UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN MECATRÓNICA

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

TITLE:

“MOBILE ROBOT FOR RESEARCH ON PATH PLANNING ALGORITHMS: PLANNING SYSTEM”

TEMA:

“ROBOT MÓVIL PARA INVESTIGACIÓN EN ALGORITMOS DE PLANEAMIENTO DE RUTAS: SISTEMA DE PLANEAMIENTO DE RUTAS”

AUTOR:

IVÁN DARÍO ACOSTA TORRES

DIRECTOR:

ING. XAVIER ROSERO, MSc.

IBARRA – ECUADOR

2017

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

La Universidad Técnica del Norte dentro del Proyecto Repositorio Digital Institucional determinó la necesidad de disponer de textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la universidad.

Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto, para lo cual pongo a disposición la siguiente información:

DATOS DEL CONTACTO	
CEDULA DE IDENTIDAD	100337633-0
APELLIDOS Y NOMBRES	ACOSTA TORRES IVÁN DARÍO
DIRECCIÓN	LA VICTORIA
E-MAIL	idacostat@utn.edu.ec
TELÉFONO MÓVIL	0982958652
DATOS DE LA OBRA	
TÍTULO	“MOBILE ROBOT FOR RESEARCH ON PATH PLANNING ALGORITHMS: PLANNING SYSTEM”
AUTOR	IVÁN DARÍO ACOSTA TORRES
FECHA	2017/12/12
PROGRAMA	PREGRADO
TÍTULO POR EL QUE OPTA	INGENIERO EN MECATRÓNICA
ASESOR	ING. XAVIER ROSERO, MSc.

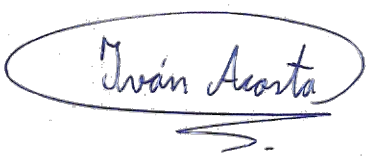
2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD

Yo, Iván Darío Acosta Torres con cédula de identidad Nro. 100337633-0, en calidad de autor y titular de los derechos patrimoniales del trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en formato digital y autorizo a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y uso del archivo digital en la biblioteca de la Universidad con fines académicos, para ampliar la disponibilidad del material y como apoyo a la educación, investigación y extensión; en concordancia con la Ley de Educación Superior Artículo 144.

3. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se desarrolló sin violar derechos de autor de terceros, por lo tanto la obra es original, y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, 12 de diciembre de 2017.



……

Iván Darío Acosta Torres

C.I.: 100337633-0

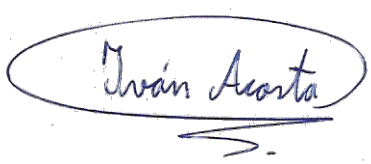
UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE GRADO A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

Yo, Iván Darío Acosta Torres con cédula de identidad Nro. 100337633-0, manifiesto mi voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la Ley de Propiedad Intelectual del Ecuador, artículos 4, 5 y 6, en calidad de autor (es) del trabajo de grado denominado: "MOBILE ROBOT FOR RESEARCH ON PATH PLANNING ALGORITHMS: PLANNING SYSTEM", que ha sido desarrollada para optar por el título de: Ingeniero en Mecatrónica, en la Universidad Técnica del Norte, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital a la biblioteca de la Universidad Técnica del Norte.

Ibarra, 12 de diciembre de 2017.



……

Iván Darío Acosta Torres

C.I.: 100337633-0

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

DECLARACIÓN

Yo, Iván Darío Acosta Torres, con cédula de identidad N°. 100337633-0, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado en ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo los derechos de propiedad intelectual correspondientes a este trabajo, a la Universidad Técnica del Norte, según lo establecido por las Leyes de la Propiedad Intelectual, Reglamentos y Normativa vigente de la Universidad Técnica del Norte.

Ibarra, 12 de diciembre de 2017.

……

Iván Darío Acosta Torres

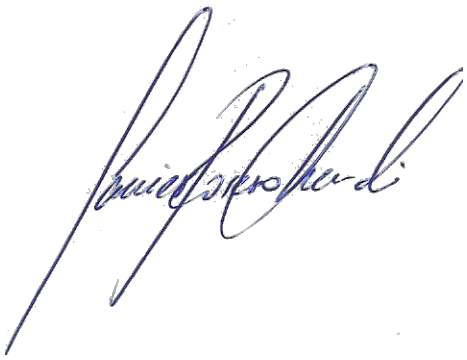
C.I.: 100337633-0

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICACIÓN

En calidad de Director del Trabajo de Grado previo a la obtención del título de Ingeniero en Mecatrónica con el tema: “MOBILE ROBOT FOR RESEARCH ON PATH PLANNING ALGORITHMS: PLANNING SYSTEM”, ha sido desarrollado y terminado en su totalidad por el Sr. Iván Darío Acosta Torres, con cédula de identidad: 100337633-0, bajo mi supervisión para lo cual firmo en constancia.



……

Ing. Xavier Rosero, MSc.

DIRECTOR DEL PROYECTO

DEDICATORIA

De manera especial a mi madre Patricia, por todo el apoyo que me brindo durante mi formación personal y profesional; llenándome de fortaleza, sabiduría, valores y mucho amor, para seguir adelante cada día. Sin ella no sería la persona que soy hoy en día.

Iván A.

AGRADECIMIENTO

Agradezco a mi familia por ser parte fundamental en mi vida; gracias por su apoyo y por confiar en mí.

A mis mejores amigos que siempre han estado presentes en los buenos y malos momentos.

A la Universidad Técnica del Norte por abrirme sus puertas y permitirme formar como profesional; a mis maestros que me han brindado sus enseñanzas a lo largo de esta carrera.

Iván A.

RESUMEN

Hoy en día la robótica móvil es sujeto de investigación y desarrollo de nuevas tecnologías debido a la necesidad de tener autómatas que realicen tareas de alto nivel sin la intervención humana; hay que considerar también que ha llegado a formar parte del servicio y entretenimiento de las personas en su diario vivir.

Entre los problemas más importantes de la robótica móvil se encuentra la planificación de trayectorias, que desempeña una tarea fundamental de la navegación autónoma del robot en un entorno ya sea estático o dinámico. Por tal motivo, este proyecto presenta el estudio y simulación de algoritmos de planificación de trayectorias de un robot móvil en un entorno estático.

Mediante una búsqueda de información objetiva sobre planificación de trayectorias y algoritmos de planificación se determinó que los más usados son el algoritmo A*, que usa una función heurística con la que se guía en la búsqueda de la ruta; y el algoritmo de Dijkstra, que mediante el incremento del costo de paso por los nodos, determina la ruta.

La interfaz gráfica del programa de simulación se realiza en MATLAB con el uso de su herramienta GUIDE. La interfaz de la ventana de simulación del algoritmo permite crear un mapa de rejilla en el que se pueden colocar las posiciones de inicio y llegada del robot, así como las posiciones de los obstáculos a través de los cuales el robot debe encontrar y graficar la ruta óptima.

Los resultados de este proyecto reflejan la correcta aplicación de los algoritmos, ya que cada uno de ellos cumple con la función de encontrar la ruta con costo óptimo. Al realizar el análisis del tiempo de procesamiento (simulaciones en mapas similares para cada algoritmo) se

puede comprobar que el algoritmo A* es más eficiente que el de Dijkstra, ya que presenta tiempos menores.

ABSTRACT

Nowadays mobile robotics is the subject of research and development of new technologies due to the need to have robots that perform high level tasks without human intervention; we must also consider that it has become part of the service and entertainment of people in their daily lives.

Among the most important problems of mobile robotics is trajectory planning, which plays a fundamental role in autonomous robot navigation in a static or dynamic environment. For this reason, this project presents the study and simulation of algorithms for planning trajectories of a mobile robot in a static environment.

By means of a search of objective information on planning of trajectories and algorithms of planning it was determined that the most used are the algorithm A *, that uses a heuristic function with which it is guided in the search of the route; and the Dijkstra algorithm, which by increasing the cost of passing through the nodes, determines the route.

The graphical interface of the simulation program is done in MATLAB with the use of its GUIDE tool. The interface of the simulation window of the algorithm allows to create a grid map in which the robot start and finish positions can be placed, as well as the obstacle positions through which the robot must find and plot the route optimal.

The results of this project reflect the correct application of the algorithms, since each of them fulfills the function of finding the route with an optimal cost. When performing the analysis of the processing time (simulations in similar maps for each algorithm) it can be verified that the algorithm A * is more efficient than the one of Dijkstra, since it presents smaller times.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ECUACIONES

INTRODUCCIÓN

PLANTEAMIENTO DEL PROBLEMA

En la actualidad existen varias plataformas de desarrollo en el área de la robótica móvil que son usadas para varias aplicaciones entre las que se encuentra la planificación de trayectorias. Sin embargo, estas plataformas tienen un costo demasiado elevado y son cerradas ya que no permiten actualización en su hardware. Es necesario tomar en consideración que la robótica móvil ha aportado con grandes soluciones para problemas cotidianos referentes a la navegación autónoma.

Hay que tener presente que dentro de la Universidad Técnica del Norte y especialmente en la Facultad de Ingeniería en Ciencias Aplicadas, no existen plataformas de robots móviles, ya sean orientados para el uso académico o como base para el desarrollo de investigación. La causa para que no exista investigación dentro del área de la robótica móvil es atribuible a la inexistencia de plataformas que fomenten el desarrollo de estudio e investigación.

Por otro lado, en los laboratorios de mecatrónica existen hardware y software poco aprovechados, como son ciertas tarjetas con sistemas embebidos de gran capacidad computacional y software propietario con módulos de robótica. Usando éstos se obtendría un robot con costo bajo y de fácil construcción, cuya arquitectura podría ser escalable tanto en software como en hardware, diferenciándose notablemente de las soluciones presentes en el mercado local.

OBJETIVOS

OBJETIVO GENERAL

Desarrollar la simulación del sistema de planificación de trayectoria de un robot móvil.

OBJETIVOS ESPECÍFICOS

- Conocer el estado del arte sobre algoritmos de planificación de trayectorias.
- Analizar los algoritmos de planificación escogidos.
- Implementar el software de simulación de planificación.
- Realizar pruebas de validación.

ANTECEDENTES

En la Universidad Técnica del Norte no se ha encontrado temas similares al propuesto, por lo que se tomará como referencia tesis y trabajos realizados dentro y fuera del país. Por ejemplo, Arellano en la Universidad Nacional Mayor de San Marcos en Lima Perú en cambio construye un robot de dos ruedas que es monitoreado en tiempo real por una computadora que a través de una interfaz gráfica desarrollada en Java permite observar los parámetros de control del robot; la implementación del algoritmo de control lo hace en un dsPIC por las características que presenta y así poder obtener la estimación odométrica; para comunicar el robot con la PC utiliza comunicación serial RS-232; utiliza un controlador PID para la regularización de la posición y orientación angular del robot [1].

Benavides en la Universidad de la Republica en Montevideo Uruguay realiza la programación en java en el software Eclipse, realiza una propuesta que se basa en la

representación del espacio de configuraciones libres a través de la construcción de roadmaps, muestreo aleatorio de configuraciones en torno a las aristas del roadmap y búsqueda de caminos óptimos con un algoritmo genético; no especifica si construyó o no el robot [2].

Zambrano en la Escuela Politécnica Nacional en Quito Ecuador utiliza la plataforma Robotino, que es una plataforma educativa de robot autónomo de alto costo fabricado por Festo; se aplica un modelo cinemático para determinar el controlador a usar; la planificación de ruta la realiza con el método de la construcción de grafos de conectividad, método por campos potenciales, método de descomposición de celdas; realiza la programación en LabVIEW [3].

Yandún en la Escuela Politécnica Nacional en Quito Ecuador realizó la selección del método por el cual el robot realizará la planificación de trayectorias, en este caso escogió los diagramas de Voronoi debido a la seguridad que presenta la trayectoria al considerarse a esta lo más alejado de los obstáculos; usa LabVIEW como software para desarrollar la programación de la planificación de la trayectoria; usa la plataforma Robotino que es una plataforma educativa de robot autónomo de la marca Festo; la comunicación con este robot es mediante wireless [4].

JUSTIFICACIÓN

Para lograr la autonomía de un vehículo, este debe tener la capacidad de conocer su posición dentro de su ambiente. Un vehículo que no pueda localizarse a sí mismo corre el riesgo de chocar contra obstáculos, elegir las rutas inadecuadas o no poder evitar las áreas peligrosas. Esas son algunas de las razones por las cuales el problema de la localización es importante.

La relevancia del dispositivo propuesto consiste en una investigación aplicativa que servirá como desarrollo de una plataforma base para investigación, estudio de sistemas robóticos móviles con planificación de trayectorias, y así obtener conocimiento práctico y no sólo teórico,

como indicador para exponer el nivel de conocimientos y el gran potencial innovador de los estudiantes de la UTN.

La implementación de este sistema se basa en nociones, habilidades, capacidades, destrezas, aptitudes, vinculadas a la competencia profesional y a los conocimientos adquiridos por el Ingeniero Mecatrónico.

ALCANCE

El proyecto a realizarse, consistirá en el desarrollo de una plataforma robótica móvil usando un sistema embebido de alta capacidad computacional conectada a una interfaz humano-robot.

