Planificación de rutas en entornos dinámicos mediante A* y visión artificial en robots móviles

Esteban A. Zapirain
Departamento de Ingeniería Electrónica y Computación - ICyTE UNMdP
Mar del Plata, Argentina
estebanzapirain@fi.mdp.edu.ar

Walter A. Gemin
Departamento de Ingeniería Electrónica y Computación - ICyTE UNMdP
Mar del Plata, Argentina
agemin@fi.mdp.edu.ar

Rodrigo E. Russo
Departamento de Ingeniería Electrónica y Computación - ICyTE UNMdP
Mar del Plata, Argentina
rodrigo.e.russo@fi.mdp.edu.ar

Juan M. López
Departamento de Ingeniería Electrónica y Computación - ICyTE UNMdP
Mar del Plata, Argentina
juanml@fi.mdp.edu.ar

Juan I. Pastore
Departamento de Ingeniería Electrónica y Computación - ICyTE UNMdP
Mar del Plata, Argentina
jpastore@fi.mdp.edu.ar

Melisa G. Kuzman
Departamento de Ingeniería Electrónica y Computación - ICyTE UNMdP
Mar del Plata, Argentina
melisakuzman@fi.mdp.edu.ar

***Resumen* — La navegación autónoma en entornos cambiantes es un problema central en robótica móvil. Este trabajo presenta una metodología basada en visión por computadora y el algoritmo A* para la planificación dinámica de trayectorias. A partir de una imagen capturada por una cámara fija, se construye una representación del entorno en forma de cuadrícula, identificando obstáculos, así como la posición y orientación del robot. Sobre esta representación se ejecuta el algoritmo A*, determinando la mejor ruta hacia el destino. Cada vez que el robot alcanza una celda objetivo, el proceso se repite con información actualizada, permitiendo adaptarse a cambios en tiempo real. Esta solución resulta eficiente, de bajo costo computacional y escalable a escenarios con múltiples robots.**

Palabras clave* — *planificación de rutas, algoritmo A*, robótica móvil, entorno dinámico, visión por computadora.

I. Introducción

La planificación de rutas consiste en llevar a un robot móvil desde una posición inicial a una final, evitando obstáculos y minimizando un cierto costo asociado, como la distancia recorrida, el consumo de energía o el tiempo de desplazamiento. Cuando el entorno en el que opera el robot contiene obstáculos estáticos, la planificación puede realizarse fuera de línea antes de comenzar el movimiento. Sin embargo, en entornos dinámicos —como ocurre frecuentemente en centros logísticos, fábricas inteligentes o laboratorios de investigación— es necesario que el sistema replantee la trayectoria en tiempo real [1][2].

Este desafío ha sido abordado desde múltiples enfoques en los últimos años. Algunas estrategias se basan en algoritmos bioinspirados como algoritmos genéticos o de enjambre [3][4], mientras que otras combinan planificación estática con mecanismos de replanificación reactiva [5][6]. No obstante, muchas de estas soluciones implican un costo computacional elevado o requieren sensores complejos para la percepción del entorno, lo cual limita su implementación práctica en sistemas de bajo costo o hardware embebido.

En este trabajo se propone una solución basada en el algoritmo A* [7], ejecutado de forma periódica en función de la información del entorno obtenida a través de una cámara web. La escena capturada se procesa mediante técnicas de visión por computadora para detectar obstáculos y la posición/orientación del robot, utilizando una cuadrícula que discretiza el espacio de trabajo. A partir de esta representación, el sistema aplica el algoritmo A* para determinar la mejor trayectoria hasta el destino, indicando al robot la siguiente celda a la que debe moverse. Una vez que el robot completa ese movimiento, se repite el proceso con información actualizada, lo que permite una navegación adaptativa y en tiempo real.

Este enfoque tiene varias ventajas: es simple de implementar, ejecutable en sistemas embebidos de bajo costo, y escalable a múltiples robots interactuando en el mismo entorno. Su diseño está orientado a aplicaciones educativas, de investigación o automatización ligera, donde se requiere flexibilidad y respuesta ante condiciones dinámicas sin recurrir a infraestructuras complejas.

II. Desarrollo

A. Entorno de Trabajo

Para las pruebas experimentales se utilizó una cámara web montada sobre un arco metálico, ubicada a una altura de 90 cm respecto al suelo. En esta configuración, la cámara capta un área de trabajo de aproximadamente 130 × 100 cm, que constituye la zona observada y procesada (véase Fig. 1). La resolución de la cámara utilizada es de 320 × 240 píxeles.

