Robot Oruga detector y esquivador de obstáculos con *Fuzzy Logic I.A*

Andrés Rodríguez [1], Jehiver Cortés [2]

(1) andfelrodriguez@uniboyaca.edu.co
(2) jjcortes@uniboyaca.edu.co

Facultad de Ciencias e Ingeniería
Universidad de Boyacá
Tunja, Colombia

Robot Oruga detector y esquivador de obstáculos con *Fuzzy Logic I.A*

Palabras clave:
Robot Oruga; Inteligencia Artificial, Fuzzy Logic.

RESUMEN

El presente artículo hace referencia a la programación y construcción de un robot tipo oruga que detecta y esquiva los obstáculos en un entorno plano. Se usó un método de inteligencia artificial llamado Fuzzy Logic para el control de los motores y de los sensores de proximidad.

El robot se ha nombrado Rover 5 y nace como una alternativa de proyecto aplicado a los conocimientos adquiridos sobre robótica con múltiples aplicaciones en los diferentes ámbitos sociales, tecnológicos y científicos. Cuenta con una programación de lógica difusa y adaptación de un módulo Arduino que facilita la comunicación y el desarrollo del objetivo de este prototipo.

La lógica difusa es un sistema matemático que modela funciones no lineales, convirtiendo unas entradas en salidas acordes con los planteamientos lógicos que se usan el razonamiento aproximado.

I. INTRODUCCIÓN

El proyecto fue realizado para poner en práctica los conocimientos de robótica buscando la autonomía del prototipo, con el propósito de facilitar las labores de ejecución del robot basado en una guía lógica de recepción, toma de decisiones y respuesta de las mismas.

La superficie donde se lleva a cabo la ejecución del trabajo objetivo del prototipo es en un ambiente plano, debido a la configuración de lógica difusa del Rover 5. Los obstáculos son detectados por medio de sensores ultrasónicos, que envía señales análogas al Arduino quien las analiza y procede a emitir las respuestas pertinentes mediante señales PWM de acuerdo con la programación, para variar el sentido de giro de los motores con el propósito de que el Rover 5 no colisione con ellos, esquivándolos bordeando el objeto.

II. MARCO TEÓRICO

A. Robot Oruga

El robot oruga consta de dos llantas, cada una con su eje y un motor unido a una banda paralela que se une a dos llantas locas.

En esta configuración, el deslizamiento en los giros es muy grande, se pierde la exactitud de los cálculos y la posición del robot. Estos robots son teleoperados y se emplean en lugares donde el piso es irregular.

Fig1. Robot Oruga Teleoperado.

B. Inteligencia Artificial

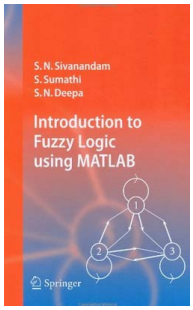
La inteligencia artificial (IA) es un área multidisciplinaria, que a través de ciencias como la computación, la lógica y la filosofía, estudia la creación y diseño de entidades capaces de resolver cuestiones por sí mismas utilizando como paradigma la inteligencia humana.

Fig. 2. Robot I.A.

C. Fuzzy Logic

La lógica difusa es uno de los métodos de la inteligencia artificial, esta nos ayuda a generar un control respecto a los problemas del entorno por eso se dice que, el fuzzy logic se basa en grados de pertenencia para obtener respuesta a varios problemas generados.

Fig.3. Introducción al fuzzy logic usando MATLAB.



III. PROCEDIMIENTO

TABLA I. MATERIALES PARA LA CONSTRUCCIÓN DEL PROYECTO.

N°	*MATERIALES*
1	32 Cables Macho y Hembra.
2	1 Caucho Termo-incogible.
3	9 Palos de Paleta.
4	1 Cinta aislante.
5	1 conector universal.
6	1 Base para batería
7	1 Mini protoboard.
8	1 etapa de Potencia con un L298.
9	2 Baterías de 3.7 Vdc a 3000 mA.
10	1 Batería Cuadrada de 9 Vdc.
11	1 Plataforma Rover 5.
12	1 Microcontrolador Arduino Mega 2560.
13	1 Micro Servo Motor 1Kg.
14	3 Sensores Sharp de 20 a 150 cm.

A. Programación

Los programas usados para la parte de simulación y puesta a prueba fueron MATLAB Y ARDUINO.

En este proceso se usó un toolbox del software MATLAB llamado FUZZY LOGIC.