Se simulará un ambiente con obstáculos y se resolverá la planificación del movimiento del robot desde una posición inicial, hasta una posición requerida. Los algoritmos de planificación serán los más simples encontrados en el estado del arte.

CAPÍTULO I

MARCO TEÓRICO

En el desarrollo de este capítulo se explicará todas las bases teóricas requeridas sobre la planificación de trayectorias en la robótica móvil, conociendo los algoritmos que se puede usar y el entorno de trabajo de los mismos; conceptos necesarios para el desarrollo de la investigación.

1.1. NAVEGACIÓN

"La navegación es la ciencia de conducir un robot móvil mientras atraviesa un entorno para alcanzar un destino o meta sin chocar con ningún obstáculo" [5].

Para la creación de una trayectoria viable y segura la navegación nos lleva a crear una serie de herramientas necesarias; se ha de crear un mapa con la información que se tiene o se lo irá creando según se explore [5].

Muchas tareas robóticas se pueden lograr sin usar ningún mapa en absoluto, utilizando un llamado método de navegación reactiva, la misma que se refiere a la reacción directa del robot con su entorno: la intensidad de la luz, la posición de una línea blanca o el contacto con una pared [6].

1.1.1. PROBLEMAS DE LA NAVEGACIÓN

Relacionado con la navegación de los robots móviles surgen tres problemas. La auto localización del robot, en la que el robot recurre a la información obtenida de sus sensores para determinar su posición inicial con relación a su entorno, sería el primer problema; identificar la posición del objetivo con relación al entorno, sería el segundo problema y el tercer problema está

relacionado con la planificación de la trayectoria, que permite establecer un camino posible desde la posición inicial a la posición del objetivo esquivando obstáculos y obviamente que el robot tenga la capacidad de cumplir físicamente la trayectoria [7].

1.1.2. ETAPAS DE LA NAVEGACIÓN

Para que un robot pueda efectuar la tarea de navegación es necesario conocer las etapas que se describirán a continuación:

- Percepción del medio: Esta se da a través del uso de sensores externos, un mapa del entorno o un modelo del ambiente que se emplea para la navegación.
- Planificación de la ruta: Se basa en el mapa de entorno, estableciendo una secuencia ordenada de posiciones hasta el punto de meta mediante el uso de algún procedimiento estratégico.
- Generación del camino: Se encarga de incluir la secuencia de posiciones definidas en la planificación de la ruta y procede a la discretización de la secuencia para generar el camino.
- Seguimiento de la trayectoria: Se trata del desplazamiento del robot móvil de acuerdo al camino generado, a través de los actuadores del robot móvil.

Las etapas descritas pueden llevarse a cabo de forma separada, aunque en el orden especificado. La interrelación entre cada una de estas etapas permite el control de la navegación en el robot móvil [4].

1.2. PLANIFICACIÓN DE TRAYECTORIAS

“Planificar es prever y decidir hoy las acciones que nos pueden llevar desde el presente hasta un futuro deseable, no se trata de hacer predicciones acerca del futuro sino de tomar las decisiones pertinentes para que ese futuro ocurra” [4].

La trayectoria es una secuencia de puntos o configuraciones determinadas por un algoritmo para circular por el mapa de entorno, que son procesados posteriormente para que un robot realice el seguimiento de estos utilizando expresiones matemáticas para alcanzar la meta [3].

El término planificación de trayectorias se ha desarrollado en muchos campos, como la robótica, la inteligencia artificial o la teoría de control; por tal motivo cada científico utiliza la definición propia de este término [8].

En la robótica, la planificación de rutas se refiere al problema de cómo mover un robot de un punto a otro. En la inteligencia artificial, la planificación de trayectoria significa una búsqueda de una secuencia de acciones lógicas que transforman una posición inicial del robot en una posición objetivo deseado. En la teoría de control, la planificación de trayectos se ocupa de cuestiones de estabilidad, retroalimentación y optimización. Es por eso que la planificación de trayectorias de un robot móvil es un problema amplio y existen muchos métodos y enfoques [8].

Correll [9], refiere que la planificación de trayectorias es algo muy importante para los robots móviles autónomos, permite a los robots encontrar la trayectoria más corta u óptima entre dos puntos. Una trayectoria óptima podría ser un camino en el que se disminuya la cantidad de giros, la cantidad de frenado o lo que una aplicación específica requiera. Por lo tanto “la

planificación de trayectorias requiere un mapa del entorno y del robot para conocer su ubicación con respecto al mapa".

1.2.1. TIPOS DE ENTORNO PARA LA PLANIFICACIÓN

La planificación de trayectorias de robots móviles puede darse en los siguientes tipos de entornos que son estáticos y dinámicos. En el caso estático los obstáculos no se mueven y se tiene total conocimiento del espacio de trabajo, mientras para el caso dinámico los obstáculos son considerados móviles y por lo tanto no se conoce completamente el espacio de trabajo [10].

Martínez, Barrero y Tibaduiza [11], refieren que cuando el ambiente es estático realizan la planificación off-line, al no presentar variaciones en el espacio de trabajo durante toda la trayectoria; mientras que cuando el ambiente es dinámico, el espacio de trabajo cambia continuamente y obliga al algoritmo a generar nuevas rutas realizando la planificación de manera on-line.

Para la planificación off-line se requiere toda la información acerca del entorno, en la que, se detalle con exactitud la localización de los obstáculos; para este caso, la planificación de la trayectoria solo se realiza una vez; mientras que para la planificación on-line la información del entorno es incompleta, por lo que, la planeación de la trayectoria es un proceso continuo que para generar las acciones de control se recurre a la información del entorno entregada por los sensores del robot móvil; ya no es necesario que se conozca la localización y forma de los obstáculos [7].

1.2.2. REPRESENTACIONES DE MAPAS

Para planear una trayectoria, es necesario que se represente el ambiente en la computadora; se distinguen entre dos enfoques complementarios: aproximaciones discretas y continuas.

En la aproximación discreta el mapa se subdivide en trozos de igual o de diferente tamaño; los mapas discretos se prestan bien a una representación gráfica ya que cada trozo del mapa corresponde a un vértice o nodo, que están conectados por los bordes y así un robot puede navegar por los vértices [9].

En la aproximación continua se requiere la definición de límites interiores que serían los obstáculos y límites externos, por lo general en forma de polígono, mientras que los recorridos pueden ser codificados como secuencias de puntos definidos por números. A pesar de las ventajas que tiene una representación continua por memoria, los mapas discretos son los que dominan en la robótica [9].

Para mapear obstáculos el mapa más común es el mapa de cuadrícula de ocupación; en un mapa de este tipo el entorno se discretiza en cuadrados de resolución arbitraria, en el que se marcan los obstáculos. “Las desventajas de los mapas de cuadrícula son sus grandes requerimientos de memoria, así como el tiempo computacional para atravesar estructuras de datos con un gran número de vértices” [9].

Un mapa de cuadrícula induce un gráfico donde cada nodo corresponde a una celda y un borde conecta nodos de las celdas que son continuas; la conectividad de cuatro puntos solo

tendrá bordes al norte, sur, este y oeste mientras que la conectividad de ocho puntos tendrá bordes a todas las celdas que rodean a la celda actual [12].

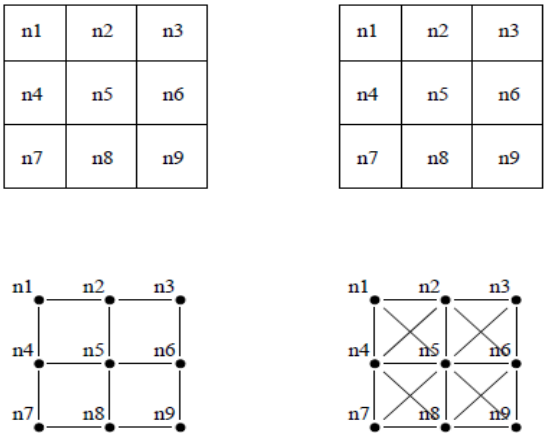


Figura 1. Conectividad entre puntos.

Fuente: [12]

En la figura 1, la conectividad de cuatro puntos supone sólo cuatro vecinos, mientras que la conectividad de ocho puntos tiene ocho.

1.3. ALGORITMOS DE PLANIFICACIÓN DE TRAYECTORIAS

El problema de encontrar una trayectoria más corta de un vértice a otro a través de un gráfico conectado es de interés en múltiples campos. El término más corto se refiere aquí al costo de borde acumulativo mínimo, que podría ser la distancia física (en una aplicación robótica) o cualquier otra métrica que sea importante para una aplicación específica. Un gráfico de ejemplo con longitudes de borde arbitrarias se muestra en la figura 2 [9].

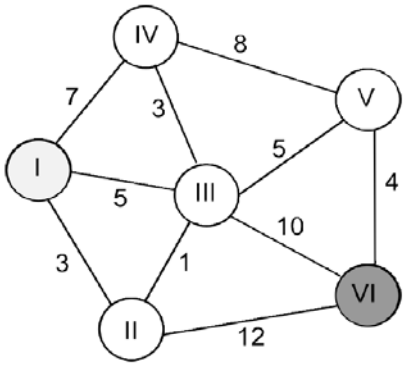


Figura 2. Problema de planificación de trayectoria.

Fuente: [9]

En la figura 2, se puede observar que es un problema de planificación de trayectoria que va desde el vértice I hasta el vértice VI; el camino más corto para este ejemplo es I-II-III-V-VI con una longitud de 13.

Dentro de la robótica, el enfoque está en diseñar algoritmos que generen movimientos útiles procesando modelos geométricos complicados. Dentro de la inteligencia artificial, el enfoque se centra en el diseño de sistemas que utilizan modelos de teoría de la decisión para calcular las acciones apropiadas. Dentro de la teoría del control, el enfoque está en los algoritmos que calculan las trayectorias factibles para los sistemas, con una cierta cobertura adicional de la regeneración y de la optimización [13].

1.4. ALGORITMO A* (A STAR)

El algoritmo de búsqueda A* fue presentado por Hart, Nilsson y Raphael en 1968, por el problema de determinar la ruta de costo mínimo a través de un gráfico [14].

En robótica, el algoritmo A* se emplea principalmente en una red ambiental que se incluye el conocimiento de la función heurística h(n). La función heurística seleccionada como función de distancia nos brinda la distancia entre la celda en expansión actual y la meta [15].

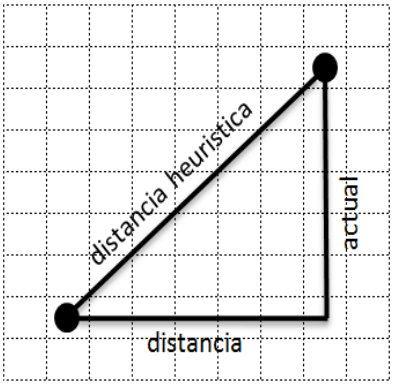


Figura 3. Distancia heurística.

Fuente: [12]

Esta función heurística entre dos nodos es la distancia euclidiana, que es menor que la longitud real de la trayectoria en la cuadrícula; la función hace que el algoritmo sea eficiente para obtener conocimiento adicional sobre la cuadrícula [12] [15].

Se evalúa la función de evaluación de costos de cada celda mediante

$$f(n) = g(n) + h(n) \qquad (1)$$

donde $n = (x_n, y_n)$ es el nodo o celda en expansión actual; $f(n)$ es el costo mínimo estimado de la celda entre todos los caminos desde el nodo de inicio S al nodo de meta G forzado a pasar por el nodo n; $g(n)$ es la función de costo real desde el nodo de inicio $S(x_S, y_S)$ hasta el nodo de expansión actual n; $h(n)$ es la estimación heurística del costo mínimo de una trayectoria desde el nodo de expansión actual n al nodo objetivo $G(x_G, y_G)$ [15].

La distancia d entre dos puntos viene dada por

$$d = \sqrt{(x_n - x_G)^2 + (y_n - y_G)^2} \quad (2)$$

y $h(n)$ en (1) puede estimarse como

$$h(n) = \sqrt{(x_n - x_G)^2 + (y_n - y_G)^2} \quad (3)$$

esta $h(n)$ es la distancia euclidiana entre el nodo objetivo y el nodo en expansión actual. Ahora la solución de menor costo se puede averiguar con la ayuda de las ecuaciones anteriores y retrocedidas desde la posición de meta G a la posición de inicio S [15].

Correl [9], explica el algoritmo A* de la siguiente manera: En lugar de explorar en todas las direcciones, el conocimiento de una dirección aproximada de la meta podría ayudar a evitar la exploración de nodos que son obviamente erróneos para un observador humano. Tal conocimiento especial que un observador tiene puede ser codificado usando una función heurística. Por ejemplo, se podría dar prioridad a los nodos que tienen una distancia estimada menor a la meta que otros. Para ello, marcaríamos cada nodo no solo con la distancia real que nos llevó a llegar, sino también con el costo estimado, por ejemplo, calculando la distancia euclidiana entre el vértice que estamos viendo y el vértice objetivo.

