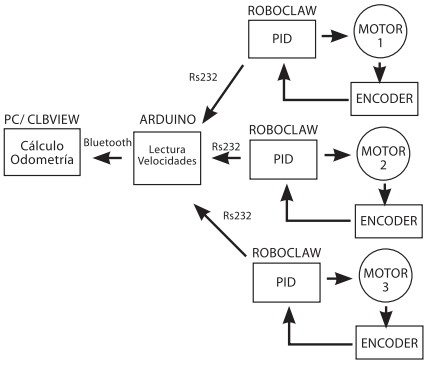



***Figura 10.*** Diagrama de bloques con el flujo de señales del robot para el cálculo de la odometría

### 2.4.4 Linealización de Sensores GP2D120

La señal del sensor GP2D120 fue linealizada y convertida a centímetros utilizando las funciones "convert voltaje" y "Query calibration parameters" incluidas en el toolkit de robótica de Labview (Figura 11).

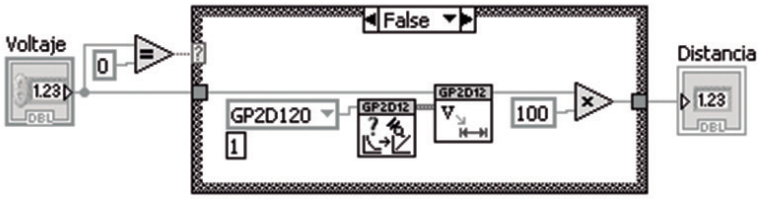



***Figura 11.*** Linealización del sensor GP2D120

### 2.4.5 Seguimiento de pared

Se implementó el algoritmo de seguimiento de pared en Labview (Figura 12).

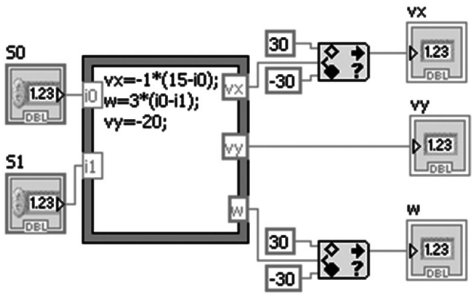



***Figura 12***. Algoritmo de seguimiento de pared implementado en Labview

En la Figura 13 se aprecia un diagrama de bloques con el flujo de señales del robot para el seguimiento de pared.

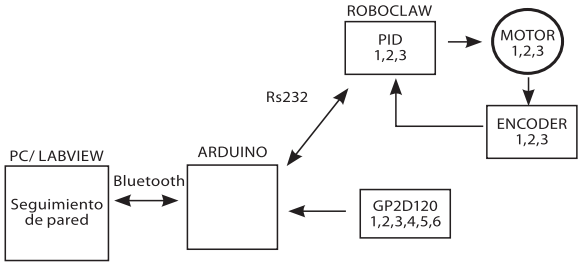



***Figura 13.*** Diagrama de bloques con el flujo de señales del robot para el seguimiento de pared

### 2.5 Evaluación

El prototipo construido se evaluó para tareas de navegación, posicionamiento respecto a paredes y seguimiento de paredes. Se realizaron las mismas pruebas con el robot comercial Robotino con el objetivo de comparar los resultados.

Las pruebas se ejecutaron en una pista de robótica de 2m x 2m con paredes de 30cm de alto, en madera color blanco, dentro de un ambiente cerrado con iluminación artificial.

Para ejecutación de datos se instaló un marcador en el robot de forma tal que la trayectoria quedó registrada para su posterior medición utilizando un flexómetro y un graduador.

## 3. Resultados

### 3.1 Control Pid de Velocidad de los Motores

En la Figura 14 se aprecia el comportamiento de la velocidad del motor 1 ante un setpoint de 5000 PPS, obtenido después de realizar la sintonización manual.