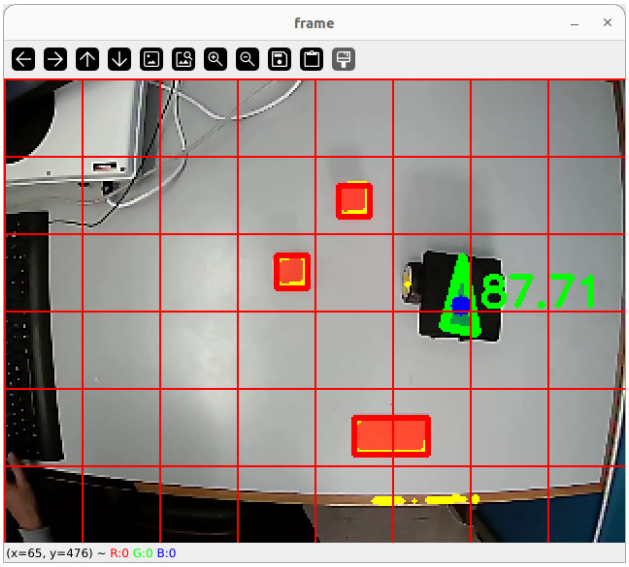

Fig. 1. Área de trabajo

Con el fin de facilitar el procesamiento de la imagen y la planificación de rutas, el área de trabajo fue dividida en una cuadrícula de 8 × 6 celdas, distribuidas uniformemente. Cada

celda fue enumerada siguiendo un patrón que permite su identificación unívoca, tal como se ilustra en la Fig. 2. Esta representación discreta del entorno permite aplicar algoritmos de búsqueda como A*, mapeando posiciones físicas reales a coordenadas lógicas dentro de la cuadrícula.

Fig. 2. Cuadrícula del área de trabajo

B. *Detección de obstáculos*

El entorno de trabajo es observado en tiempo real mediante la cámara mencionada anteriormente. A partir de la imagen capturada, se realiza un procesamiento mediante técnicas de visión por computadora para identificar la presencia de obstáculos.

Los obstáculos son marcados con un color distintivo (por ejemplo, naranja) que permite su segmentación mediante filtrado por umbral en el espacio de color HSV. Una vez binarizada la imagen, se detectan los contornos correspondientes y se determina su ubicación dentro de la cuadrícula previamente definida.

Cada celda que contiene al menos un píxel perteneciente a un contorno de obstáculo se considera ocupada y, por lo tanto, no transitable por el robot. Esta información se utiliza como entrada para el algoritmo de planificación de rutas.

C. *Representación del robot*

El robot móvil es identificado visualmente en el área de trabajo mediante un marcador distintivo de forma triangular y color verde, lo que permite su detección mediante los mismos mecanismos de filtrado por color y contornos. A partir de la forma del triángulo, se obtiene no solo la posición del robot dentro de la cuadrícula, sino también su orientación (ángulo respecto al eje horizontal).

Este enfoque permite al sistema conocer la celda actual del robot, así como su dirección de movimiento, lo cual resulta útil para ajustar las instrucciones de desplazamiento y para planificar trayectorias que consideren maniobras más eficientes.

D. *Implementación del algoritmo A**

El algoritmo A* se implementó para calcular la ruta óptima desde la celda actual del robot hasta una celda objetivo, evitando aquellas marcadas como ocupadas por obstáculos. La función de evaluación utilizada por el algoritmo combina el costo real desde el inicio hasta un nodo ($g(n)$) con una heurística que estima el costo restante hasta el objetivo ($h(n)$), utilizando como heurística la distancia de Manhattan, adecuada para entornos discretos con movimiento en cuatro direcciones.

Cada vez que el robot se desplaza a una nueva celda, se captura nuevamente la imagen del entorno, se actualiza el mapa de ocupación, y se vuelve a ejecutar el algoritmo A*. Esto permite adaptar dinámicamente la planificación ante la aparición de nuevos obstáculos o cambios en la posición de los existentes.

El resultado del algoritmo es una secuencia de celdas que el robot debe recorrer. En esta implementación, solo se transmite al robot la siguiente celda objetivo, simplificando el control de movimiento y permitiendo una ejecución más reactiva.