Se hizo una prueba de escritorio en Excel donde se hacen los siguientes pasos:

1. FUZIFICACIÓN:

a) Se crean las variables de entrada y salida.

b) A cada variable se le da un rango de medición.

2. REGLAS DE SINTAXIS

a) Se crea una tabla donde las variables de entrada se comparan ellas mismas y sus valores son las variables de salida.

3. DESFUZIFICACIÓN:

a) En este punto se toma una de dos alternativas, debido a que se deben analizar los puntos bajos de las gráficas. Las dos alternativas son o tomar el máximo que es el de Mandani o el mínimo que es el de Segeno. En este caso por optimizar la inteligencia artificial se escogió la de Mandani.

Esta prueba de escritorio se simulo en MATLAB. Todos los pasos mencionados anteriormente son los mismos que el toolbox de Fuzzy Logic hacen pero de la siguiente forma:

Fig. 6. Editor del Toolbox de MATLAB Fuzzy Logic.

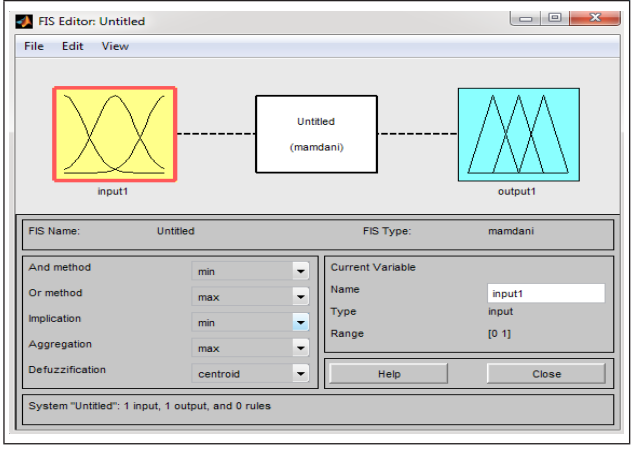
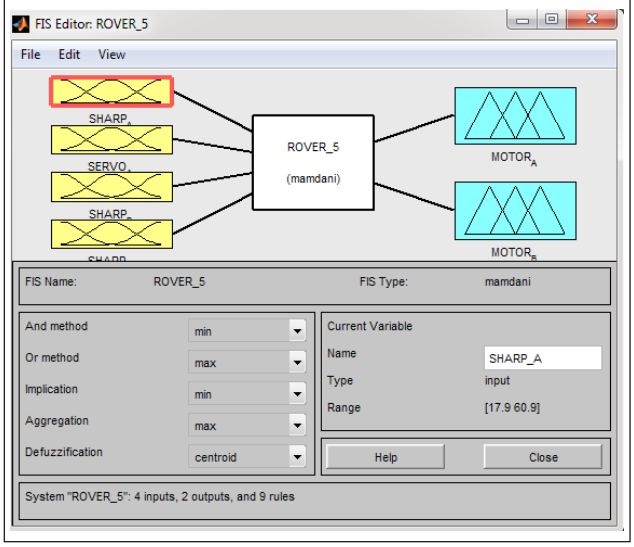


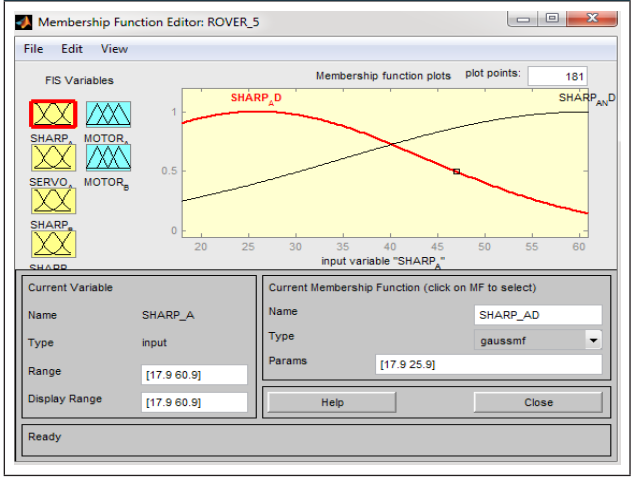
Fig. 7. Fuzzy Logic para el Rover 5.



En la figura 7 se puede observar en la parte izquierda las entradas del sistema de control y a su derecha las variables de salida.

4. FUZIFICACIÓN SOFTWARE

Fig. 8. Fuzzy Logic Variable de entrada.