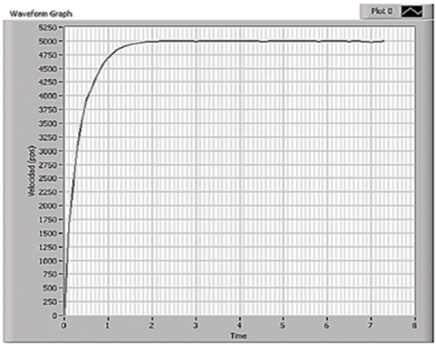



***Figura 14.*** Gráfica de velocidad del motor 1 vs tiempo

### 3.2 Navegación en Línea Recta

La prueba consistió en mover el robot hacia adelante en línea recta una distancia de 1 metro, utilizando odometría.

Como se aprecia en los resultados de la Tabla 1 y las Figuras 15 y 16, el robot construido tiene un error menor que el Robotino para el desplazamiento de 1 metro. Esto se presenta posiblemente por problemas de calibración en la odometría del robotino.

La trayectoria seguida por el Robotino es mucho más recta que la del robot construido, lo cual se asocia al algoritmo de control utilizado para el seguimiento de la trayectoria y a que el Robotino es más insensible a los defectos e imperfecciones de la pista.

***Tabla 1.*** Datos obtenidos en la evaluación de navegación en línea recta

| Intento | Robot | Robot | Error (mm) |
|---|---|---|---|
| 1 | Construido | Construido | 3 |
| 2 | Construido | Construido | -8 |
| 3 | Construido | Construido | 25 |
| 1 | Robotino | Robotino | 33 |
| 2 | Robotino | Robotino | 15 |
| 3 | Robotino | Robotino | 20 |


***Figura 15.*** Prueba de desplazamiento en línea recta del robot construido

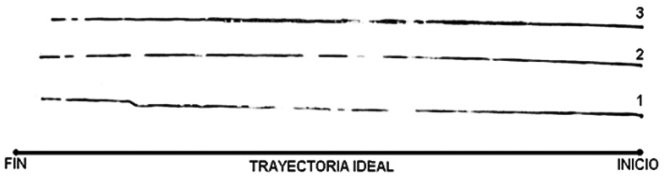



***Figura 16.*** Prueba de desplazamiento en línea recta del Robotino

### 3.3 Navegación en Trayectoria Cuadrada

Esta prueba consistió en mover el robot siguiendo una trayectoria cuadrada de 500mm de lado hasta llegar a la posición inicial.

Como se observa en la Tabla 2 y Figuras 17 y 18, el error en la posición del robot construido es mucho mayor que la del robotino, debido a un error en el ángulo del robot durante el último segmento. Cualquier pequeño error en el ángulo del robot se verá reflejado en una gran desviación en la posición final. El robot construido es mucho más sensible a imperfecciones de la pista debido a su bajo peso y menor tamaño.

***Tabla 2.*** Error en navegación en trayectoria cuadrada

| Robot | Error (mm) (Distancia entre Inicio y Fin) | ErrorÁngulo (°) Ángulo inicio - Ángulo fin |
|---|---|---|
| Construido | 95 | 7 |
| Robotino | 12 | 0 |

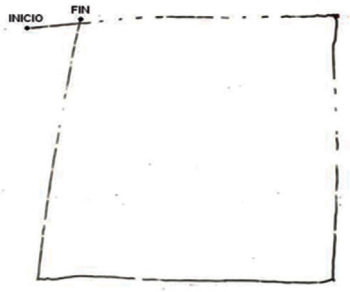



***Figura 17.*** Prueba de desplazamiento en trayectoria cuadrada con robot construido

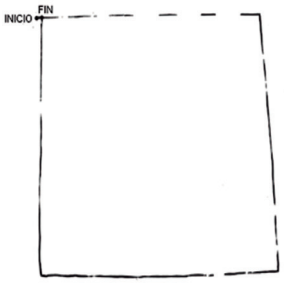



***Figura 18.*** Prueba de desplazamiento en trayectoria cuadrada con Robotino

### 3.4 Seguimiento de Pared

Para esta prueba se ubicó el robot a 50 cm de frente a la pared y se realizó seguimiento de pared hacia la derecha. La distancia ente la pared y el robot para el seguimiento fue de 15 cm.