Lo descrito anteriormente se puede resumir en el diagrama de la figura 4:

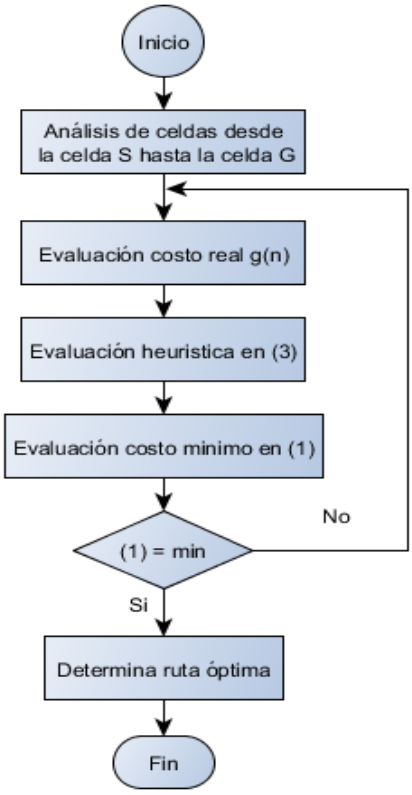


Figura 4. Diagrama algoritmo A.*

1.4.1. EJEMPLO ALGORITMO A*

El ejemplo en la figura 5, presenta el valor de la heurística h(n) en sus nodos y el valor del costo de recorrido g(n) en los bordes.

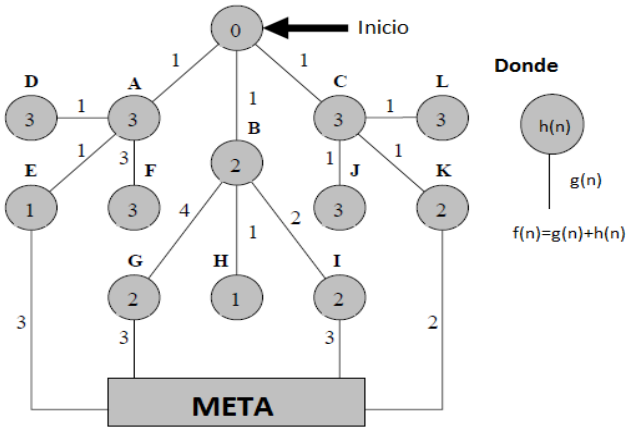


Figura 5. Ejemplo del algoritmo A.*

Para el primer paso, existen las siguientes opciones:

- {Inicio, A} con f(n) = 1 + 3 = 4
- {Inicio, B} con f(n) = 1 + 2 = 3
- {Inicio, C} con f(n) = 1 + 3 = 4

Para el segundo paso, se explora primero el camino que se estima es el más corto:

- {Inicio, B, G} con f(n) = 5 + 2 = 7
- {Inicio, B, I} con f(n) = 3 + 2 = 5

Como se puede observar en la figura 5, a partir del nodo I que sería el camino más corto, ya se puede llegar a la meta con un costo de recorrido de 6.

Se continúa explorando el camino desde el nodo A, que nos da una ruta hacia la meta:

- {Inicio, A, D} con f(n) = 2 + 3 = 5
- {Inicio, A, E} con f(n) = 2 + 1 = 3
- {Inicio, A, F} con f(n) = 4 + 3 = 7

El nodo E, presenta un valor de f(n) menor que el del nodo I, por lo tanto tiene prioridad y el costo de recorrido a la meta es de 5.

Se explora el camino desde el nodo C, que igualmente presenta una ruta hasta la meta:

- {Inicio, C, J} con f(n) = 2 + 3 = 5
- {Inicio, C, K} con f(n) = 2 + 2 = 4
- {Inicio, C, L} con f(n) = 2 + 3 = 5

El nodo K, presenta un valor de f(n) menor que el nodo I e igualmente se calcula el costo de recorrido hacia la meta que es de 4.

Después de haber realizado la exploración de las distintas rutas, se determina que la ruta con el costo de recorrido mínimo sería por los nodos {Inicio, C, K, Meta} como se muestra en la figura 6, con un costo de 4.

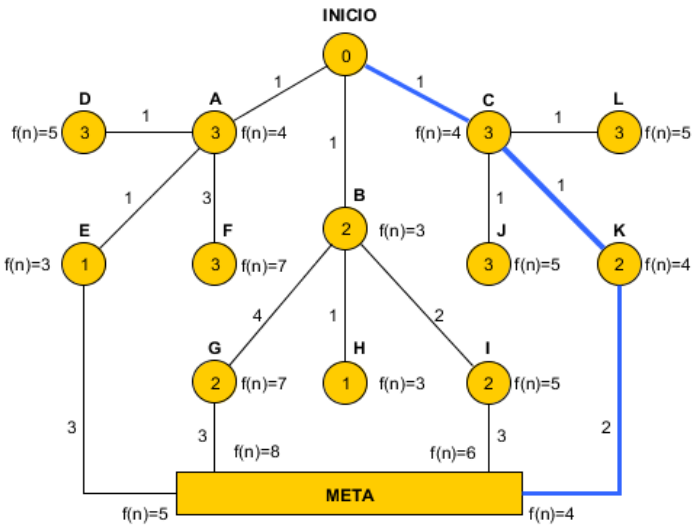


Figura 6. Solución ejemplo algoritmo A.*

1.5. ALGORITMO DE DIJKSTRA

El algoritmo de Dijkstra fue presentado por el holandés Edsger Wybe Dijkstra en 1959, quien lo describió por el problema de construir un árbol de longitud total mínima entre “n” nodos y de encontrar la ruta de longitud total mínima entre dos nodos [16].

Es un algoritmo clásico para resolver el problema del camino más corto, es simple y fácil, un método excelente para la trayectoria del camino del robot y ha sido ampliamente utilizado en optimización de red, transporte, logística, electrónica y otros campos [17].

Correll [9] explica el algoritmo de Dijkstra de la siguiente manera: A partir del vértice inicial donde debe iniciarse la ruta, el algoritmo marca todos los vecinos directos del vértice inicial con el costo para llegar allí. Luego procede desde el vértice con el menor costo a todos sus vértices adyacentes y los marca con el costo de llegar a ellos por sí mismo; si este costo es menor. Después que se han comprobado todos los vecinos de un vértice, el algoritmo procede al vértice siguiente con el costo más bajo. Una vez que el algoritmo alcanza el vértice de la meta, termina y el robot puede seguir los bordes que apuntan hacia el costo del borde más bajo.

Lo descrito anteriormente se puede resumir en el diagrama de la figura 7:

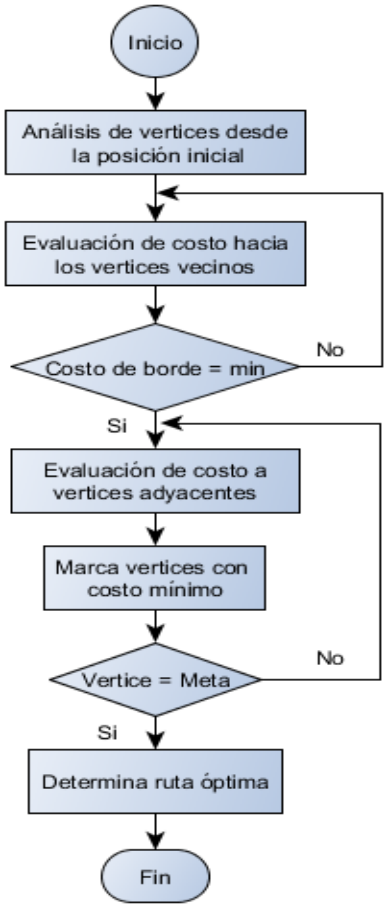


Figura 7. Diagrama algoritmo de Dijkstra.

Como Dijkstra siempre explorará nodos con el menor costo total primero, el entorno se explora comparativamente a un frente de onda que se origina desde el vértice de inicio, llegando finalmente a la meta. Esto es altamente ineficiente en particular si Dijkstra está explorando nodos lejos de la meta [9].

1.5.1. EJEMPLO ALGORITMO DE DIJKSTRA

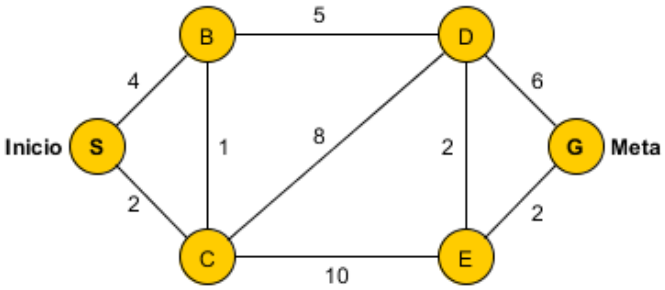


Figura 8. Ejemplo del algoritmo de Dijkstra.

En la tabla 1, podemos observar la solución del ejemplo de la figura 8. Se resuelve paso a paso cual es el costo de recorrido por cada vértice, tomando en cuenta el valor mínimo.

Tabla 1

Solución ejemplo del algoritmo de Dijkstra

	Paso 1	**Paso 2**	**Paso 3**	**Paso 4**	**Paso 5**	**Paso 6**
S	**(0,--)**	*	*	*	*	*
B	(4,S)	(3,C)	**(3,C)**	*	*	*
C	(2,S)	**(2,S)**	*	*	*	*
D	∞	(10,C)	(8,B)	**(8,B)**	*	*
E	∞	(12,C)	(12,C)	(10,D)	**(10,D)**	*
G	∞	∞	∞	(14,D)	(12,E)	**(12,E)**

En la figura 9, podemos observar cual es la ruta mínima encontrada desde el vértice S hasta el vértice G.

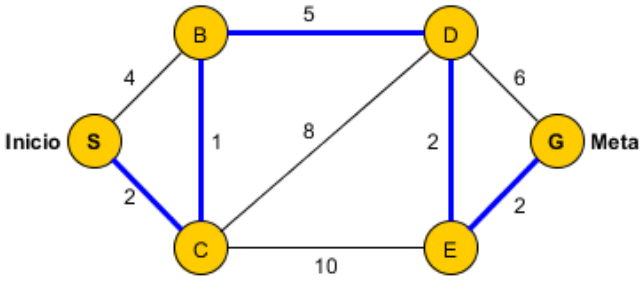


Figura 9. Solución algoritmo de Dijkstra.

CAPÍTULO II

METODOLOGÍA

A continuación se detalla todo lo referente al desarrollo del trabajo de grado, partiendo desde la investigación, el análisis de los algoritmos, la realización de la interfaz gráfica de simulación que está dividida desde una ventana principal y dos ventanas secundarias de cada algoritmo; se determina el funcionamiento de los algoritmos del simulador de planificación de trayectorias.

2.1. INVESTIGACIÓN

Se realizó la búsqueda de información respecto al tema procedente de trabajos de grado, libros y artículos científicos que permitan tener un mayor conocimiento acerca de la robótica móvil, planeación de trayectorias y los algoritmos que pueden ser usados. Se investigó sobre el software que se usó para la programación de los algoritmos y creación de la interfaz gráfica necesaria para el simulador.

2.2. SELECCIÓN DE LOS ALGORITMOS

Los algoritmos a usar son el A* y el de Dijkstra por ser de los más usados en investigaciones de robótica móvil y son de los primeros algoritmos en aprender para problemas de planeación de trayectorias; estos dos algoritmos son para gráficos y sirven para el tipo de simulación en mapas de cuadricula [18].

El algoritmo A* tiene las siguientes características:

- Es uno de los algoritmos de planificación más usados, ya que de existir una solución, busca la ruta más corta entre dos nodos.
- Usa una función heurística entre dos nodos que sirve como referencia para evaluar el camino con el menor costo.
- La función de evaluación de la ruta viene dado por la sumatoria del costo del camino recorrido y el valor de la heurística en (1), para hallar la ruta óptima.
- Es óptimamente eficiente ya que expande sólo los nodos que tienen el menor valor en la función de evaluación, haciendo que su búsqueda sea más rápida.
- Evita caminos que no necesitan ser evaluados.
- Es considerado un algoritmo completo.

El algoritmo de Dijkstra tiene las siguientes características:

- Este algoritmo es considerado como uno de los más relevantes en la teoría de grafos.
- Es un algoritmo conocido para la búsqueda de ruta óptima entre dos nodos.
- Sirve para determinar el camino más corto dado un nodo origen al resto de nodos.
- Es eficiente pero puede llegar a desperdiciar tiempo analizando nodos que no son necesarios o que se encuentran lejos del nodo de llegada.
- No trabaja con costos de aristas negativos.
- Es considerado un algoritmo completo.

2.3. PARTES DE LA INTERFAZ GRÁFICA

Para explicar cómo es la interfaz gráfica desarrollada para la simulación, es necesario identificar las partes de las que consta la interfaz. Se inicia con una ventana principal que es de

selección en la que se tiene la opción de escoger el tipo de algoritmo, una vez que se ha seleccionado el algoritmo se abre una nueva ventana, la misma que es idéntica para los dos tipos de algoritmos y en la que ya podemos realizar cualquier ejercicio de simulación de planificación.

2.4. CARACTERÍSTICAS DE LA INTERFAZ DE SIMULACIÓN

La interfaz de simulación consiste en mostrar un mapa, que está representado por una cuadrícula de ocupación, la misma que aparece en la ventana de la interfaz después de escoger el tipo de algoritmo; la ventana presenta las opciones para ingresar la coordenada (X,Y) del punto de llegada del robot móvil, así como las coordenadas de los puntos que son ocupados por los obstáculos que debe evadir el robot.