Como se puede observar en la figura 8 antes de crearla aparece un mensaje de dialogo que pregunta si se desea crear una entrada y al escoger la ventana pregunta si desea hacer una curva gaussiana, triangular, etc. En este caso se escogió gaussiana.

TABLA II. REGLAS DE SINTAXIS.

	SHARPBD	SHARPBND	SHARPCD	SHARPCND
SHARPAD SERVOMOTORA	MOTORABDI	MOTORABDI	MOTORABDD	MOTORABDI
SHARPAD SERVOMOTORA	MOTORABDD	MOTORABDD	MOTORABDD	MOTORABDD
SHARPAND	MOTORABDD	MOTORABDA	MOTORABDI	MOTORABDA

Esta tabla nos permite observar con claridad las decisiones que el robot puede tomar de acuerdo con lo que nuestros sensores detecten. Lo que significa cada palabra en nuestra tabla es:

SHARPAD SERBOMOTORAD: sensor A con el servomotor en la dirección de derecha detecto algo.

SHARPAD SERVOMOTORAI: sensor A con el servomotor en la dirección de izquierda detecto algo.

SHARPAND: sensor A no detecto algo.

SHARPBD: sensor B detecto algo.

SHARPBND: sensor B no detento algo

SHARPCD: sensor C detento algo.

SHARPCND: sensor C no detento algo.

MOTORABDI: motores toman dirección de izquierda.

MOTORABDD: motores toman dirección de derecha

MOTORABDA: motores seguir de frente.

El sensor A esta posicionado al frente del motor, sujeto al servomotor.

El sensor B está posicionado en la esquina izquierda del robot.

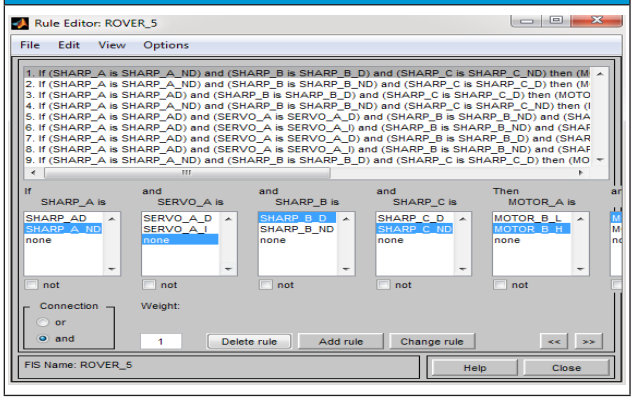
El sensor C está posicionado en la esquina derecha del robot.

Un ejemplo claro de las decisiones que puede tomar el robot es si el sensor B y sensor A junto con el servomotor con giro a la derecha detectan, la decisión que debe tomar el robot es dirigirse hacia la izquierda. Gracias a este cuadro se pueden ver claramente la programación y la inteligencia que tiene el robot para tomar decisiones y en este caso evitar los obstáculos.

5. REGLAS DE SINTAXIS SOFTWARE

Para esta parte cuando se hace click en crear rules o reglas, el toolbox nos lleva a una ventana donde se le da click en variables y el software da la opción de crear los condicionales. En la siguiente figura 9 se puede observar lo planteado.

Fig. 9. Fuzzy Logic Reglas de sintaxis.



Para poder visualizar en una gráfica 3D, el comportamiento de las variables de entrada respecto a las de salida se da click en la pestaña Surface, esta ventana nos da la posibilidad de conocer los puntos buenos y críticos del sistema de inteligencia artificial.

Fig. 10. Fuzzy Logic Visualizador 3D del comportamiento de sensor y el servomotor respecto los motores.

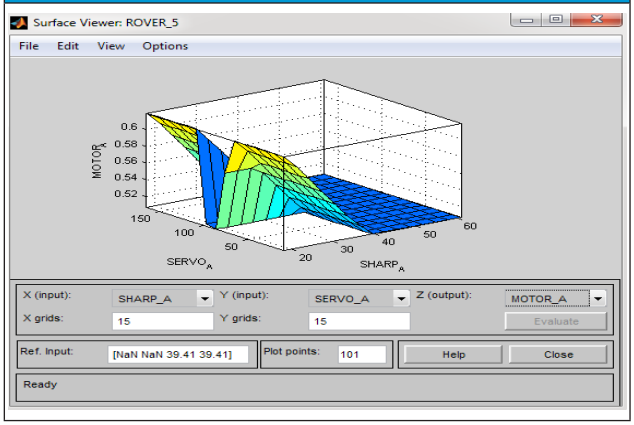


Fig. 11. Fuzzy Logic Visualizador 3D del comportamiento del sensor A y el sensor B respecto los motores.