E. *Control del movimiento del robot*

Una vez determinada la siguiente celda objetivo mediante el algoritmo A*, se calcula el vector de desplazamiento desde la posición actual del robot hacia dicha celda. A partir de este vector y de la orientación actual del robot (obtenida por la detección del triángulo), se determina el giro necesario y el avance correspondiente para alcanzar la nueva posición.

El robot recibe instrucciones en forma de comandos simples: giro a la izquierda, giro a la derecha, avance o detenerse. Estas órdenes son transmitidas a través de una interfaz de comunicación inalámbrica y ejecutadas por un microcontrolador embebido que controla los motores del robot.

El sistema no asume movimientos perfectos, por lo que una vez ejecutado el movimiento hacia una celda, se captura nuevamente la imagen del entorno y se recalcula la trayectoria. Este enfoque permite corregir desviaciones, adaptarse a interferencias externas y responder ante la aparición de nuevos obstáculos en tiempo real.

Este control reactivo, combinado con la planificación periódica, otorga robustez al sistema sin requerir sensores de posición adicionales ni una navegación totalmente inercial o autónoma.

F. *Diagrama de flujo*

La Figura 3 muestra el diagrama de flujo correspondiente al proceso de navegación autónoma implementado en este trabajo. En él se representa la secuencia completa de pasos necesarios para la planificación dinámica y ejecución del movimiento del robot en tiempo real.

El ciclo comienza con la captura de imagen del entorno mediante una cámara montada en una posición cenital. A partir de esta imagen, se realiza el procesamiento mediante la biblioteca OpenCV para detectar y localizar los objetos relevantes del entorno: el robot y los obstáculos. La detección se basa en el análisis de color, permitiendo identificar no solo las posiciones (en coordenadas de celda) sino también el ángulo de orientación del robot.

Una vez obtenidos estos datos, se utilizan como entrada del algoritmo A*, el cual calcula la ruta óptima (mínimo costo) hasta el destino. Sin embargo, para favorecer la

replanificación dinámica y reducir la latencia, de esta secuencia solo se selecciona la meta inmediata: la celda siguiente hacia la cual debe desplazarse el robot, junto con la orientación deseada.

Con esta información, el sistema genera los comandos de control para los motores del robot, indicando velocidades y sentido de giro, y los envía mediante el protocolo MQTT. Una vez alcanzada la meta inmediata, el proceso completo se reinicia con una nueva captura de imagen, permitiendo así la adaptación continua ante cambios en el entorno.

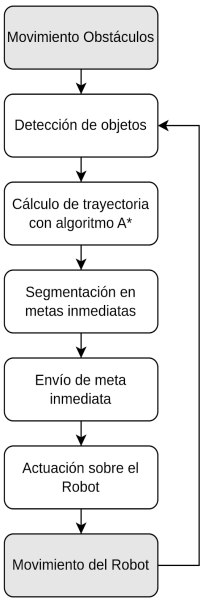

Fig. 3. Diagrama de flujo general

III. Resultados

Las pruebas experimentales se llevaron a cabo en el entorno descripto en la sección 3, con diferentes configuraciones de obstáculos y destinos. El sistema fue evaluado en términos de su capacidad para generar rutas válidas, adaptarse a cambios en el entorno, y mantener un bajo tiempo de cómputo en cada iteración.

A. Tiempo de cómputo

El algoritmo A* se ejecutó en una Raspberry Pi 400, equipada con un microprocesador Broadcom BCM2711 quad-core Cortex-A72 (ARM v8) de 64 bits a 1.8 GHz y 4 GB de memoria RAM LPDDR4-3200. Esta unidad fue responsable de ejecutar todos los procesos del sistema en tiempo real, incluyendo:

- El procesamiento de imágenes capturadas por la cámara.
- El cálculo de trayectorias mediante el algoritmo A*.
- La interfaz de usuario.
- La gestión del sistema de control a través de ROS (Robot Operating System).
- La comunicación inalámbrica con el robot mediante el protocolo MQTT (Message Queuing Telemetry Transport).

A pesar de sus especificaciones modestas en comparación con sistemas de escritorio, la Raspberry Pi 400 fue capaz de ejecutar todos estos procesos de manera fluida. El tiempo promedio requerido para una iteración completa del ciclo (captura de imagen, segmentación de obstáculos, detección del robot, cálculo de trayectoria y emisión de comandos) fue de aproximadamente 80 a 100 milisegundos, lo cual representa un tiempo despreciable frente a los tiempos mecánicos de movimiento del robot.

