Robótica reactiva a través de una herramienta gráfica de simulación

Antonio Benitez[1], Rodolfo Vasconcelos López[3], Ma. Concepción de la Cruz Gómez[3]
Ma. Auxilio Medina[2], Jorge de la Calleja[2]
[1]Maestría en Ingeniería, [2]Ingeniería en Informática, Universidad Politécnica de Puebla
[3]Licenciatura en Sistemas Computacionales, Universidad Juárez Autónoma de Tabasco

{antonio.benitezruiz, mauxmedina, jorgedelacalleja}@gmail.com

Resumen

La robótica es una área de desarrollo interesante pero desafiante. Un problema de interés en esta área es la robótica móvil, aquella en donde se busca el desarrollo de algoritmos capaces de mover a un vehículo sujeto a restricciones. Este tipo de proyectos puede resultar costosos, por lo que una alternativa es proponer una solución desde la simulación por computadora. Este documento describe los elementos básicos necesarios para simular de manera gráfica un robot móvil tipo diferencial. Esta simulación incluye el diseño del robot, los elementos algebraicos para la solución del problema cinemático, la interfaz de usuario para mover a este robot y las aplicaciones desarrolladas a partir de los movimientos básicos.

1. Introducción

Hoy en día la graficación por computadora juega un papel importante en el desarrollo de las tecnologías y en la perfección de las mismas. La graficación por computadora tiene un gran campo de aplicación, por mencionar algunas: la representación de ambientes mediante gráficos, en la producción de videos musicales, anuncios de televisión, en películas, en el análisis de datos, claro sin dejar de mencionar la importancia de los simuladores virtuales en 2D y 3D, estos últimos teniendo gran impacto por el realismo que reproducen.

Una de las áreas más interesantes de la robótica es la relacionada con los robots móviles. Estos cuentan con gran capacidad de desplazamiento, basados en carros o plataformas y dotados de un sistema locomotor de tipo rodante. Siguen su camino por telemando o guiándose por la información recibida de su entorno a través de sus sensores.

Sin embargo, realizar pruebas de comportamiento con robots móviles de manera física implica gastos económicos elevados y posiblemente la mala utilización de éstos dispositivos. Por esta razón, el presente trabajo propone el desarrollo de un ambiente gráfico de simulación para representar un robot móvil con características similares a las que ofrece el Robot comercial Pololu 3Pi [2]. Así, este simulador podrá utilizarse para reproducir el mismo tipo de comportamientos que pueden implementarse sobre la plataforma Pololu 3 Pi. La propuesta de este simulador se soporta en un ambiente gráfico desarrollado con anterioridad.

Por otro lado, esta propuesta puede utilizarse como una herramienta de soporte didáctico para explicar conceptos de cinemática de cuerpos rígidos en el área de la robótica y elementos básicos del área de graficación por computadora. Así, simulación por computadora se convierte en una alternativa importante.

Este artículo esta organizado de la siguiente manera. En la sección 2, se describe el diseño de la herramienta de simulación. Los elementos matemáticos necesarios para el control cinemático del robot se presentan en la sección 3. En la sección 4, se describe el desarrollo de la interfaz de usuario propuesta para el envió de comandos al robot. En la sección 5. Se presentan los resultados obtenidos de dos aplicaciones desarrolladas para este robot, y finalmente se concluye el trabajo propuesto.

2. Desarrollo del Ambiente Gráfico

Los ambientes gráficos de simulación requieren de un diseño que vaya de acuerdo a su aplicación. Este proyecto se soporta en un ambiente gráfico llamado

GEMPA (*Graphic Environment for Motion Planning Algorithm*) [1].

2.1 Ambiente gráfico de soporte

GEMPA es una herramienta gráfica creada para simular entornos virtuales tridimensionales. GEMPA fue desarrollado para simular el comportamiento de algoritmos de planificación de movimientos de cuerpos rígidos. Para ello, esta herramienta incluye; la representación de objetos utilizando distintos tipos de archivos; la animación de la ruta libre de colisión de un problema de planificación de movimientos, y el desarrollo de una interfaz gráfica de usuario para navegar a través del entorno virtual [1].