Un botón es el que da inicio al funcionamiento del algoritmo así como también un botón se encarga de limpiar el mapa y otro para salir de la ventana de simulación.

El robot es representado por un punto sin dimensiones y se coloca en la mitad de los cuadros blancos del mapa, los mismos en los que el robot tiene la capacidad de moverse; la posición de inicio del robot móvil es por defecto en la parte inferior izquierda de la cuadrícula o mapa en la coordenada (1,1); los obstáculos son cuadrados negros que no pueden ser ocupados por el robot.

2.5. DESARROLLO DE LA INTERFAZ GRÁFICA

La interfaz se realizó en MATLAB, que es una herramienta muy utilizada para aplicaciones de ingeniería; nos da facilidad en el trabajo con matrices, en la representación tanto de datos como de funciones, permite la implementación de algoritmos y la creación de una interfaz gráfica de usuario o mejor conocida como GUI (Graphical User Interface).

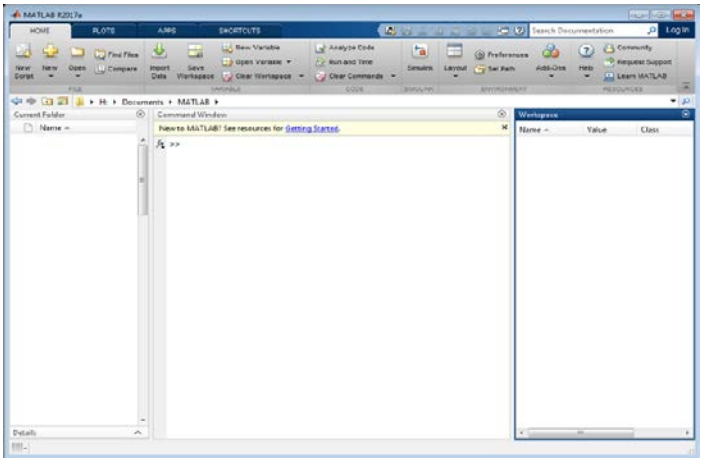


Figura 10. Escritorio de MATLAB.

A través de la herramienta GUIDE que es el entorno de desarrollo de GUI y los componentes que tiene, nos permitió diseñar gráficamente la interfaz de usuario, la cual generó un código de los elementos que incluye y el mismo que se modificó para programar los algoritmos de planeación de trayectorias del robot móvil.

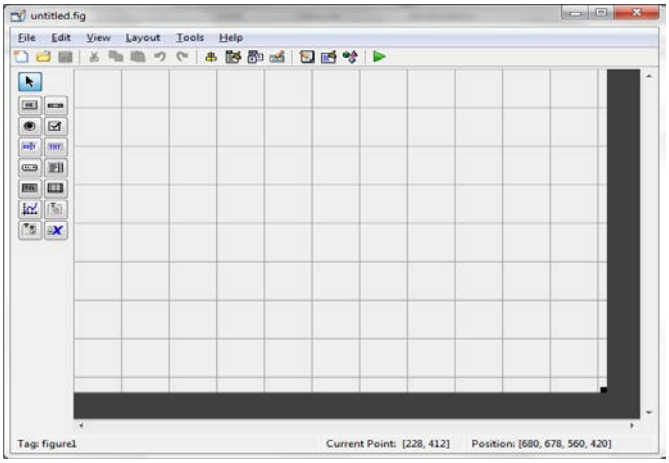


Figura 11. Entorno GUIDE.

La interfaz gráfica es de fácil entendimiento, de tal forma que no se cometa errores al momento de su uso. En el anexo 1 se proporciona un manual de usuario para el correcto uso del simulador de planeación de trayectorias.

2.5.1. VENTANA PRINCIPAL

La ventana principal es una interfaz de inicio que nos da a conocer los tipos de algoritmos con los que se puede trabajar las simulaciones.

Esta ventana tiene botones que al ser pulsados ejecutan la acción de abrir la ventana del simulador, de igual manera cuenta con un botón para cerrar todo el programa de la interfaz (figura 12).

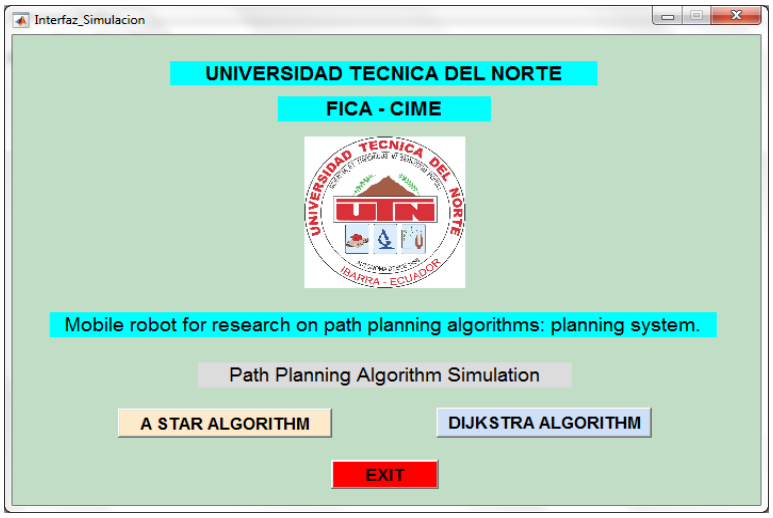


Figura 12. Ventana principal de la interfaz.

2.5.2. VENTANA DEL SIMULADOR

La ventana del simulador está formada por cuadros de texto editable que permiten el ingreso de valores numéricos, botones para ejecutar una acción y un cuadro gráfico para la visualización del mapa. Presenta sus respectivas partes que son para ejecutar las funciones del simulador:

- Ingresar el tamaño del mapa y la creación del mismo,

- Ingresar la posición de llegada e inicio del robot,
- Ingresar las posiciones de los obstáculos,
- Ejecutar el algoritmo para visualizar la simulación de planeación de trayectoria,
- Limpiar el cuadro grafico para poder realizar una nueva simulación y
- Salir de la ventana del simulador del algoritmo.

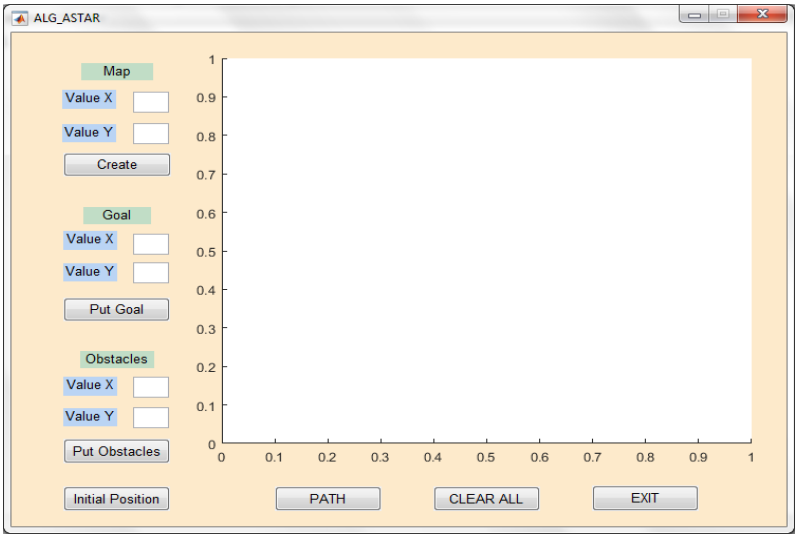


Figura 13. Ventana del simulador.

Como se puede ver en la figura 13, el diseño de la interfaz es el mismo para el simulador del algoritmo A* como de Dijkstra.

2.6. DIAGRAMA DE FUNCIONAMIENTO DEL SIMULADOR

2.6.1. SIMULADOR DEL ALGORITMO A*

En la figura 14, se presenta un diagrama el cual describe el funcionamiento del simulador del algoritmo A*. Se aprecia el procedimiento que inicia con el ingreso de datos para la creación del mapa, la colocación de las posiciones del robot y los obstáculos; el uso de la ecuación (1) que viene dado por $f(n) = g(n) + h(n)$ que ayuda en la búsqueda de la ruta óptima en este

algoritmo, el cual debe llegar a la meta para finalizar el proceso de búsqueda y recorrer la trayectoria.

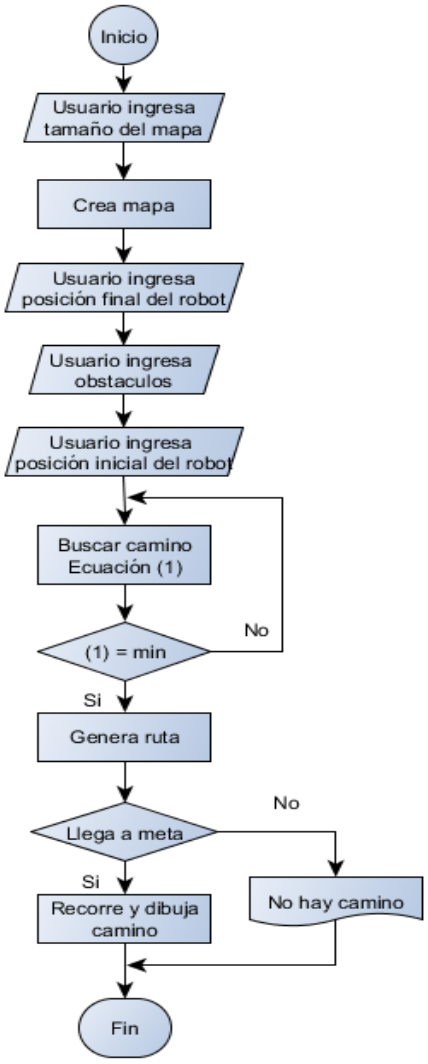


Figura 14. Diagrama de flujo del simulador del algoritmo A.*

2.6.2. SIMULADOR DEL ALGORITMO DE DIJKSTRA

En la figura 15, se presenta un diagrama el cual describe el funcionamiento del simulador del algoritmo de Dijkstra; se aprecia el procedimiento que inicia con el ingreso de datos para la creación del mapa, la colocación de las posiciones del robot y los obstáculos; inicia la búsqueda

del camino tomando el valor de costo de recorrido por las celdas, teniendo en cuenta el costo mínimo hasta llegar a la meta para finalizar el proceso de búsqueda y recorrer la trayectoria.

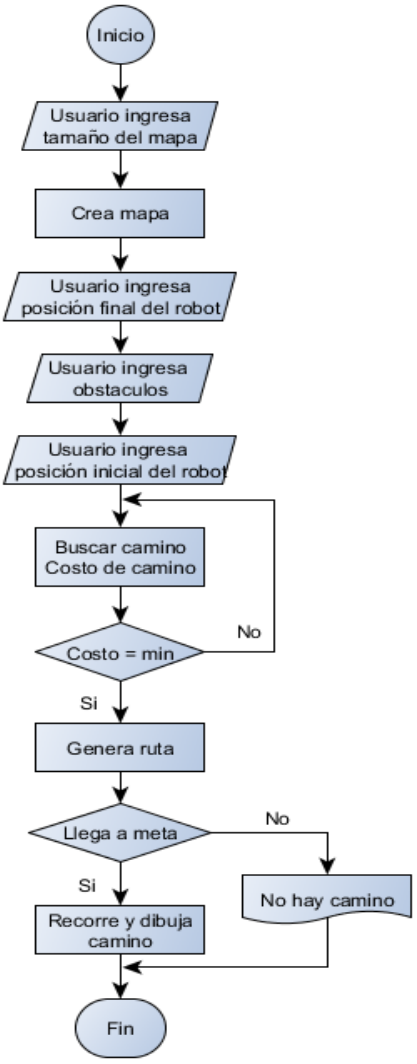


Figura 15. Diagrama de flujo del simulador del algoritmo de Dijkstra.

CAPÍTULO III

RESULTADOS

En este capítulo se describe una serie de procedimientos en base a los que se da la comprobación de los algoritmos de planificación de trayectorias en el simulador. Para comprobar el funcionamiento del algoritmo se realizará ejemplos, los mismos que serán pasados al programa de simulación y así poder verificar el correcto funcionamiento del mismo con la determinación de la ruta óptima así como una comparación del tiempo de procesamiento para cada algoritmo en mapas similares. Para los ejemplos se realizó cuadrículas, las cuales tienen marcadas con la letra S la posición de inicio y con la letra G la posición de llegada. Las posiciones de los obstáculos son cuadros pintados.

3.1. PRUEBAS DE FUNCIONAMIENTO DEL ALGORITMO A*

Las posiciones de análisis están en blanco y cada posición presenta las variables que deben ser consideradas, las mismas que son el costo de recorrido g(n) y el valor de la heurística h(n) para conocer cuál es el valor del costo mínimo hasta la posición de meta; es importante que se visualice estos valores en cada una de las posiciones que se analizan para así tener un mejor entendimiento de cómo funciona el algoritmo A* y observar que en función del valor del f(n) se llega a la posición deseada.

