DISEÑO Y SIMULACIÓN DE UN ROBOT MÓVIL RECOLECTOR DE OBJETOS

Maikel O. Torres Piñeiro [1], Valery Moreno Vega [2]

[1]Instituto Superior Politécnico J.A. Echevarría. Calle 114 No11901 e/119 y 127.Ciudad de la Habana.Cuba. maikel@electrica.cujae.edu.cu

[2] Instituto Superior Politécnico J.A. Echevarría. Calle 114 No11901 e/119 y 127.Ciudad de la Habana.Cuba. valery@electrica.cujae.edu.cu

RESUMEN / *ABSTRACT*

El trabajo aborda la implementación de un algoritmo de recolección de objetos basado en un mapa conocido del ambiente donde se encuentra el robot. Para lograr que el robot se mueva de un punto a otro y a su vez esquive obstáculos se implementaron algoritmos de planificación de la trayectoria y control de la posición del robot basados en regiones geométricas. Estos resultados fueron simulados en el programa Virtual Robot Simulator.

Palabras clave: control, mapa, planificación, simulación.

DESIGN AND SIMULATION OF A MOBILE ROBOT OBJECT RECOLECTOR

This paper addresses the implementation of an object gathering algorithm based on a well-known map of the surface where the robot is moving around. Both trajectory planning and robot positioning control algorithms were implemented to guide the robot from an initial point to a final one and in turn avoid obstacles. These results were simulated with a framework named Virtual Robot Simulator.

Key word: control, mapa, path planing, simulation

INTRODUCCIÓN

Hoy en día las aplicaciones en el campo de la robótica móvil son diversas. Por ejemplo en la minería, la exploración espacial, en la recogida de desechos tóxicos, etc.

Una aplicación de interés es un robot recolector de objetos. Para este tipo de problema, donde el robot tiene que identificar y recoger objetos dentro del área de trabajo, se necesita primeramente suministrarle información del medio donde se moverá. Por ejemplo obstáculos, posición de los objetos, límites, etc. A partir de esa información y utilizando determinados algoritmos se planifican y generan trayectorias y se controla el movimiento del robot.

En este trabajo se propone suministrarle al robot la información requerida a través de un mapa del área de trabajo donde se codifican los obstáculos, los objetos y los caminos libres así como la posición dentro del espacio de movimiento. Se propone además el uso de dos algoritmos para la planificación y control de las trayectorias que debe seguir el robot.

La aplicación se simula utilizando la herramienta Virtual Robot Simulator (VRS) y el robot utilizado para las simulaciones es del tipo diferencial con una separación entre ruedas de 0.8 m.

MATERIALES Y MÉTODOS

MAPA DEL TERRENO

Para el desarrollo de este trabajo se asumió que el robot se mueve por un terreno bidimensional, por lo cual su representación se puede realizar mediante una matriz cuyas dimensiones dependerán de las dimensiones del terreno y la resolución que se quiera del mismo[1].

Por ejemplo, un terreno de 20 m x 20 m puedes ser representado por una matriz de dimensión 20x20 si se desea una resolución de 1 m, o puede ser representado por una matriz de 4x4 si se desea una resolución de 5 m.

En cada celda de la matriz se guarda la información relacionada con esa porción de terreno, esta puede ser:

- libre de obstáculos (representada por 0)
- hay obstáculos (representada por 2)
- hay objeto para recoger (representada por 1)

El terreno elegido para realizar las pruebas de los algoritmos que se emplearán en este trabajo tiene las dimensiones de 20 m x 20 m y se tomó una resolución de 1 m. La figura 1 muestra el ambiente sobre el cual se va a trabajar. El mismo ha sido modelado en el programa Virtual Robot Simulator (VRS). El área pintada de verde representa el lugar donde hay objetos para recoger y el área roja es el lugar donde los tiene que llevar. La información del mapa está representada por una matriz M de dimensión 20x20.