El presente proyecto se soporta el IDE de Dev C++ y Librerías de OpenGL, ya que son las herramientas de desarrollo de GEMPA.

OpenGL es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D [5].

2.2 Características del robot a simular

El Pololu 3pi [4] es un pequeño robot móvil autónomo de alto rendimiento, diseñado para competencias de seguimiento de líneas, resolución de laberintos y detección de obstáculos. El robot tiene una arquitectura diferencial (ver Figura 1.) y algunas aplicaciones interesantes sobre esta plataforma pueden verse en [2].

Por la simplicidad y versatilidad de la plataforma asociada al Pololu 3pi, se selecciono a este robot como el primer candidato a ser simulado a través del ambiente gráfico.

Figura. 1. Robot Pololu 3Pi utilizado para la simulación.

Para la simulación del robot es necesario asociarle una malla de triángulos a cada uno de los elementos que lo conforman, por ejemplo, las llantas, el cuerpo del robot, los sensores, etc. Estas mallas son procesadas por un módulo de pintado (vértice por vértice) que posee el GEMPA. Estas mallas tienen una estructura bien definida [1].

En la Figura 2. Se muestra la forma en la que se proporciona la información necesaria para la estructura del robot al simulador. En esta estructura se pueden diferenciar elementos como las mallas de triángulos asociadas a cada una de sus partes, así como los parámetros necesarios de posición, orientación, escala y color.

El concepto de configuración se asocia a un cuerpo rígido en un espacio tridimensional. Esta configuración esta formada por tres parámetros (dos de ellos determinan posición y uno de ellos determina orientación). Así, una configuración puede ser presentada como (*x, y, alfa*).

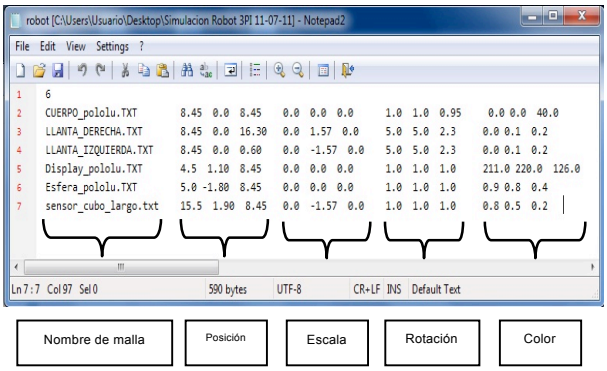

Figura. 2. Archivo de datos con las especificaciones necesarias para que el ambiente gráfico pinte cada una de las partes que integran al robot.

Esta configuración nos permite especificar de manera completa la localización del robot dentro del ambiente. Para ello, se utiliza un sistema de referencia global, así el robot simulado puede verse dentro del ambiente como se muestra en la Figura 3.

2.3 Dimensiones y estructura del Ambiente

La dimensión del ambiente en el que el robot se desplaza es finita, por esta razón se proponen tres dimensiones diferentes, una de tamaño reducido, una de tamaño regular y una más de gran tamaño. Así, se dibuja un piso sobre el cual trabaja el robot. El tamaño del ambiente esta determinado por el número de cuadros de tamaño finito que lo componen. Se sugiere que cada cuadro no sea mayor a las dimensiones

asociadas al robot, de esta manera el robot avanzará o retrocederá distancias no mayores a su dimensión. Es importante decir que esta es una primera versión del ambiente, en el que la locomoción del robot se hace en términos distancias fijas, por lo que una secuencia de distancias fijas provocaran un desplazamiento mayor para el robot.

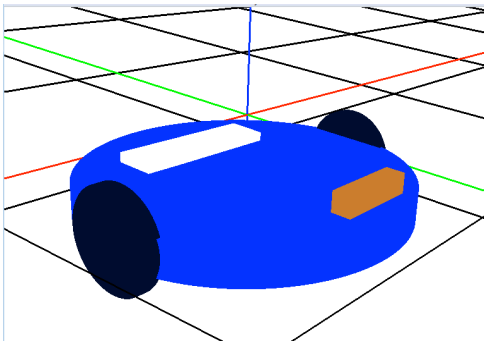

Figura. 3. Simulación del Robot Pololu 3Pi.