Como se observa en las Figuras 19 y 20, juntos robots realizan correctamente el seguimiento de la pared siendo su comportamiento diferente debido a la sintonización de los controladores proporcionales de distancia. El robot construido llega más rápido a la distancia deseada y realiza sobrepaso, a diferencia del robotino, el cual no sobrepasa la distancia deseada.

## 4. Recomendaciones

Este artículo muestra los resultados parciales del proyecto de investigación, por lo que se recomienda mejorar la sintonización de los controladores de posición y seguimiento de pared para obtener una respuesta mucho más estable.

Es importante la inclusión de un giróscopio para reducir los errores de la odometría en el cálculo del ángulo de giro del robot, debido a que al utilizar encoders se acumulan a lo largo

del tiempo. Se recomienda utilizar filtro Kalman para la fusión de los datos del encoder y el giroscopio.

En el control de los motores es importante utilizar rampas de aceleración y desaceleración ya que al realizar cambios bruscos en la velocidad de las llantas se produce deslizamiento generando errores en la posición del robot.

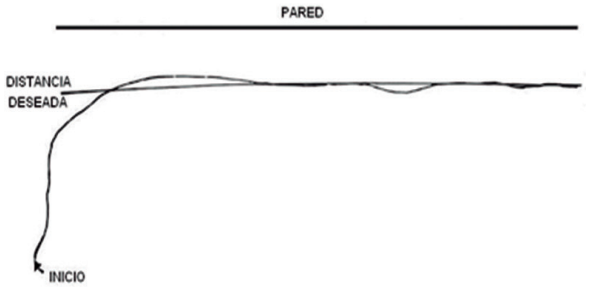



***Figura 19.*** Seguimiento de pared con robot construido

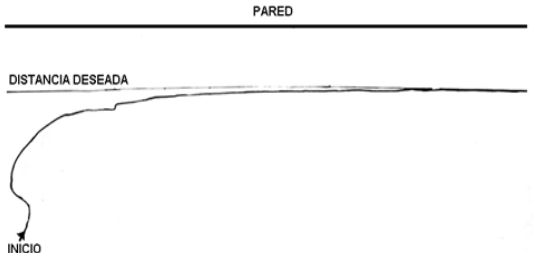



***Figura 20.*** Seguimiento de pared con Robotino

## Bibliografía

Andaluz Ortiz, G. M. (2011). Modelación, Identificación y Control de Robots Móviles. QUITO/ EPN/2011.

Borenstein, J., Everett, H., & Feng, L. (1996). Where am I? Sensors and methods for mobile robot positioning. University of Michigan, 119(120), 27.

Lung, C., Sabou, S., Orha, I., & Buchman, A. (2010). FPGA implementation of a linearization algorithm for Sharp GP2D120

range sensor. Carpathian Journal of Electronic and Computer Engineering, 3, 57.

Martínez, S., & Sisto, R. Diseño Mecánico de un Robot Omnidireccional.

Rojas, R., & Förster, A. G. (2006). Holonomic control of a robot with an omnidirectional drive. KI- Künstliche Intelligenz, 20(2), 12-17.

Salih, J. E. M., Rizon, M., Yaacob, S., Adom, A. H., & Mamat, M. R. (2006). Designing omni-directional mobile robot with mecanum wheel. American Journal of Applied Sciences, 3(5), 1831-1835.

Wada,M.,&Mori,S.(1996). Holonomic and omnidirectional vehicle with conventional tires. Paper presented at the Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on.

Kalmár-Nagy, Tamás, Pritam Ganguly and Raffaello D.Andrea (2002). Real-time trajectory generation for omnidirectional vehicles. Proceedings of the American Control Conference pp. 286–291.