Los ejercicios se resolvieron partiendo de la posición de inicio S = (1,1), se realizó el cálculo del costo de recorrido y de la heurística en cada posición de análisis para obtener el valor de f(n).

3.1.1. EJERCICIO 1.1

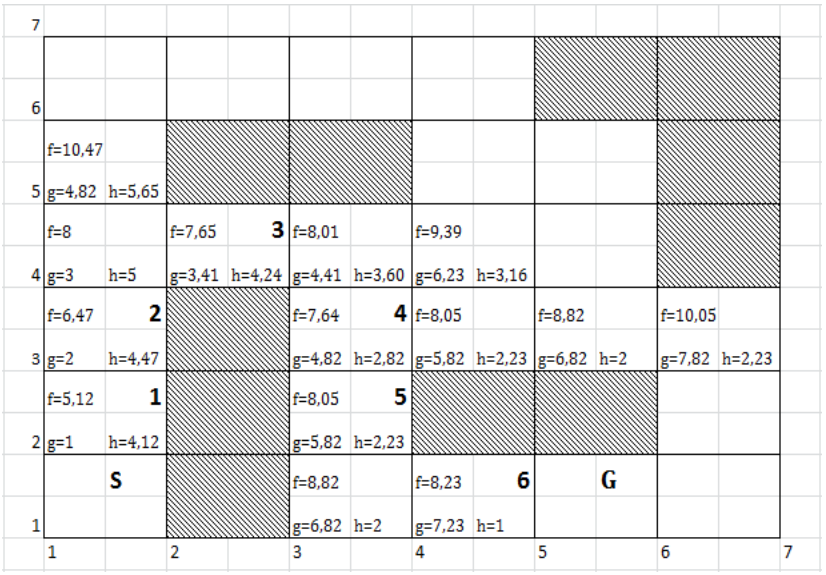


Figura 16. Ejercicio 1.1 del algoritmo A.*

Al resolver el ejercicio 1.1, se puede observar en la figura 16 que las posiciones de análisis con f(n) mínimo hasta llegar a la meta son (1,2), (1,3), (2,4), (3,3), (3,2) y (4,1). Las posiciones (1,5) y (6,3) son descartadas para la continuación del análisis ya que de continuar se presenta un valor de f(n) mayor con respecto a la ruta calculada como óptima.

En la tabla 2 se puede ver cuáles son los valores de costo de recorrido, la heurística y la sumatoria de estos para cada una de las coordenadas de ruta desde la salida hasta la meta.

Tabla 2

Valores del costo y coordenadas de ruta del ejercicio 1.1

	g(n)	**h(n)**	**f(n)**	**(X,Y)**
S	0	4	4	(1,1)
1	1	4.12	5.12	(1,2)
2	2	4.47	6.47	(1,3)

3	3.41	4.24	7.65	(2,4)
4	4.82	2.82	7.64	(3,3)
5	5.82	2.23	8.05	(3,2)
6	7.23	1	8.23	(4,1)
G	8.23	0	8.23	(5,1)

3.1.1.1. SIMULACIÓN EJERCICIO 1.1

En la figura 17, se puede observar que la ruta óptima que se determinó en el ejercicio 1.1 es la misma en el simulador con una distancia de recorrido de 8.23.

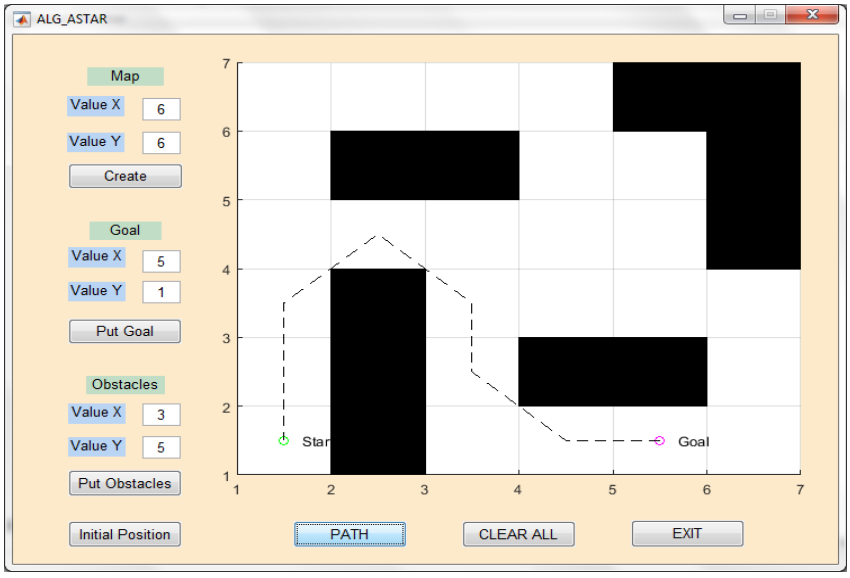


Figura 17. Ejercicio 1.1 comprobado en el simulador.

3.1.2. EJERCICIO 1.2

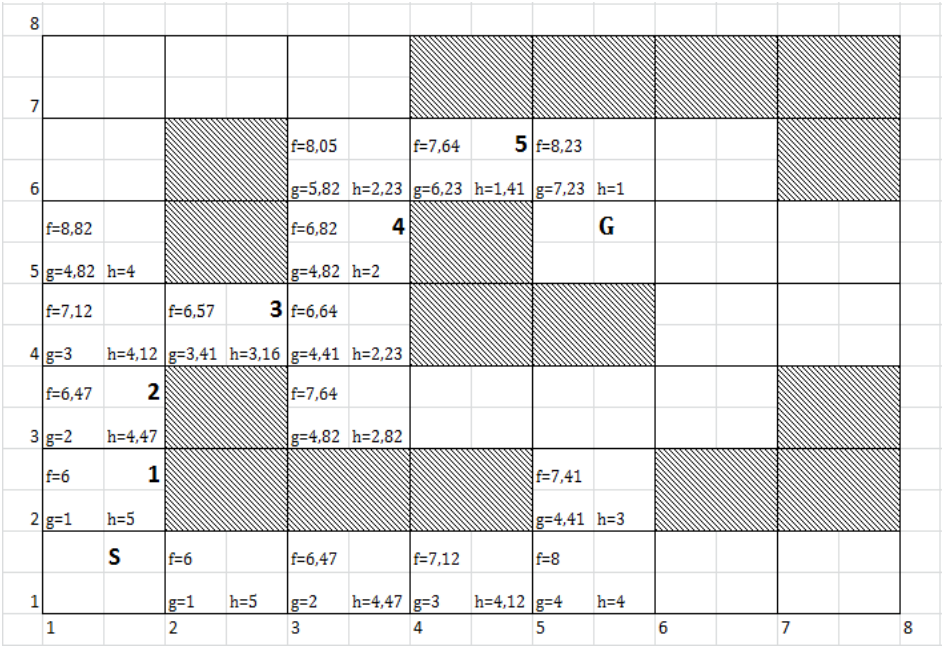


Figura 18. Ejercicio 1.2 del algoritmo A.*

Al resolver el ejercicio 1.2, se puede observar en la figura 18 que las posiciones de análisis con f(n) mínimo hasta llegar a la meta son (1,2), (1,3), (2,4), (3,5) y (4,6). Las posiciones (1,5), (5,2) y (3,3) son descartadas para la continuación del análisis ya que de continuar se presenta un valor de f(n) mayor con respecto a la ruta calculada como óptima.

En la tabla 3, se puede ver cuáles son los valores de costo de recorrido, la heurística y la sumatoria de estos para cada una de las coordenadas de ruta desde la salida hasta la meta.

Tabla 3

Valores del costo y coordenadas de ruta del ejercicio 1.2

	g(n)	**h(n)**	**f(n)**	**(X,Y)**
S	0	5.65	5.65	(1,1)
1	1	5	6	(1,2)

2	2	4.47	6.47	(1,3)
3	3.41	3.16	6.57	(2,4)
4	4.82	2	6.82	(3,5)
5	6.23	1.41	7.64	(4,6)
G	7.64	0	7.64	(5,5)

3.1.2.1. SIMULACIÓN EJERCICIO 1.2

En la figura 19, se puede observar que la ruta óptima que se determinó en el ejercicio 1.2 es la misma en el simulador con una distancia de recorrido de 7.64.

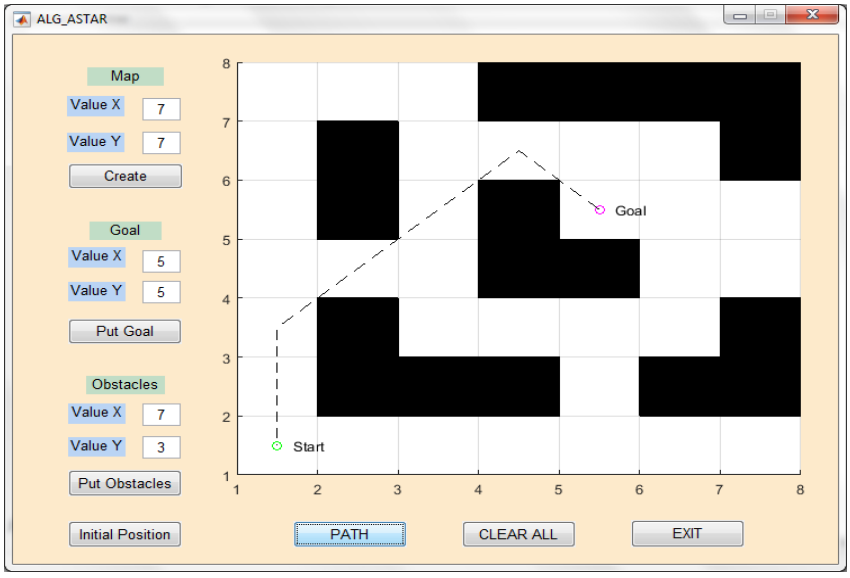


Figura 19. Ejercicio 1.2 comprobado en el simulador.

3.1.3. EJERCICIO 1.3

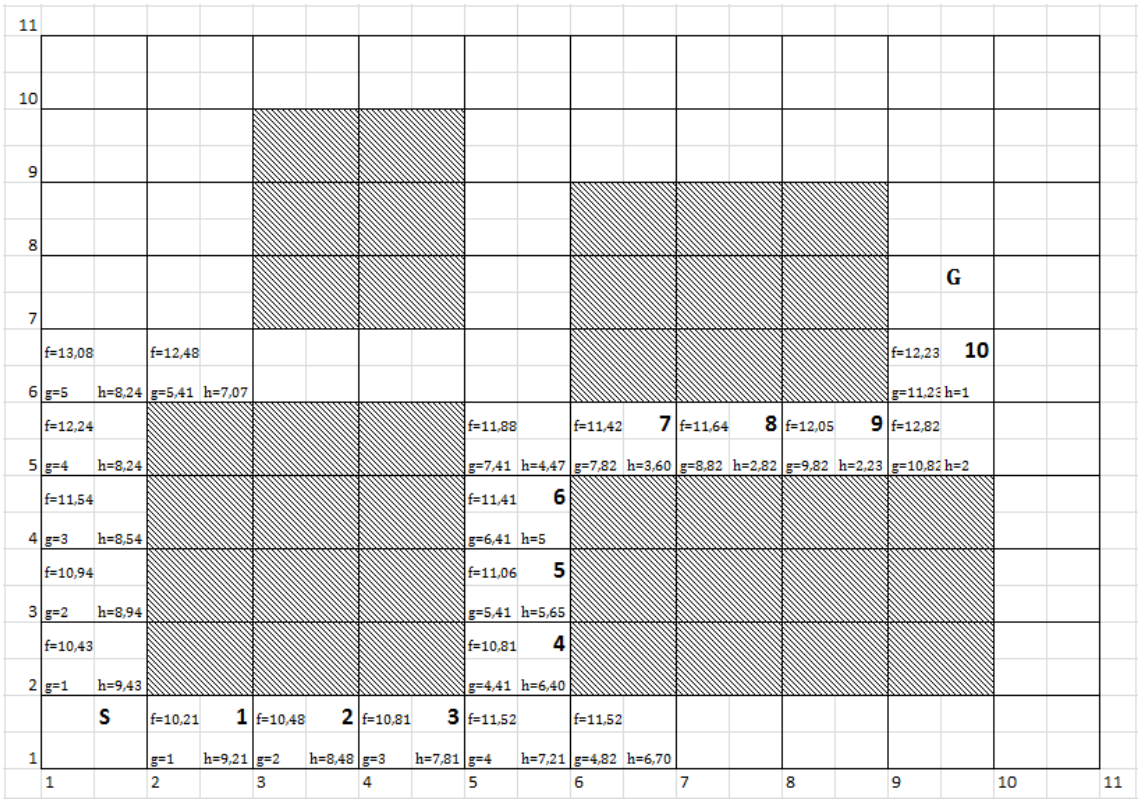


Figura 20. Ejercicio 1.3 del algoritmo A.*

Al resolver el ejercicio 1.3, se puede observar en la figura 20 que las posiciones de análisis con f(n) mínimo hasta llegar a la meta son (2,1), (3,1), (4,1), (5,2), (5,3), (5,4), (6,5), (7,5), (8,5) y (9,6). Las posiciones (6,1), (2,6) y (5,5) son descartadas para la continuación del análisis ya que de continuar se presenta un valor de f(n) mayor con respecto a la ruta calculada como óptima.