Así, las dimensiones propuestas para esta primera versión del ambiente son, piso de 4 x 4 cuadros, piso de 8 x 8 cuadros y piso de 12 x 12 cuadros. La Figura 4. muestra un ejemplo de ambiente de dimensión 8x8.

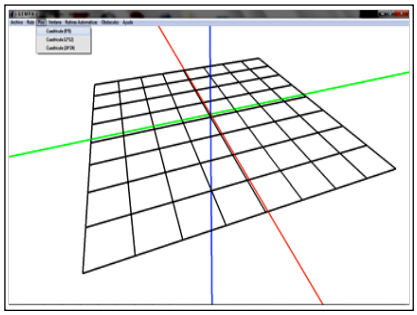

Figura. 4. Vista de ambiente gráfico de dimensiones 8x8.

3. Locomoción del Robot

El Robot en el ambiente es capaz de realizar cuatro movimientos básicos para navegar dentro de él. Sin embargo, aunque se propone una interfaz que le permite al usuario de manera muy sencilla enviar estos comandos hacia el robot, es necesario implementar estos movimientos a través del concepto de cinemática de un cuerpo rígido.

3.1 Movimientos propuestos

1.- Avanzar: Consiste en desplazar el robot hacia adelante el número de cuadros que se le indique.

2.- Retroceder: Consiste en desplazar el robot hacia atrás el número de cuadros que se le indique.

3.- Vuelta a la Izquierda: Consiste en dar un giro de 90 grados en sentido contrario a las manecillas del reloj.

4.- Vuelta a la Derecha: Consiste en dar un giro de 90 grados en sentido de las manecillas del reloj.

Los cuatro movimientos propuestos se resuelven de manera finita y discreta. Por esta razón el piso en el que se desplaza el robot es cuadriculado, ya que para los movimientos de avanzar y retroceder se resuelven de manera reducida para un solo cuadro en el ambiente. En el caso de los movimientos de vuelta a la izquierda y a la derecha solo se gira en un rango de 90 grados. El esquema de la Figura 5. muestra las direcciones hacia las que el robot se puede desplazar sobre el plano XZ.

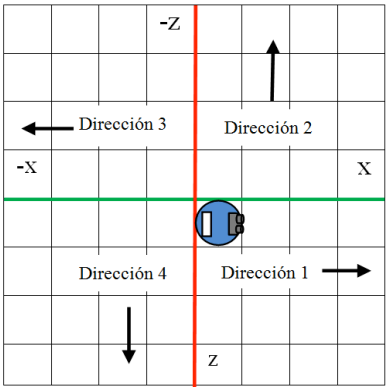

Figura. 5. Vista superior del ambiente y plano XZ sobre el que se desplaza el robot.

Para resolver los problemas asociados con los movimientos de avanzar y retroceder se hace la discretización de una recta entre las coordenadas del centro de dos cuadros adyacentes dentro del ambiente.

El proceso consiste en trazar un segmento de recta dirigido y dividirlo en un número finito de partes, en donde cada una de estas partes se convertirá en la configuración en donde se pintará al robot para generar la simulación gráfica deseada. La Figura 6. muestra la idea de este proceso de discretización en donde el número de partes en el que se divide la línea recta es de 17. Es importante mencionar que el grado de discretización, es decir el número de partes en las que el segmento de recta será dividido puede ser

proporcionado por el usuario, sin embargo hay que considerar que mientras más fina sea la discretización la animación será mas costosa (tiempo de procesamiento y velocidad a la que la simulación es presentada).

Los movimientos vuelta a la derecha y vuelta a la izquierda, los cuales están acotados a vueltas de 90 grados consisten en la discretización del ángulo dado en radianes. Para esto, se divide π/2 en sentido contrario a las manecillas del reloj; de esta manera tenemos un movimiento de 90 grados que se fracciona en tantas partes como se desee (con las mimas restricciones que la discretización de la línea recta). Esto puede verse en la Figura 7.