M =

$$\begin{pmatrix}
2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2\\
2&0&0&0&0&0&2&0&0&0&0&0&0&0&0&0&0&0&0&2\\
2&0&0&0&0&0&2&0&0&0&0&0&0&0&0&0&0&0&0&2\\
2&0&0&0&0&0&2&0&0&0&0&0&0&0&0&0&0&0&0&2\\
2&0&0&0&0&0&2&0&0&0&0&0&0&0&0&0&0&0&0&2\\
2&0&0&0&0&0&2&0&0&0&0&0&0&0&0&0&0&0&0&2\\
2&0&0&0&0&0&2&0&0&1&0&0&0&0&0&0&0&0&0&2\\
2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2\\
2&0&0&0&0&0&0&0&0&0&0&0&2&2&2&0&0&2&0&2\\
2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2\\
2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2\\
2&2&2&2&2&0&0&0&0&0&0&0&0&0&1&0&0&0&2&\\
2&0&0&0&0&0&0&2&0&0&0&0&0&0&0&0&0&0&2&\\
2&0&0&0&0&0&0&2&0&0&0&0&0&0&2&0&0&0&2&\\
2&0&0&0&1&0&0&2&0&0&0&0&0&0&0&0&0&0&2&\\
2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&\\
2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&\\
2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&\\
2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&\\
2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2&2
\end{pmatrix}$$

Una vez obtenida la matriz del mapa solo resta implementar algún algoritmo que permita recoger dicha información y le indique al módulo planificador de trayectoria los puntos a los que debe ir el robot.

En este caso se utilizó un barrido del mapa, se fue analizando celda por celda de izquierda a derecha y obteniendo las coordenadas de los obstáculos y los objetos a recoger.

El algoritmo de recolección se implementó en C++ y a continuación se muestra un seudo código del mismo:

1) leer el mapa (obtener coordenas de los objetos y obstáculos así como la cantidad de cada uno de ellos)
2) mientras quede un objeto por recoger
   a) planificar trayectoria para llegar al objeto
   b) planificar trayectoria para ir al punto donde se depositan los objetos
3) fin del ciclo
4) fin

PLANIFICACIÓN DE LA TRAYECTORIA

Al módulo de planificación de trayectorias se le pasa la información recogida del mapa, la cual consiste en las coordenadas de todos los objetos que se deben recoger y las coordenadas de todos los obstáculos que haya en el terreno.

Este módulo debe ser capaz de generar una secuencia de puntos por los cuales debe transitar el robot. Pare esto se utilizó un algoritmo basado en regiones geométricas propuesto en [2] con algunas modificaciones.

Este algoritmo consiste en modelar los obstáculos mediante un círculo cuyo radio sea tal que el obstáculo quede contenido en el círculo. Luego trazar una recta entre la posición del robot y el punto destino y determinar si la misma se intercepta con algunos de los círculos que representan los obstáculos. En caso de no interceptar con ningún círculo la recta es el camino que debe seguir el robot, de lo contrario se hallan las rectas tangentes al círculo que pasen por el punto donde se encuentra el robot. Con dichas rectas se pueden obtener dos puntos que bordean al obstáculo y por los cuales podrá pasar el robot. Esta secuencia de pasos se repite hasta que el robot llegue al punto destino.

Este algoritmo fue implementado en C++ y trabajando con la matriz M ofreció el resultado mostrado en la figura 2. El seudo código del algoritmo de planificación se muestra a continuación:

1) modelar obstáculos
2) si hay obstáculos entre el robot y el punto destino
   a) obtener los dos puntos que bordean al obstáculo
   b) seleccionar el correcto
   c) si hay obstáculos entre el robot y el punto seleccionado ir al paso a
   d) llevar el robot al punto seleccionado
   e) ir al paso 2.
3) llevar el robot al punto destino
4) fin

En la figura 2 se puede observar como el robot es capaz de llegar a los puntos de interés sin chocar con ningún obstáculo.

CONTROL DE LA POSICIÓN DEL ROBOT

Una vez obtenida la secuencia de puntos por las cuales debe transitar el robot, resta ejercer un control sobre la velocidad de cada una de las ruedas del robot para poder llegar a la posición deseada. Existen diversos algoritmos de control para un robot móvil [3,4,5], en este módulo se implementó el controlador propuesto en [6] el cual se basa en la consideración geométrica mostrada en la figura 3.