En la tabla 4 se puede ver cuáles son los valores de costo de recorrido, la heurística y la sumatoria de estos para cada una de las coordenadas de ruta desde la salida hasta la meta.

Tabla 4

Valores del costo y coordenadas de ruta del ejercicio 1.3

	g(n)	h(n)	f(n)	(X,Y)
S	0	10	10	(1,1)
1	1	9.21	10.21	(2,1)
2	2	8.48	10.48	(3,1)
3	3	7.81	10.81	(4,1)
4	4.41	6.40	10.81	(5,2)
5	5.41	5.65	11.06	(5,3)
6	6.41	5	11.41	(5,4)
7	7.82	3.60	11.42	(6,5)
8	8.82	2.82	11.64	(7,5)
9	9.82	2.23	12.05	(8,5)
10	11.23	1	12.23	(9,6)
G	12.23	0	12.23	(9,7)

3.1.3.1. SIMULACIÓN EJERCICIO 1.3

En la figura 21, se puede observar que la ruta óptima que se determinó en el ejercicio 1.3 es la misma en el simulador con una distancia de recorrido de 12.23.

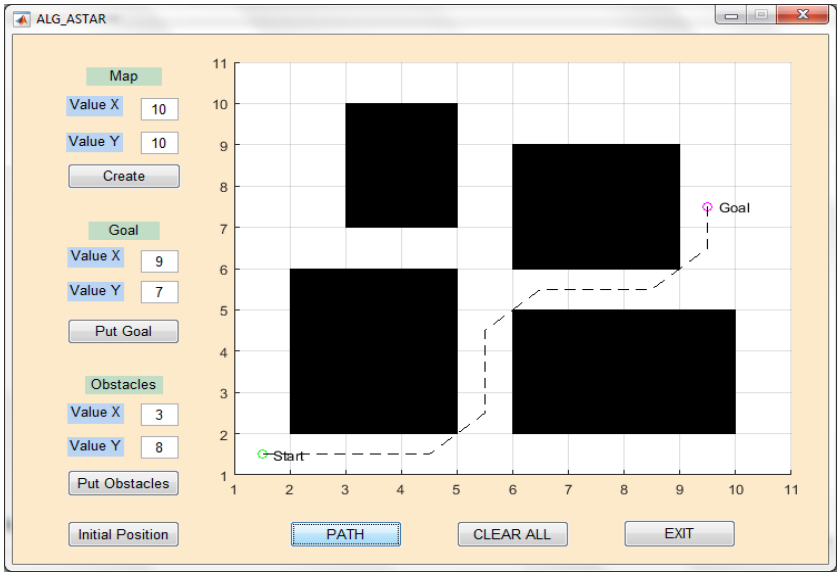


Figura 21. Ejercicio 1.3 comprobado en el simulador.

3.1.4. SIMULACIÓN 1.4

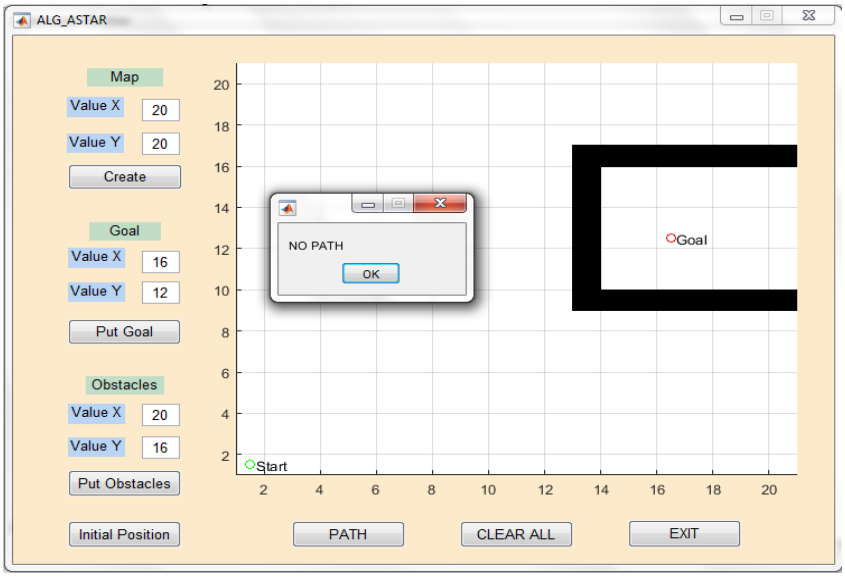


Figura 22. Simulación 1.4 del algoritmo A.*

La simulación 1.4 que se observa en la figura 22, viene dado para el caso en que no se encuentre una ruta.

3.2. PRUEBAS DE FUNCIONAMIENTO DEL ALGORITMO DE DIJKSTRA

Las posiciones de análisis están en blanco y cada posición presenta una letra del abecedario que vendría a ser con la que identificamos cada nodo o vértice; de esta forma se puede recorrer cada celda e identificar cual es la celda de la que proviene con el costo de recorrido que se tiene en cada una; por lo tanto la letra del abecedario que representa cada celda y el valor del costo ayudan a entender cómo funciona el algoritmo de Dijkstra hasta llegar a la posición deseada.

En la solución de los ejercicios se realiza el análisis desde la posición de salida hasta la meta pasando por los nodos vecinos identificando la relación que existe entre cada una de las posiciones del mapa, por lo que en las tablas de solución se muestra en cada paso el nodo marcado como ruta con su respectivo costo de recorrido, los nodos con los que es adyacente y con los que no tiene relación directa se le pone infinito (∞).

3.2.1. EJERCICIO 2.1

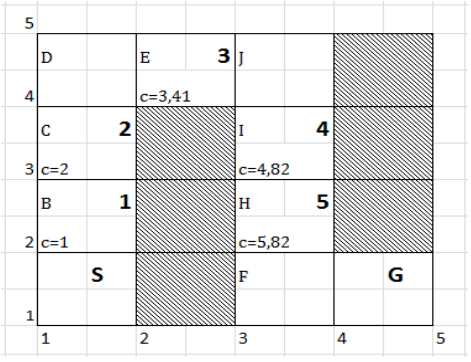


Figura 23. Ejercicio 2.1 del algoritmo de Dijkstra.

En la tabla 5, se realizó la solución del ejercicio 2.1; como se puede observar en la figura 23 existe sólo una ruta a la posición de llegada.

Tabla 5

Solución del ejercicio 2.1

	Paso 1	**Paso 2**	**Paso 3**	**Paso 4**	**Paso 5**	**Paso 6**	**Paso 7**
S	**(0,--)**	*	*	*	*	*	*
B	(1,S)	**(1,S)**	*	*	*	*	*
C	∞	(2,B)	**(2,B)**	*	*	*	*
D	∞	∞	(3,C)	(3,C)	(3,C)	(3,C)	(3,C)
E	∞	∞	(3.41,C)	**(3.41,C)**	*	*	*
F	∞	∞	∞	∞	∞	(6.82,H)	(6.82,H)
H	∞	∞	∞	∞	(5.82,I)	**(5.82,I)**	*
I	∞	∞	∞	(4.82,E)	**(4.82,E)**	*	*
J	∞	∞	∞	(4.41,E)	(4.41,E)	(4.41,E)	(4.41,E)
G	∞	∞	∞	∞	∞	(7.23,H)	**(7.23,H)**

En la tabla 6, se puede ver cuáles son los valores de costo del recorrido para cada una de las coordenadas de ruta, desde la salida hasta la meta.

Tabla 6

Valores del costo y coordenadas de ruta del ejercicio 2.1

	Costo de ruta	**(X,Y)**
S	0	(1,1)
1	1	(1,2)
2	2	(1,3)
3	3.41	(2,4)
4	4.82	(3,3)

5	5.82	(3,2)
G	7.23	(4,1)

3.2.1.1. SIMULACIÓN EJERCICIO 2.1

En la figura 24, se puede observar que la ruta óptima que se determinó en el ejercicio 2.1 es la misma en el simulador con una distancia de recorrido de 7.23.

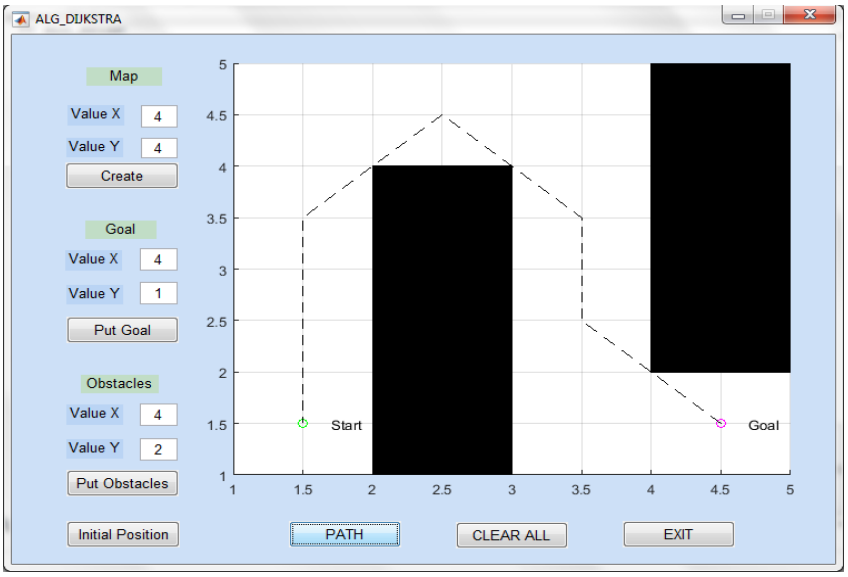


Figura 24. Ejercicio 2.1 comprobado en el simulador.

3.2.2. EJERCICIO 2.2

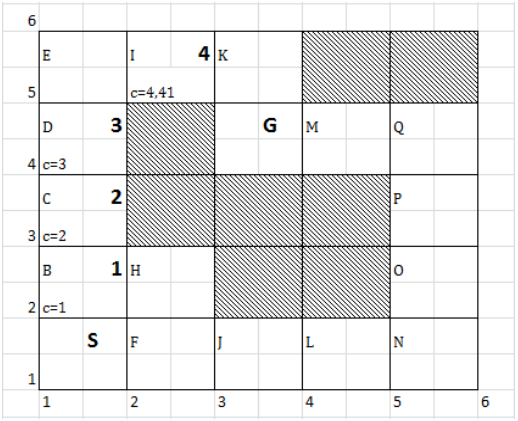


Figura 25. Ejercicio 2.2 del algoritmo de Dijkstra.

En la tabla 7, se realizó la solución del ejercicio 2.2; como se puede observar en la figura 25 existen dos rutas a la posición de llegada.

Tabla 7

Solución del ejercicio 2.2

	Paso 1	**Paso 2**	**Paso 3**	**Paso 4**	**Paso 5**	**Paso 6**
S	**(0,--)**	*	*	*	*	*
B	(1,S)	**(1,S)**	*	*	*	*
C	∞	(2,B)	**(2,B)**	*	*	*
D	∞	∞	(3,C)	**(3,C)**	*	*
E	∞	∞	∞	(4,D)	(4,D)	(4,D)
F	(1,S)	(1,S)	*	*	*	*
H	(1.41,S)	(1.41,S)	(1.41,S)	(1.41,S)	(1.41,S)	(1.41,S)
I	∞	∞	∞	(4.41,D)	**(4.41,D)**	*
J	∞	(2,F)	(2,F)	*	*	*
K	∞	∞	∞	∞	(5.41,I)	(5.41,I)

L	∞	∞	(3,J)	(3,J)	*	*****
M	∞	∞	∞	∞	∞	(6.82,P)
N	∞	∞	∞	(4,L)	(4,L)	(4,L)
O	∞	∞	∞	(4.41,L)	(4.41,L)	*
P	∞	∞	∞	∞	(5.41,O)	(5.41,O)
Q	∞	∞	∞	∞	∞	(6.41,P)
G	∞	∞	∞	∞	(5.82,I)	**(5.82,I)**

En la tabla 8, se puede ver cuáles son los valores de costo del recorrido para cada una de las coordenadas de ruta, desde la salida hasta la meta.

Tabla 8

Valores del costo y coordenadas de ruta del ejercicio 2.2

	Costo de ruta	**(X,Y)**
S	0	(1,1)
1	1	(1,2)
2	2	(1,3)
3	3	(1,4)
4	4.41	(2,5)
G	5.82	(3,4)

3.2.2.1. SIMULACIÓN EJERCICIO 2.2

En la figura 26, se puede observar que la ruta óptima que se determinó en el ejercicio 2.2 es la misma en el simulador con una distancia de recorrido de 5.82.