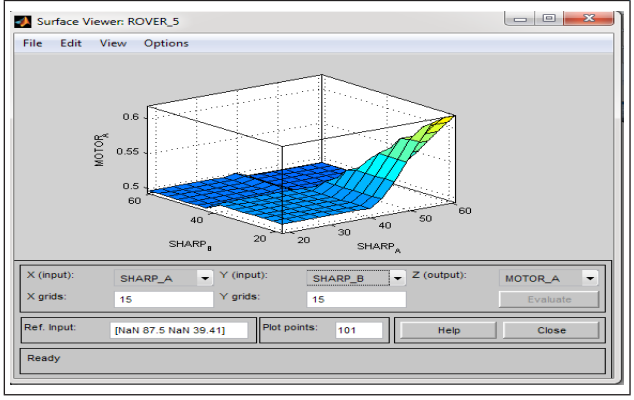
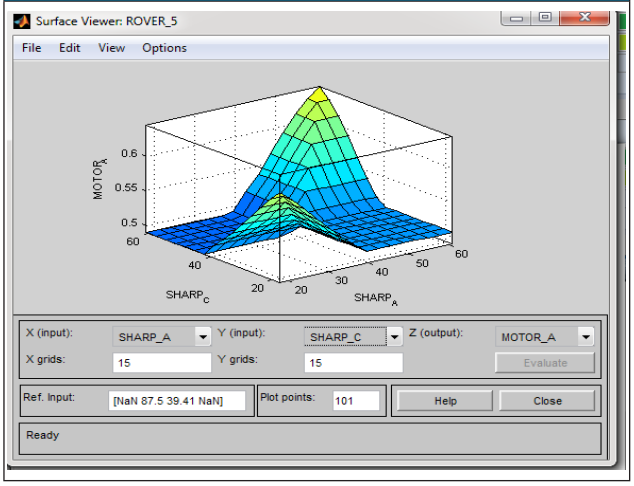


Fig.12. Fuzzy Logic Visualizador 3D del comportamiento del sensor A y el sensor C respecto los motores.

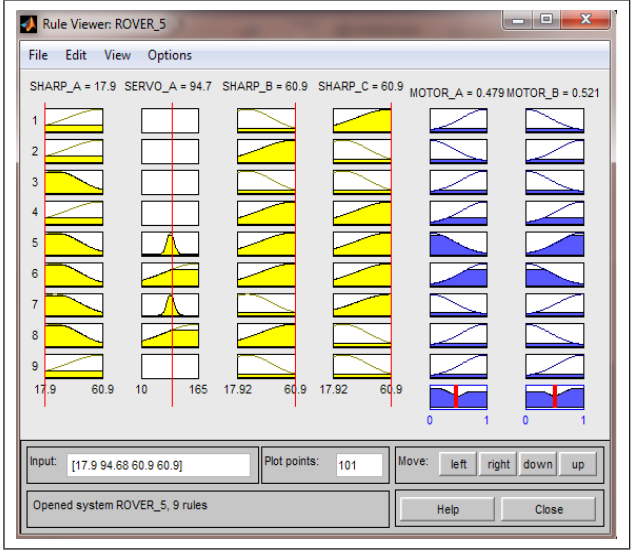


En la figura 12 se puede observar claramente la detección de los sensores a 45°.

6. DESFUZIFICACIÓN Y SIMULACIÓN SOFTWARE

En la simulación se puede ver el comportamiento de las reglas de sintaxis. Por ejemplo, en la siguiente figura 13 se puede observar el comportamiento de los parámetros de entrada respecto a los de salida donde se visualiza en el grafico azul una franja roja que indica la dirección de giro de los motores.

Fig.13. Fuzzy Logic Simulación.



En la figura 13 se puede visualizar que los sensores A y B están detectando hacia la derecha un objeto es decir que los motores deben girar en sentidos contrarios para que este se vaya hacia la izquierda y no choque contra el obstáculo.

Fig.14. Fuzzy Logic Simulación Rover 5.