B. Adaptación al entorno

Durante las pruebas se introdujeron obstáculos nuevos mientras el robot se desplazaba hacia su objetivo. El sistema fue capaz de detectar dichos cambios en el entorno, invalidar rutas previamente calculadas y generar nuevas trayectorias de forma automática, sin intervención externa. Esto se logró gracias a la ejecución periódica del algoritmo A* en cada paso del robot, permitiendo una navegación adaptativa en tiempo real.

C. Precisión del movimiento

La orientación del robot se mantuvo dentro de un margen aceptable de error al utilizar detección visual basada en la forma triangular. Si bien pequeñas desviaciones angulares pueden afectar la precisión a largo plazo, el recalculo periódico de la trayectoria permite corregir estos errores acumulativos sin necesidad de sensores adicionales de posicionamiento o navegación inercial.

D. Escenarios probados

Se ensayaron distintos escenarios, incluyendo:

- Entorno libre de obstáculos (trayectoria directa).
- Obstáculos estáticos en el trayecto (recalculo con desvíos).
- Obstáculos que aparecen durante el movimiento (replanificación dinámica).

En todos los casos, el sistema logró llevar al robot al destino deseado, sin colisiones y con trayectorias eficientes.

IV. Conclusiones

Este trabajo presentó una solución práctica y eficiente para la planificación de rutas dinámicas en entornos observados mediante visión artificial. La combinación del algoritmo A* con una estrategia de replanificación periódica permitió al robot adaptarse en tiempo real a cambios en el entorno, garantizando trayectorias seguras y eficientes.

Entre las principales ventajas del enfoque propuesto se destacan:

- Bajo costo computacional, adecuado para sistemas de hardware limitado.
- Robustez ante cambios en el entorno, gracias a la detección visual y replanificación continua.
- Simplicidad de implementación, sin necesidad de sensores de navegación costosos.

Las pruebas experimentales demostraron que el sistema es capaz de operar de forma estable y confiable en una variedad de escenarios, incluyendo la aparición inesperada de obstáculos. Esta solución resulta especialmente atractiva para entornos controlados como almacenes, laboratorios o entornos educativos.

Como líneas de trabajo futuro se propone:

- Incorporar marcadores planares (fiduciales) para mejorar la precisión en la identificación y seguimiento de objetos en espacios tridimensionales, facilitando una detección más

robusta ante oclusiones o variaciones de iluminación.

- Explorar estrategias de planificación multirobot, permitiendo la coordinación simultánea de varios agentes móviles en el mismo entorno, con replanificación colaborativa ante interferencias o conflictos de trayectorias.
- Implementar un sistema de gestión de tareas, como el traslado de cargas o elementos entre posiciones dentro del área de trabajo, integrando aspectos de logística interna y asignación dinámica de objetivos.

REFERENCIAS

[1] J. P. van den Berg and M. H. Overmars, "Roadmap-based motion planning in dynamic environments," 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sendai, Japan, 2004, pp. 1598–1605, vol. 2, doi: 10.1109/IROS.2004.1389624.

[2] S. Koenig and M. Likhachev, "D* Lite," in Proceedings of the AAAI Conference on Artificial Intelligence, 2002.

[3] A. Tuncer and M. Yildirim, "Dynamic path planning of mobile robots with improved genetic algorithm," Computers & Electrical Engineering, vol. 38, no. 6, pp. 1564–1572, 2012.

[4] A. T. Mathew, A. Paul, A. Rojan and A. Thomas, "Implementation of Swarm Intelligence Algorithms for Path Planning," Journal of Physics: Conference Series, vol. 1831, no. 1, 2021, doi: 10.1088/1742-6596/1831/1/012008.

[5] D. Lin, B. Shen, Y. Liu, F. E. Alsaadi and A. Alsaedi, "Genetic algorithm-based compliant robot path planning: an improved bi-RRT-based initialization method," Assembly Automation, vol. 37, no. 3, pp. 301–309, 2017.

[6] Y. C. Pradeep, Z. Ming, M. Del Rosario and P. C. Y. Chen, "Human-inspired robot navigation in unknown dynamic environments," 2016 IEEE International Conference on Mechatronics and Automation (ICMA), Harbin, China, 2016, pp. 971–976, doi: 10.1109/ICMA.2016.7558694.

[7] P. E. Hart, N. J. Nilsson and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100–107, 1968, doi: 10.1109/TSSC.1968.300136.