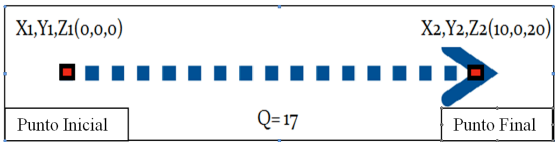

Figura. 6. Discretización de un segmento de recta dirigido para la animación del robot en las operaciones de avanzar y retroceder.

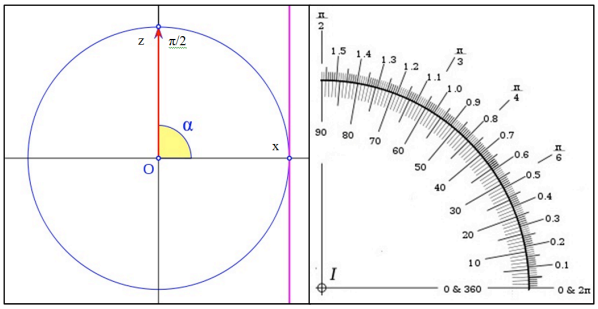

Figura. 7. Discretización de un ángulo dado en radianes para la animación del robot en las operaciones de vuelta a la derecha y vuelta a la izquierda.

3.2 Matrices de Transformación

La graficación por computadora tiene como base para su desarrollo las operaciones matriciales de *traslación*, *rotación* y *escala* de objetos en dos y en tres dimensiones. Estas transformaciones permiten el desarrollo de herramientas de animación por computadora que se propagan en los ambientes de simulación gráfica.

Las operaciones matriciales de *traslación*, *rotación* y *escala* se utilizan para calcular la posición y/o el tamaño de los objetos, con respecto a los sistemas de referencia, son también llamadas transformaciones lineales. Estas operaciones de transformación son siempre matriciales, pueden descomponerse en sumas y multiplicaciones, y por tanto caen dentro de la categoría de operaciones lineales que podemos incluir de forma eficiente en el procesamiento gráfico de los vértices que definen a los objetos [3].

Estas transformaciones deben aplicarse a cada uno de los vértices de los objetos que se desea manipular y es importante que cada vez que un objeto se mueve, rota o se escala, los vértices asociados al objeto deberán recalcularse.

A continuación se describe brevemente cada una de las transformaciones utilizadas.

Traslación

La transformación de traslación consiste en mover una cierta distancia un objeto en una dirección determinada, como consecuencia el objeto se encontrará en una nueva posición. La expresión (1) muestra a transformación necesaria para realizar dicha operación (Tx, Ty, Tz representan el vector de traslación).

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix} \quad (1)$$

Rotación

Para realizar una rotación o giro de los objetos en 3D, el usuario debe establecer un eje de rotación, así como el ángulo y el sentido de giro alrededor de dicho eje. Según sea la naturaleza del ángulo, los giros podrán ser relativos, o bien absolutos. Así, cuando se deba rotar el objeto un ángulo θ sobre un eje dado, partiendo de la posición actual del objeto, el giro será relativo. En cambio, si el objeto se ha de girar un ángulo θ a partir del estado cero entonces se tratará de un giro absoluto.

En las rotaciones, es especialmente importante el sentido en el que giran los objetos y esta es la razón por la cual se trata de seguir una convención para que no se produzcan ambigüedades. En general, las medidas de ángulos de rotación positiva generan giros en sentido anti horario (cuando se mira hacia el origen, desde una posición con coordenadas positivas). La expresión (2) permite rotar un objeto con respecto al eje Z un ángulo θ. Los valores de X_R, Y_R y Z_R, corresponden a un punto dentro del objeto, mismo que

será considerado como un punto fijo [3]. De esta manera la transformación esta formada por una traslación del objeto hacia el origen de sistema, posteriormente se aplica una rotación respecto al eje Z y finalmente se regresa el objeto a su posición inicial con una transformación de traslación.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_R & -y_R & -z_R & 1 \end{bmatrix} * \begin{bmatrix} cos\theta & sen\theta & 0 & 0 \\ -sen\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x_R & y_R & z_R & 1 \end{bmatrix}$$

$$\begin{bmatrix} cos\theta & sen\theta & 0 & 0 \\ -sen\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ (1-\cos\theta)x_R + y_R \cdot sen\theta & (1-\cos\theta)y_R - x_R \cdot sen\theta & 0 & 1 \end{bmatrix} \quad (2)$$