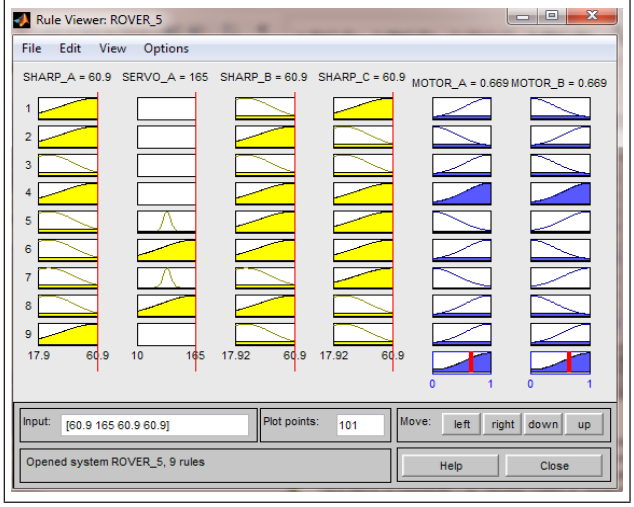
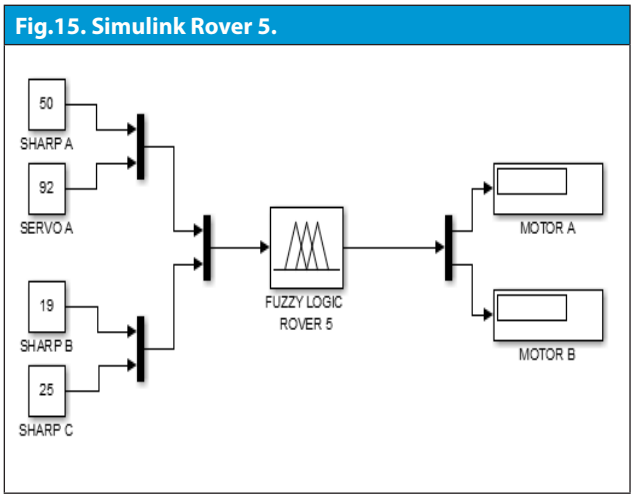


Fig.15. Simulink Rover 5.



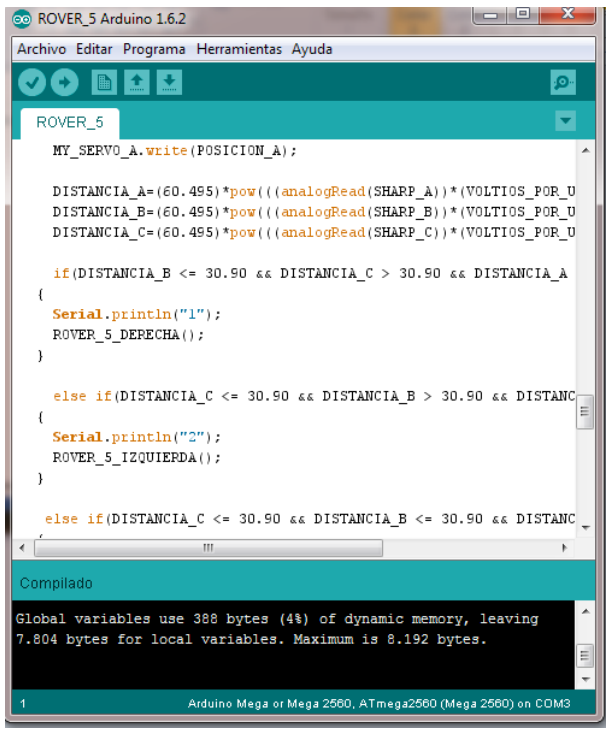
En la figura 14 se puede observar que todos los sensores no están detectando el objeto es decir que en el grafico azul con la franja roja dice que los motores deben ir en el mismo sentido o hacia adelante.

Se desarrolló un pequeño programa en simulink donde se observa el comportamiento de las variables de salida.

Ya teniendo conocimiento de cómo se genera un Fuzzy logic y corroborando de que la simulación quedo bien. Se empezó hacer la programación en Arduino.

Fig.16. Variables declaradas en Arduino.

Fig 17. Rover 5 Programación Arduino.



C. Construcción del robot

Las figuras 18 y 19 representan la plataforma Rover 5 donde posee una tabla de 25x25 en su parte superior que fue usada como base de los sensores, actuador, etapa de potencia, microcontrolador y baterías.