En este caso se desea que $\varphi \rightarrow 0$ y $\rho \rightarrow 0$ y se plantean como leyes de control las ecuaciones. 1) y 2).

$$v = k1 \cdot \rho \cdot \cos\varphi \qquad 1)$$

$$w = -k1 \cdot \sin\varphi \cdot \cos\varphi - k2 \cdot \varphi \qquad 2)$$

donde: v es la velocidad lineal del robot,

w es la velocidad angular,

k1 y k2 parámetros de diseño.

La implementación del controlador se realizó en C++ de tal manera que se pudiera establecer una comunicación con el simulador VRS y pasarle los parámetros de las variables a controlar. A continuación se muestra el seudo código del algoritmo implementado.

1) comunicar con VRS y obtener posición del robot (posición y orientación)
2) calcular parámetros del controlador (φ, ρ)
3) calcular variables de control (v,w)
4) calcular los valores de velocidad de cada rueda del robot (vl,vr)
5) comunicarse con VRS y darle al robot los valores de vl,vr

VIRTUAL ROBOT SIMULATOR

Virtual Robot Simulator (VRS) es un programa desarrollado por el Grupo de Robótica de la Universidad Politécnica de Valencia (UPV) que sirve para el desarrollo de simulaciones de robots industriales y móviles. En este trabajo se utilizó un ambiente desarrollado en VRS y el mismo se muestra en la figura 1.

VRS permite que otras aplicaciones se comuniquen con él a través de una dll o librería de acceso externo. Esto posibilita la creación de programas externos a VRS que se encarguen de ejecutar las tareas que debe realizar el robot, en este caso la generación y planificación de las trayectorias a seguir para recoger los objetos y llevarlos al punto designado.

La gran potencialidad que brinda esta forma de trabajo consiste en poder crear aplicaciones que se comuniquen tanto con el robot real como con VRS. Esto posibilitará que los algoritmos que se creen puedan ser probados primero en el simulador y unas vez hayan sido puestos a punto no tengan que reprogramarse para el robot real.

Para la comunicación con VRS creó una clase (TCraftControl) capaz de gestionar todo lo relacionado con el robot móvil. Esta clase es la encargada de cargar la dll y utilizar sus funciones para establecer el intercambio de información con el VRS. La figura 4 muestra como se establece dicha comunicación.

La aplicación mostrada en la figura 5 fue programada en Borland C++ Builder 6.0 y en ella se implementaron todos los algoritmos mencionados a lo largo de este trabajo.

CONCLUSIONES

El método propuesto para representar el terreno por el cual se mueve el robot dio buenos resultados ya que permite obtener información de una manera muy sencilla. Igualmente el algoritmo que se implementó para ir a los puntos de recogida de objetos aún dista de ser el más eficiente, pues se pudiera optimizar el recorrido del robot según determinada función objetivo como tiempo o consumo de energía. No obstante, como primer paso a la solución de este problema pensamos que los resultados obtenidos son satisfactorios.

REFERENCIAS

1. **LATOMBE, J. C.:** "Robot Motion Planning", *Kluwer Academic Pub*, Boston, 1991.
2. **MATTEO, L. M. DI; MANGONE, A. C.; MUZZIO, M. L.; VERRASTRO, C.:** "Route planning for vehicle autonomous navigation based on geometrical regions", *XI Reunión de Trabajo en Procesamiento de la Información y Control*, 2005.
3. **PEREIRO, F. ; VERRASTRO, C.:** "Sistema de Comando y Navegación para Robot Móvil con Arquitectura Distribuida", *X RPIC Proceedings*, San Nicolás, Bs. As, pp. 565-569, 2003.
4. **NIÑO-SUÁREZ, P.A; ARANDA- BRICAIRE, E.; VELASCO-VILLA, M.**: "Control mediante modos deslizantes en tiempo discreto para el seguimiento de trayectorias de un robot móvil", *Revista Iberoamericana de Automática e Informática Industrial*. vol 4, No. 4, pp 31-38, 2007.
5. **SILVA-ORTIGOZA, R.; MOLINA-VILCHIS, M.A.H.; SILVA-ORTIGOZA, G.**: "Control de un Robot Móvil de Ruedas Mediante Linealización de Entrada-Salida", *III Congreso Internacional Tendencias Tecnológicas en Computación*, México, Noviembre, 2007.
6. **MARTINS-FILHO, LUIZ S.; MACAU, ELBERT E. N.; ROCHA, RONILSON; MACHADO, ROMUEL F.; HIRANO, LAOS A.:** " Kinematic control of mobile robots to produce chaotic trajectories" , *ABCM Symposium Series in Mechatronics*, vol 2, pp 258-264, 2006.