Escala

Dentro de un espacio de referencia los objetos pueden modificar su tamaño relativo en uno, dos, o los tres ejes. Para ello se utiliza la matriz de escala. Si no se toman precauciones, los cambios de escala (como los de rotación), además de suponer una variación en las proporciones de los objetos, también implican una traslación de los mismos, un efecto colateral no deseado en muchas ocasiones.

Para evitar este efecto no deseado, la escala se realiza con respecto a un punto fijo, esta técnica consiste en escalar un objeto, sin que éste se vea afectado por un cambio de posición, respecto a un punto establecido. Para llevar a cabo un escalamiento respecto a un punto fijo, se procede multiplicando una matriz de traslación para llevar el punto fijo al origen, después se multiplica por una matriz de escalamiento y posteriormente por otra de traslación para llevar al punto fijo a su posición original. La expresión (3) presenta la transformación mencionada, en donde los valores de X_F, Y_F y Z_F, representan un punto dentro del objeto (punto fijo) y los valores de S_x, S_y y S_z representan los factores de escala con respecto a cada uno de los ejes [3].

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_F & -y_F & -z_F & 1 \end{bmatrix} * \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x_F & y_F & z_F & 1 \end{bmatrix}$$

$$= \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ (1-S_x)x_F & (1-S_y)y_F & (1-S_z)z_F & 1 \end{bmatrix} \quad (3)$$

4. Funcionalidades e Interfaz

Uno de los objetivos iniciales del proyecto es implementar locomoción para un robot móvil. Por ello, es necesario que estas funcionalidades se presenten al usuario a través de una interfaz básica que le permita mover el robot dentro del ambiente. Sin embargo, fue necesario incorporar otras funcionalidades con el objetivo de enriquecer la herramienta de simulación gráfica para ofrecer al usuario mayor flexibilidad.

4.1 Interfaz

En la Figura 8. se presenta la interfaz de usuario, misma que permite enviar comandos al robot, enviando parámetros a través de botones que permiten la manipulación directa con el robot para realizar los movimientos deseados.

Es importante decir que existe un estado inicial para el robot, en este estado el robot se coloca en una posición inicial con una orientación específica, lo que determina una configuración inicial. A partir de esta configuración, después de la ejecución de un comando de movimiento cualesquiera, la configuración del robot cambia. El usuario puede enviar el número de comandos que desee y todos estos comandos son almacenados como un histórico para que cuando el usuario lo solicite pueda ejecutar de una sola vez toda la secuencia solicitada.

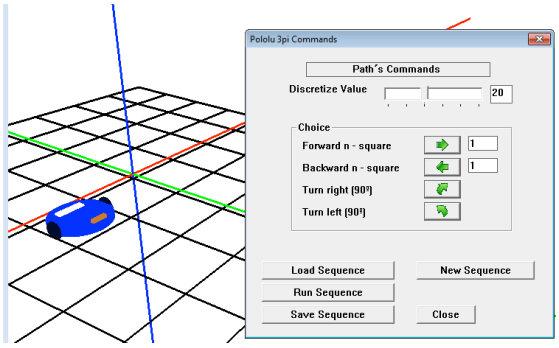

Figura. 8. Interfaz de usuario que permite el envió de comandos para la locomoción del robot.

Además de permitir la manipulación del robot a través de la interfaz, esta aplicación también ofrece al usuario las siguientes operaciones:

- *Guardar secuencia de movimientos ejecutados en un archivo*. Esta funcionalidad permite al usuario almacenar la secuencia de

movimientos que ejecuto sobre el robot para que pueda volver a utilizarla cuando lo requiera, recuperándola desde el mismo archivo.

- *Recuperar desde un archivo la secuencia de movimientos a ejecutar*. Esta opción le permite al usuario recuperar desde un archivo de texto, una secuencia de comandos específica para el robot, sin importar que la secuencia de comando sea grande.