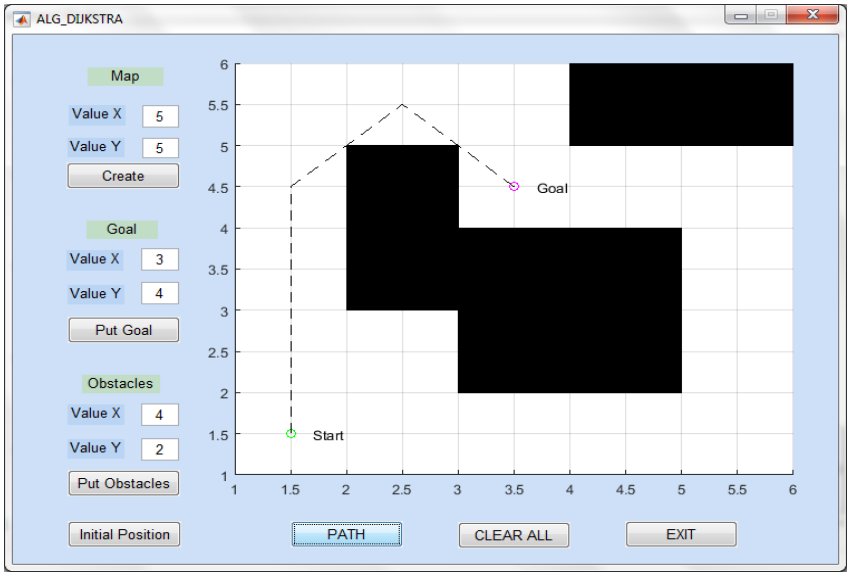


Figura 26. Ejercicio 2.2 comprobado en el simulador.

3.2.3. EJERCICIO 2.3

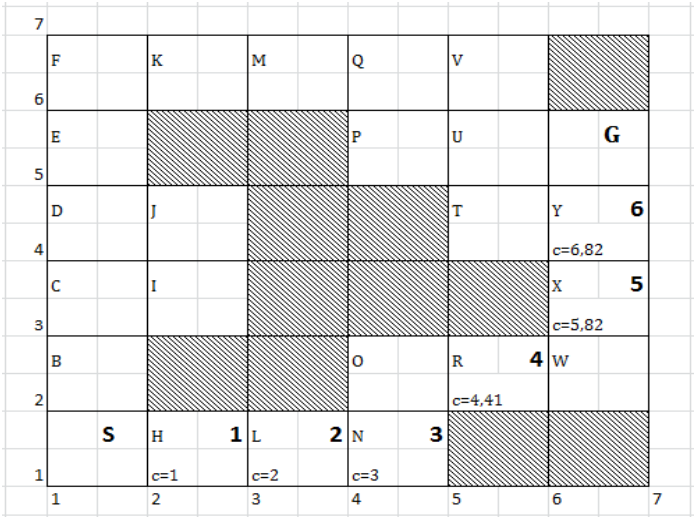


Figura 27. Ejercicio 2.3 del algoritmo de Dijkstra.

En la tabla 9 se realizó la solución del ejercicio 2.3; como se puede observar en la figura 27 existen dos rutas a la posición de llegada.

Tabla 9

Solución del ejercicio 2.3

	Paso 1	**Paso 2**	**Paso 3**	**Paso 4**	**Paso 5**	**Paso 6**	**Paso 7**	**Paso 8**
S	**(0,--)**	*	*	*	*	*	*	*
B	(1,S)	(1,S)	*	*	*	*	*	*
C	∞	(2,B)	(2,B)	*	*	*	*	*
D	∞	∞	(3,C)	(3,C)	*	*	*	*
E	∞	∞	∞	(4,D)	(4,D)	*	*	*
F	∞	∞	∞	∞	(5,E)	(5,E)	(5,E)	(5,E)
H	(1,S)	**(1,S)**	*	*	*	*	*	*
I	∞	(2.41,B)	(2.41,B)	(2.41,B)	(2.41,B)	(2.41,B)	(2.41,B)	(2.41,B)
J	∞	∞	(3.41,C)	(3.41,C)	(3.41,C)	(3.41,C)	(3.41,C)	(3.41,C)
K	∞	∞	∞	∞	(5.41,E)	(5.41,E)	*	*
L	∞	(2,H)	**(2,H)**	*	*	*	*	*
M	∞	∞	∞	∞	∞	(6.41,K)	(6.41,K)	*
N	∞	∞	(3,L)	**(3,L)**	*	*	*	*
O	∞	∞	(3.41,L)	(3.41,L)	(3.41,L)	(3.41,L)	(3.41,L)	(3.41,L)
P	∞	∞	∞	∞	∞	∞	(7.82,M)	(7.82,M)
Q	∞	∞	∞	∞	∞	∞	(7.41,M)	(7.41,M)
R	∞	∞	∞	(4.41,N)	**(4.41,N)**	*	*	*
T	∞	∞	∞	∞	∞	∞	∞	∞
U	∞	∞	∞	∞	∞	∞	∞	(8.82,Q)
V	∞	∞	∞	∞	∞	∞	∞	(8.41,Q)
W	∞	∞	∞	∞	(5.41,N)	(5.41,N)	(5.41,N)	(5.41,N)
X	∞	∞	∞	∞	(5.82,N)	**(5.82,N)**	*	*
Y	∞	∞	∞	∞	∞	(6.82,X)	**(6.82,X)**	*

G	∞	∞	∞	∞	∞	∞	(7.82,Y)	**(7.82,Y)**

En la tabla 10, se puede ver cuáles son los valores de costo del recorrido para cada una de las coordenadas de ruta, desde la salida hasta la meta.

Tabla 10

Valores del costo y coordenadas de ruta del ejercicio 2.3

	Costo de ruta	**(X,Y)**
S	0	(1,1)
1	1	(2,1)
2	2	(3,1)
3	3	(4,1)
4	4.41	(5,2)
5	5.82	(6,3)
6	6.82	(6,4)
G	7.82	(6,5)

3.2.3.1. SIMULACIÓN EJERCICIO 2.3

En la figura 28, se puede observar que la ruta óptima que se determinó en el ejercicio 2.3 es la misma en el simulador con una distancia de recorrido de 7.82

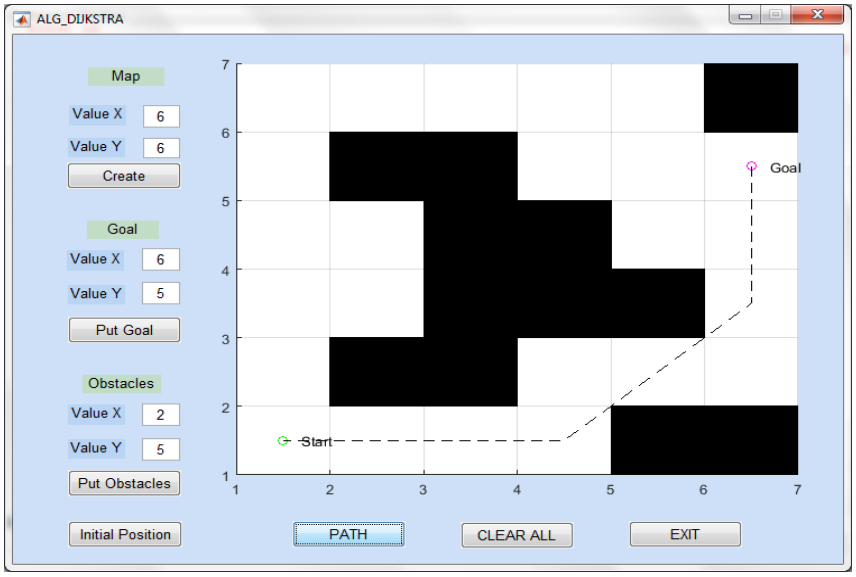


Figura 28. Ejercicio 2.3 comprobado en el simulador.

3.2.4. SIMULACIÓN 2.4

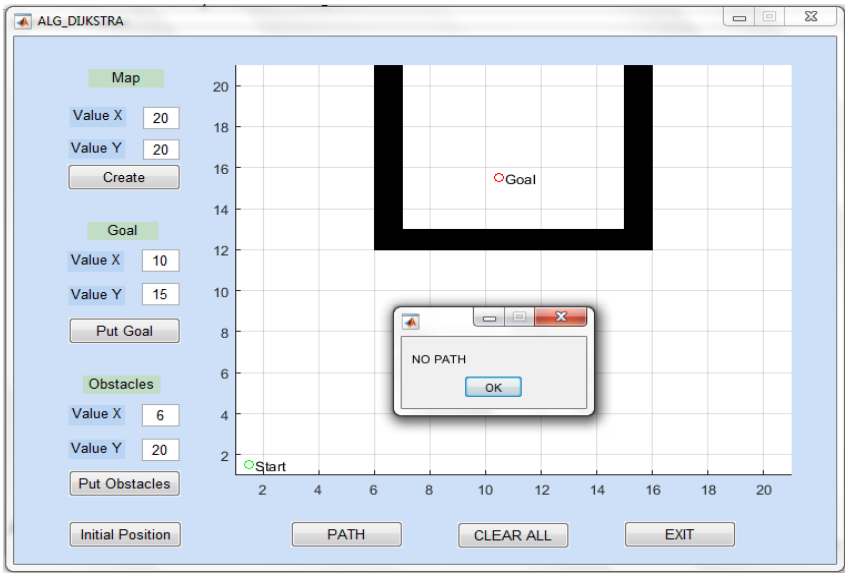


Figura 29. Simulación 2.4 del algoritmo de Dijkstra.

La simulación 2.4 que se observa en la figura 29, viene dado para el caso en que no se encuentre una ruta.

3.3. REPETITIVAD

Al realizar la simulación tanto con el algoritmo A* y de Dijkstra en un mismo tipo de mapa por más de una ocasión, se puede observar que el simulador presenta repetitividad en las rutas calculadas y graficadas.

3.4. TIEMPO DE PROCESAMIENTO DE SIMULACIÓN

El tiempo de procesamiento se obtiene como resultado del programa MATLAB. Se realiza simulaciones para cada uno de los algoritmos con el propósito de saber cuál presenta mayor eficiencia en velocidad de análisis y ejecución de la simulación. Las simulaciones son realizadas en cuatro mapas, los cuales tienen un variado número de obstáculos.

En las figuras 24 y 25 se muestran los mapas de tamaño 10x10 y en las figuras 26 y 27 se muestran los mapas de tamaño 20x20.

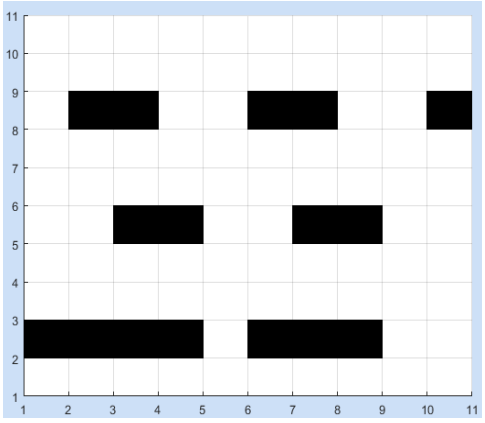


Figura 30. Mapa 1.

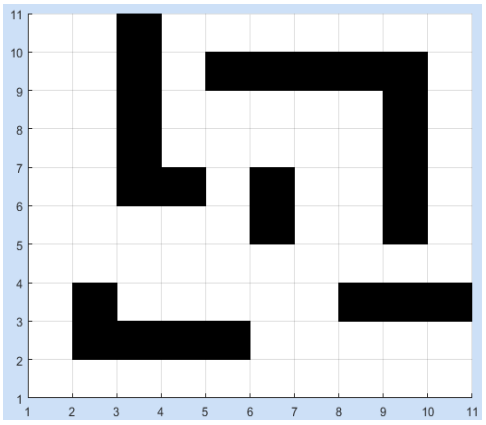


Figura 31. Mapa 2.

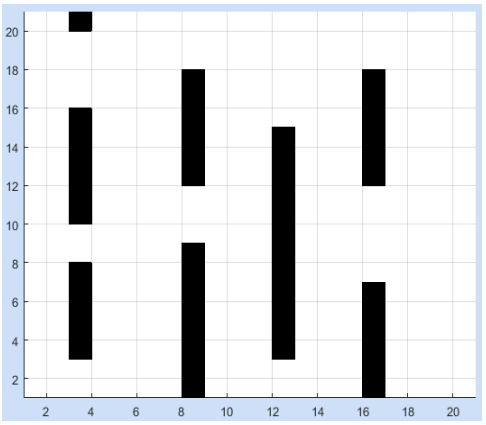


Figura 32. Mapa 3.

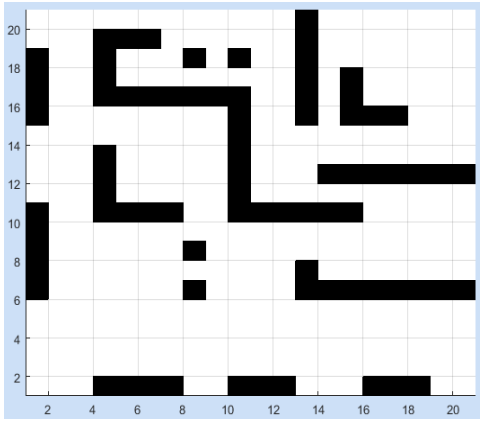


Figura 33. Mapa 4.