Fig.18. Robot Oruga Perfil Frontal Lateral

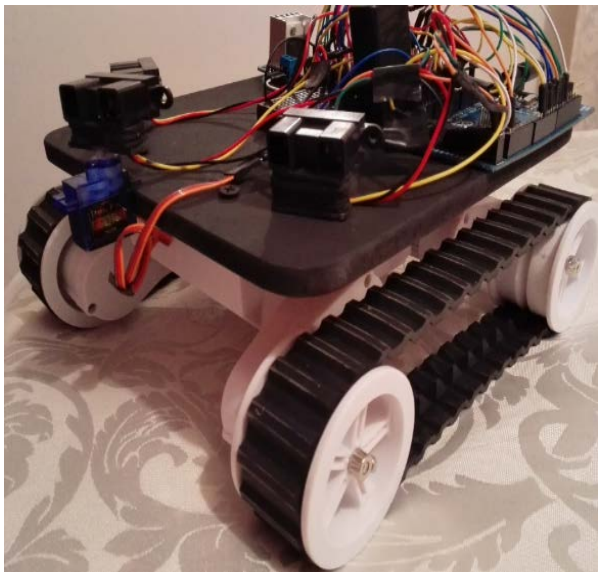
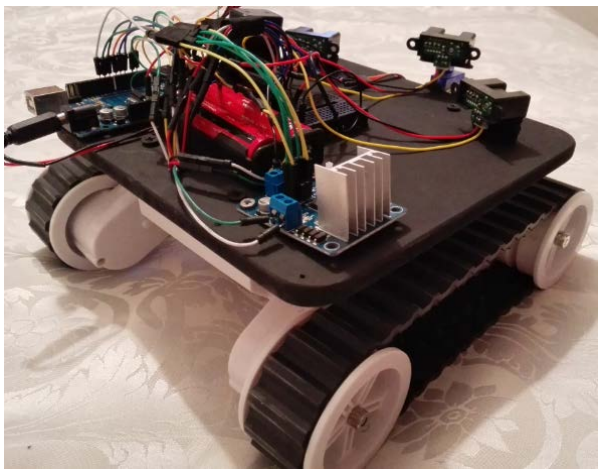


Fig.19. Robot Oruga Perfil Trasero Lateral



VI. RECOMENDACIONES Y RESULTADOS

El Rover 5 es un robot diseñado para objetos que estén a 15 cm de altura, esto se debe a que las posiciones de los sensores están encima de la base del robot.

El Rover 5 no detecta obstáculos con diámetros menores a los 7 cm, también tiene puntos ciegos en su diseño, en cualquier oportunidad el objeto puede estar ubicado preciso donde el robot no lo detecte y este choque contra él.

El Rover 5 está diseñado para suelos con irregularidades mínimas.

La precisión de los sensores y la inteligencia artificial aplicada al robot es óptima porque cumple con los requerimientos de la tabla de condicionales.

Para futuros diseños se deberán implementar más sensores para no generar puntos ciegos y así permitir que el robot tenga un excelente desempeño como detector y esquivador de obstáculos.

Los sensores que tiene en las esquinas delanteras le permiten reducir los puntos ciegos al robot. El robot está diseñado para evitar obstáculos de frente ya que como tal no es capaz de dar retroceso y sus movimientos se limitan a dar dirección derecha e izquierda y seguir de frente.

V. CONCLUSIONES

El Rover 5 cumple con los parámetros del fuzzy logic, este atiende o es capaz de visualizar los objetos y dependiendo de la posición de los objetos los intenta esquivar.

Debido a sus puntos ciegos el Rover 5 no es capaz de esquivar todos los objetos que se encuentra en el camino.

Este robot no detecta objetos a una altura menor a los 20 cm y con un diámetro menor a los 7 cm.

El diseño de posición de los sensores ayudo a reducir los puntos ciegos del robot.

La precisión de detección de distancia es óptima, a los 17.9 cm.

REFERENCIAS

[1] A. Barrientos, L. Peñín, C. Balaguer y R. Aracil. "Fundamentos de Robótica". Mc Graw Hill. Segunda edición. 2007.

[2] A. Ollero Baturone. "Robótica, Manipuladores y robots móviles". Alfaomega. Marcombo. 2001.

[3] F. Escolano, M. Cazorla, M. Alfonso, O. Colomina y M. Lozano. "Inteligencia Artificial Modelos, Técnicas y Areas de Aplicación". Thomson Editores Spain. 2001. ISBN:84-9732-183-9.

[4] D.A. Tibaduiza y M. Anaya. "Campos de Potencial Aplicados al Planeamiento de Caminos de Robots Móviles". Revista de Ingeniería. Vol. 1, N° 3, pp. 23-34. Enero 2007.

[5] D.A. Tibaduiza. "Planeamiento de Trayectorias de Robots Móviles". Tesis para optar al grado de Maestría. Universidad Industrial de Santander. 2005.