- *Ejecutar nuevamente la secuencia de movimientos solicitada*. Esta tarea permite al usuario ejecutar nuevamente la secuencia de comandos cargada o actual. Tomando en cuenta que la secuencia actual es aquella que el usuario fue construyendo con cada comando solicitado o bien una secuencia recuperada desde un archivo.

- *Reiniciar al robot para ejecutar una nueva secuencia de movimientos*. Esta funcionalidad permite restaurar las condiciones iniciales para el robot, borrando la secuencia de movimientos que exista e inicializa su configuración inicial.

4.2 Navegación Aleatoria

Esta opción no requiere que el usuario envíe ningún comando al robot, ya que automáticamente y de manera aleatoria se genera el comando a ejecutarse y en su caso (para comandos de avanza y retrocede) el número de cuadros sobre el que se aplica dicho comando. Esto permite que el robot se desplace dentro del ambiente de manera continua y de forma aleatoria. Esta operación se ejecuta hasta que el usuario lo desee, ya que para detener dicha operación deberá hacerlo de manera manual a través de la interfaz.

Como consecuencia de esta navegación aleatoria, se propuso simular a través del uso de una estructura de datos, un sensor de proximidad. Este sensor tiene la tarea de generar un evento cada vez que el robot se acerque a un obstáculo. Así, se integra la opción de navegación aleatoria junto con la evasión de obstáculos. Para ello, el usuario puede generar de manera aleatoria la posición de los obstáculos dentro del ambiente. En esta primera versión del proyecto, las dimensiones de los obstáculos serán idénticas a la dimensión de un cuadro dentro del ambiente.

El número de obstáculos a generar lo proporciona el usuario y este número se valida contra el número de cuadros que forman el ambiente activo. La Figura 9. muestra la simulación dentro del ambiente gráfico del robot desempeñando las dos tareas a la vez, por una lado la generación aleatoria de comandos y por otro la evasión de obstáculos a través de la simulación de un sensor de proximidad.

Es importante señalar que todas estas funcionalidades ya han sido desarrolladas sobre la plataforma real del Pololu 3Pi [2], por lo que es posible inicialmente comparar el comportamiento de los robots en ambos ambientes. Un ejemplo de la navegación aleatoria evitando obstáculos desde el ambiente gráfico y desde un ambiente real puede verse en la Figura 10.

5. Resultados Conclusiones y Trabajo Futuro

Este proyecto logró la implementación satisfactoria de la simulación del robot en un ambiente gráfico, la adecuación de una plataforma gráfica para robots móviles y establecimiento de varias dimensiones para el ambiente de trabajo.

Por otro lado, se implementó y probó las funcionalidades que permiten al robot desplazarse dentro del ambiente, esto es; moverse hacia adelante, hacia atrás, dar vuelta a la derecha o hacia la izquierda. Estas funcionalidades se proponen operar a través de una interfaz gráfica de usuario, facilitando el envió o recuperación de comandos hacia el robot.

Con estas funcionalidades se propusieron tareas más complejas como la navegación aleatoria y la navegación evitando obstáculos.

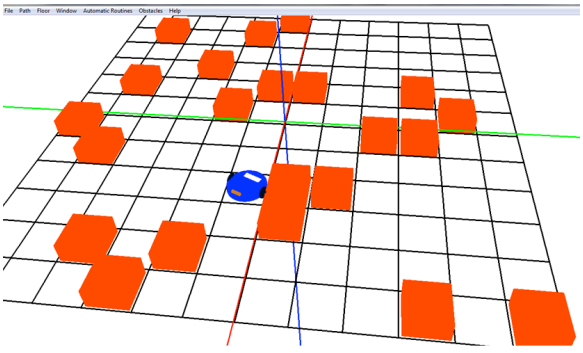

Figura. 9. Simulación de navegación aleatoria evitando obstáculos, simulando un sensor de proximidad.