AUTORES

Maikel O. Torres Piñeiro, Ingeniero en Automática, profesor instructor, Instituto Superior Politécnico José Antonio Echeverría, 266-3341, maikel@electrica.cujae.edu.cu., Investiga en el área de la robótica móvil.

Valery Moreno Vega, Ingeniero en Máquinas Computadoras., Doctor en Ciencias, profesor titular, Instituto Superior Politécnico José Antonio Echeverría, 266-3343, valery@electrica.cujae.edu.cu., Sus intereses de investigación son en el área de Control No Lineal, Optimización, Informática Industrial y Robótica

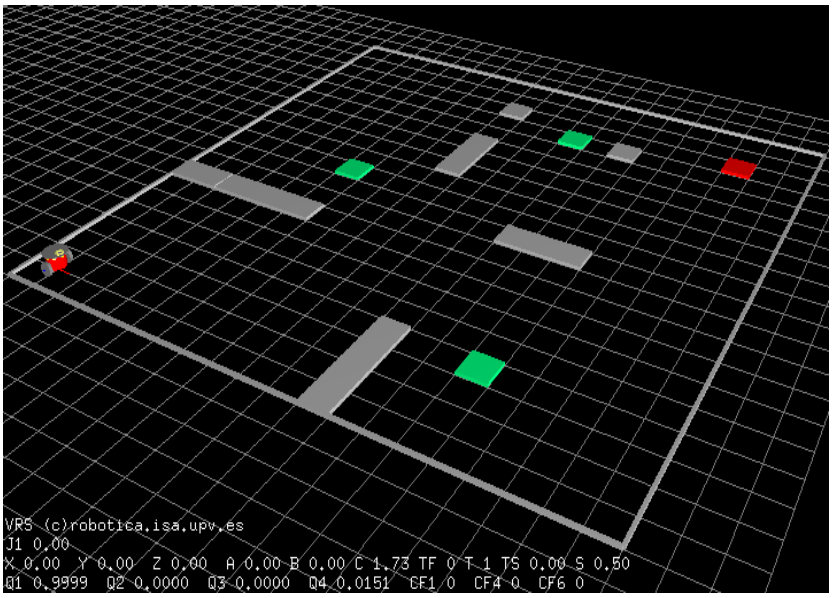

Fig 1. Ambiente para la simulación. Modelado en VRS.

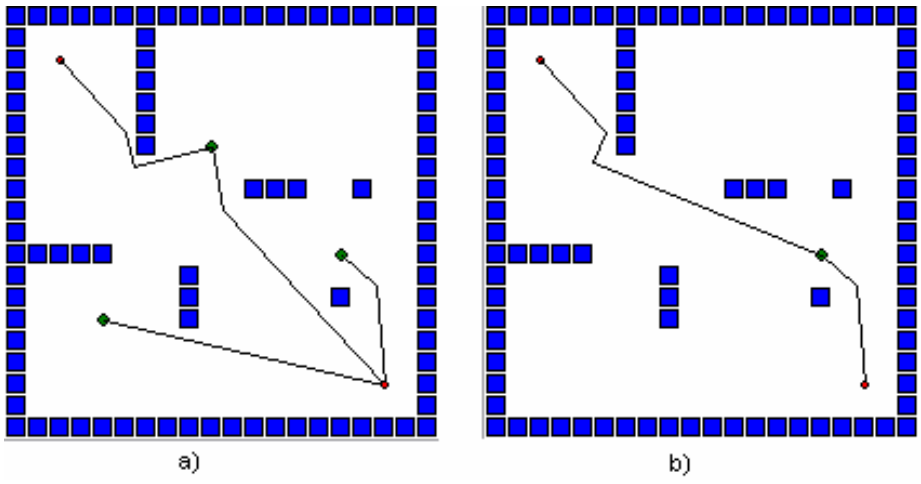

Fig. 2. Resultados de la simulación del algoritmo de planificación de trayectorias. a) Terreno representado por la matriz M. b) Otro terreno.

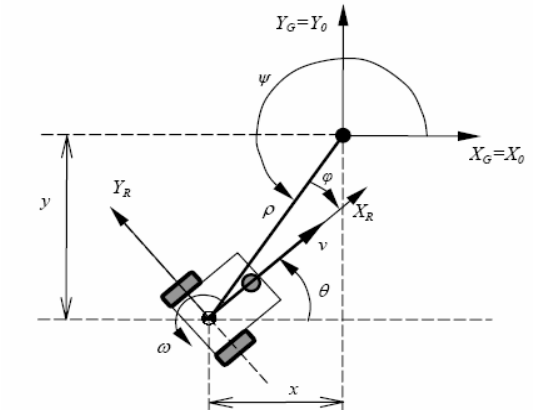

Fig. 3. Situación geométrica en la que se basa el control de la posición del robot.

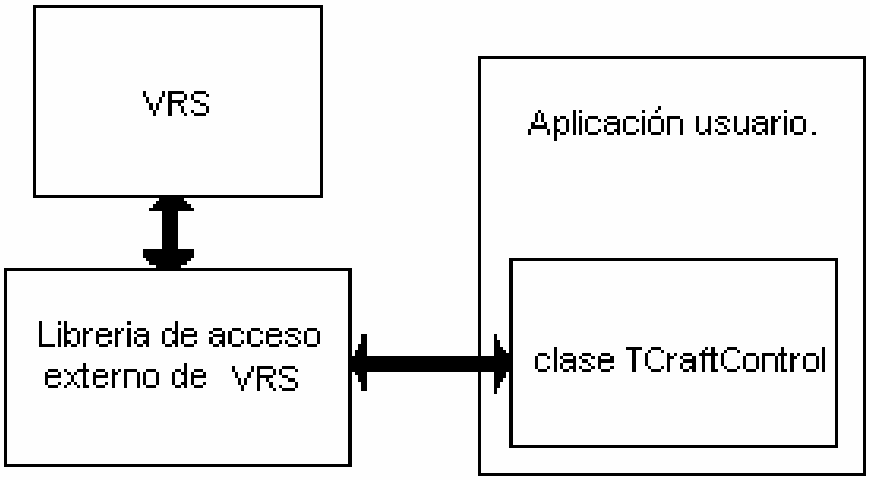

Fig. 4. Arquitectura de la comunicación con VRS.

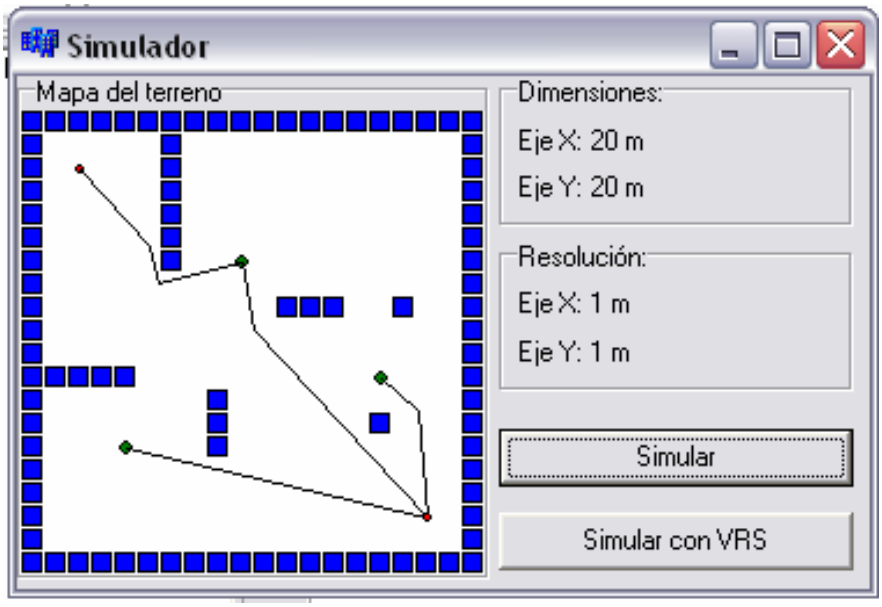

Fig. 5. Aplicación desarrollada en C++ Buider 6.