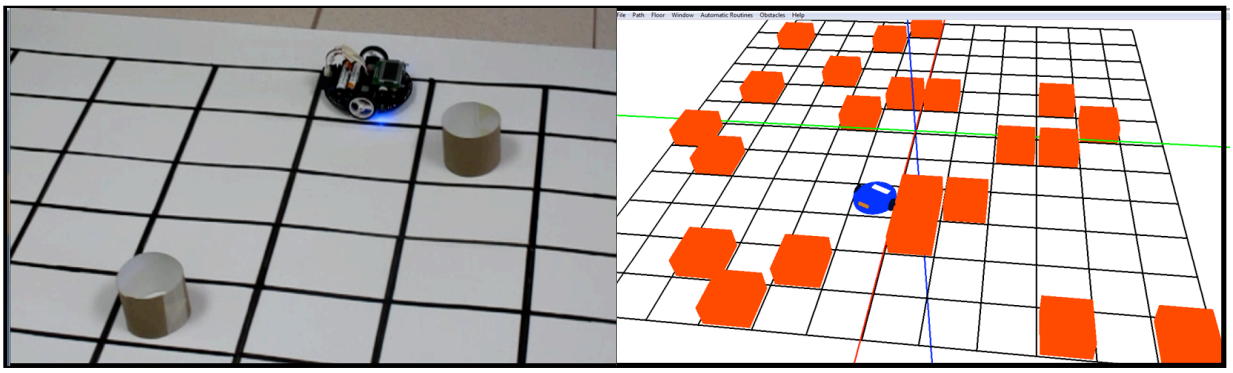

Figura 10. Navegación aleatoria evitando obstáculos en un ambiente real (lado izq.) y en el ambiente virtual (lado der.)

Con la implementación de este proyecto se cumplen todas las expectativas propuestas dado que el objetivo principal es implementar los movimientos básicos del

Pololu 3Pi (rotación derecha-izquierda, avanzar y retroceder), así como la conformación de su estructura dentro del ambiente donde pueda ejecutar diferentes secuencias de movimiento.

Este proyecto tiene la posibilidad de continuar con nuevas propuestas de simulación. Por ejemplo; la programación de un sigue líneas a través de la simulación de sensores infrarrojos; la programación de algoritmo de sigue muros utilizando los sensores de proximidad; mejorar la simulación de los sensores de proximidad; incrementar el realismo dentro del ambiente incorporando elementos de iluminación y texturas. Este proyecto se ha realizado con el propósito de hacer nuevas aportaciones a la herramienta GEMPA, con lo que se permite la simulación de un robot Pololu 3Pi en un ambiente en 3D.

6. Reconocimientos

Este proyecto se lleva a cabo con la colaboración de el cuerpo académico de Sistemas y Cómputo Inteligente de la Univeridad Politécnica de Puebla y el Cuepro Académico de Sistemas Inteligentes de la Universidad Juárez Autónoma de Tabasco (UJAT). De manera particular se agradece la colaboración de los alumnos Rodolfo Vasconcelos López y María Concepción de la Cruz Gómez, estudiantes de la licenciatura en Sistemas Computacionales de la UJAT, ya que el trabajo presentado aquí es parte de su tesis de licenciatura.

7. Referencias

[1] Antonio Benitez and Alejandro Mugarte. "*GEMPA: Graphic Environment for Motion Planning algorithm*", In Research in Computer Science and Engineering. Volumen 42, pag. 225-236., Mayo 2009.

[2] Antonio Benitez et al. *Programación de Vehículos móviles bajo Plataforma Pololu 3Pi*., en Revista Visión Politécnica. Año 6/Núm. 1. Pag. 23-27. Junio del 2011.

[3] Informática Gráfica, "Manipulación del espacio: Trasformaciones y proyecciones", Universidad de Salamanca Facultad de Ciencias Ingeniería en Informática, Disponible: < http://gsii.usal.es/~igrafica/> Recuperado el día 15 de enero de 2011.

[4] Pololu Corporation. (2009). *Pololu 3pi Robot Users Guide*. Robot Pololu 3pi Guía de usuario (pp. 1-48). Con dirección electrónica en: <http://www.pololu.com/docs>.

[5] Wright Haemel, Sellers Lipchak, *OPpenGL Superbible,* Fifth Edition. 2011.