Los mapas simulados para el algoritmo A* nos dieron los siguientes resultados:

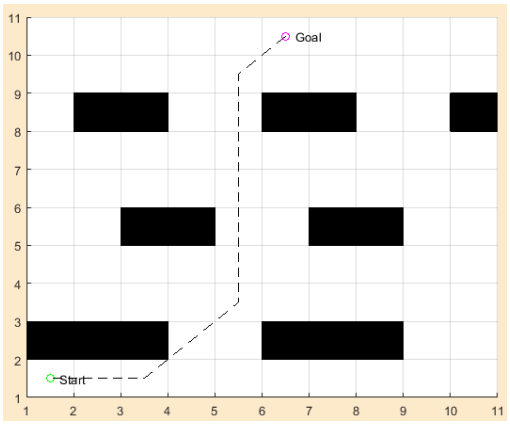


Figura 34 . Simulación mapa 1, algoritmo A.*

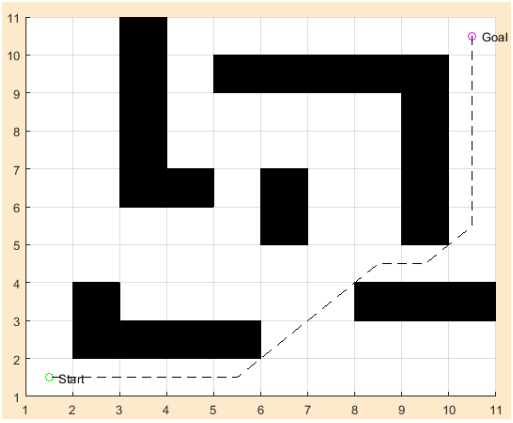


Figura 35. Simulación mapa 2, algoritmo A.*

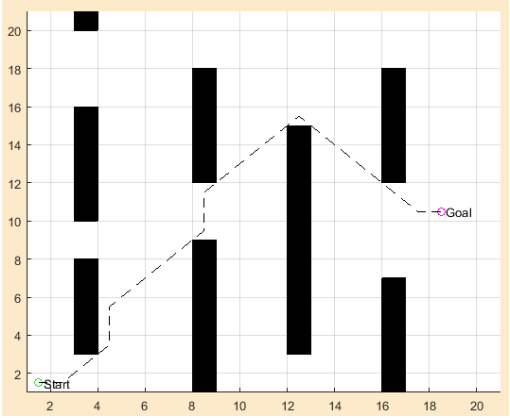


Figura 36. Simulación mapa 3, algoritmo A.*

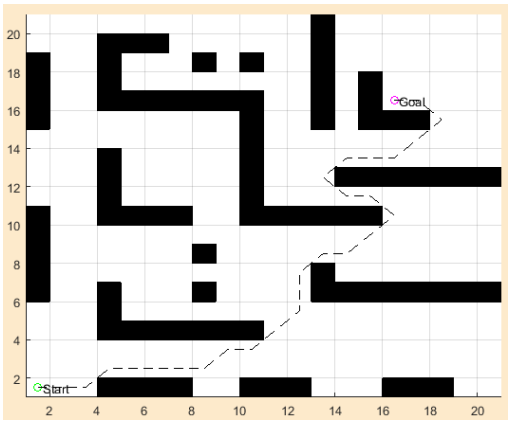


Figura 37. Simulación mapa 4, algoritmo A.*

Los mapas simulados para el algoritmo de Dijkstra nos dieron los siguientes resultados:

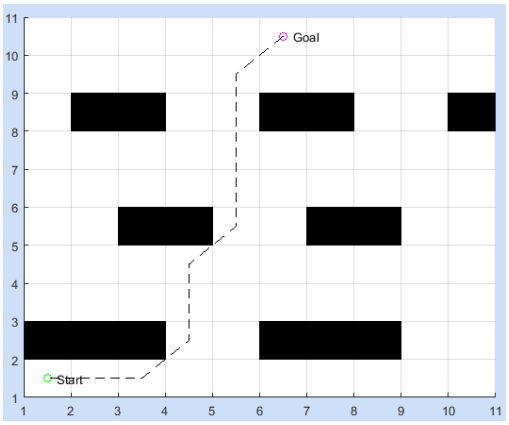


Figura 38. Simulación mapa 1, algoritmo de Dijkstra.

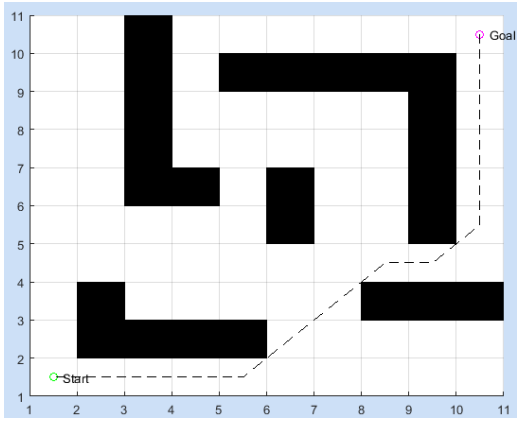


Figura 39. Simulación mapa 2, algoritmo de Dijkstra.

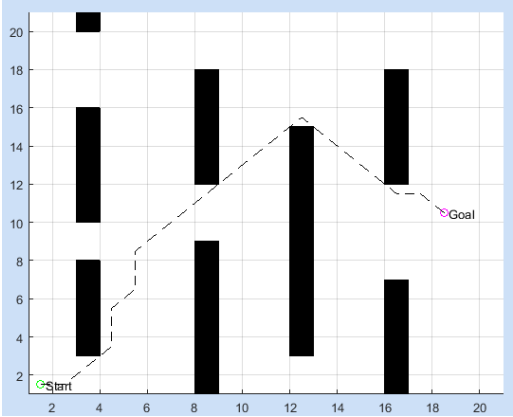


Figura 40. Simulación mapa 3, algoritmo de Dijkstra.

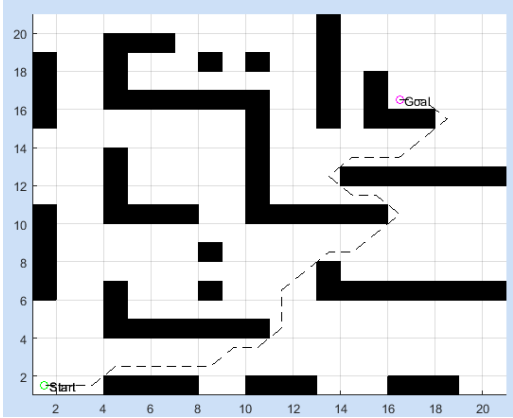


Figura 41. Simulación mapa 4, algoritmo de Dijkstra.

Las simulaciones se realizaron en una computadora Intel Core i5-2450M CPU 2.50GHz, 8,0GB RAM; en la tabla 11 se observa las coordenadas de posición de inicio y llegada del robot, así como el tiempo que se demoró en ejecutar el análisis y simulación cada algoritmo.

Tabla 11

Posición del robot y tiempo de procesamiento.

Mapa	Posición de inicio	Posición de meta	A* (s)	Dijsktra (s)
1	Inicio=(1,1)	Meta=(6,10)	3.3934	3.6447
2	Inicio=(1,1)	Meta=(10,10)	4.0627	4.3390

3	Inicio=(1,1)	Meta=(18,10)	6.5727	6.6351
4	Inicio=(1,1)	Meta=(16,16)	8.6146	8.7523

Como se puede observar el tiempo de procesamiento de la simulación varía según el tamaño del mapa y de la cantidad de obstáculos que este presenta. Hay que tomar en cuenta que los tiempos de cálculo de respuesta obtenidos pueden variar, dependiendo de la velocidad en la que se esté ejecutando MATLAB en la computadora.

Tabla 12

Diferencia de velocidad en segundos y porcentual

Mapa	Diferencia de velocidad (s)	Diferencia de velocidad (%)
1	0.2513	6.89%
2	0.2763	6.36%
3	0.0624	0.94%
4	0.1377	1.57%

Al analizar los valores de tiempo obtenidos para el algoritmo A* y de Dijkstra, se puede ver que es más rápido el algoritmo A* en un promedio estimado entre los cuatro mapas del 3.94%, ya que se guía con el valor de la heurística y analiza los nodos que presentan una función de costo menor.

Hay que tomar en cuenta el resultado de la simulación del mapa 2 en las figuras 35 y 39, que presentan la misma trayectoria para los dos algoritmos.

CAPÍTULO IV

CONCLUSIONES Y TRABAJO FUTURO

En este capítulo se dan a conocer las conclusiones referentes a la realización del proyecto de titulación, que se dieron como parte de la investigación, desarrollo y resultados; también se dará recomendaciones que ayudarán al uso del simulador y al desarrollo a futuro del proyecto.

4.1. CONCLUSIONES

- La planeación de trayectorias representa un estudio muy importante para la navegación autónoma de robots móviles por la necesidad de obtener una ruta que se encuentre libre de obstáculos y que sea óptima para el robot.
- Con el análisis de la información obtenida en nuestra bibliografía se determinó que para el desarrollo del proyecto se necesita algoritmos de búsqueda de gráficos y árboles.
- Uno de los algoritmos de búsqueda más utilizados en investigaciones de aplicación de planificación de caminos es el algoritmo A*.
- Tanto el algoritmo A* como el de Dijkstra son considerados completos por su eficiencia en encontrar el camino de menor costo; lo que diferencia al algoritmo A* del de Dijkstra es la función heurística que usa como referencia de distancia para buscar la ruta al puto de llegada del robot móvil.
- Mediante el uso de MATLAB se pudo desarrollar la interfaz de simulación por la herramienta GUIDE que presenta y por su facilidad de trabajar con matrices, que fueron necesarias para la programación de los algoritmos de planificación.

- Después de investigar cómo se desarrolla el funcionamiento de cada algoritmo aplicado a un mapa de rejilla y de su comprobación con el software de simulación, se puede observar que el simulador de los algoritmos funcionan de manera correcta dando como resultado una trayectoria óptima; presentando el 100% de fiabilidad en las pruebas de repetitividad.
- Dependiendo del entorno o mapa, el algoritmo A* realiza el análisis de búsqueda mucho más rápido que el algoritmo de Dijkstra, superándolo en un promedio estimado del 3.94%; estos algoritmos incluso pueden llegar a determinar la misma ruta si fuera el caso.

4.2. TRABAJO FUTURO

- Desarrollar mejoras en la interfaz de simulación y en la programación de los algoritmos para aumentar la eficiencia de los mismos y facilitar su uso.
- Realizar un análisis como lo propone Goyal y Nagla [15], en el que consideran una dimensión para el robot.
- Realizar investigación en otros tipos de algoritmos para implementarlos en la interfaz del simulador.
- Continuar desarrollando el proyecto, para que los programas de los algoritmos de planificación puedan ser aplicados en la plataforma robótica móvil que se deja construida y así comprobar el funcionamiento en la misma.

BIBLIOGRAFÍA

[1] L. A. Arellano Zea, *Diseño e implementación de un robot móvil con control de trayectoria mediante principios odométricos,* Lima, 2015.

[2] F. Benavides, *Planificación de movimientos aplicada en robótica autónoma móvil,* Montevideo, 2012.

[3] V. Zambrano, *Implementacíon de Algoritmos de Determinacion de Rutas para el Robotino® de Festo,* Quito, 2015.

[4] A. Yandún, *Planeación y Seguimiento de Trayectorias para un Robot Móvil,* Quito, 2011.

[5] M. Fernández, D. Fernández y C. Valmaseda, *Planificacion de trayectorias para un Robot Móvil,* Madrid, 2009.

[6] P. Corke, Robotics, Vision and Control, Springer, 2011.

[7] L. Álvarez y J. Figueroa, *Implementación de algoritmos de navegación utilizando la plataforma iRobot Create y módulos de comunicación inalambrica Xbee,* 2011.

[8] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico y L. Jurišica, «Path Planning with Modified A Star Algorithm for a Mobile Robot,» *ScienceDirect,* vol. 96, pp. 59-69, 2014.

[9] N. Correll, Introduction to Autonomous Robots, 1 ed., 2016.

[10] H. E. Espitia Cuchango y J. I. Sofrony Esmeral, «Algoritmo para Planear Trayectorias de Robots Móviles, Empleando Campos Potenciales y Enjambres de Partículas Activas Brownianas,» *Ciencia e Ingeniería Neogranadina,* vol. 22, nº 2, pp. 75-96, Diciembre 2012.

[11] R. Martínez Ángel, J. Barrero Pérez y D. A. Tibaduiza Burgos, «Algoritmos de Planificación de Trayectorias para un Robot Móvil,» *ResearchGate,* Agosto 2006.

[12] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki y S. Thrun, Principles of Robot Motion: Theory, Algorithms, and Implementation, Cambridge: MIT Press, 2007.

[13] S. M. LaValle, Planning Algorithms, Cambridge, 2006.

[14] P. E. Hart, N. J. Nilsson y B. Raphael, «A Formal Basis for the Heuristic Determination of Minimum Cost Paths.,» *IEEE Transactions on Systems Science and Cybernetics,* vol. 4, pp. 100-107, 1968.

[15] J. K. Goyal y K. Nagla, «A New Approach of Path Planning for Mobile Robots,» *International Conference on Advances in Computing, Communications and Informatics (ICACCI),* 2014.

[16] E. W. Dijkstra, «A Note on Two Problems in Connexion with Graphs,» *Numerische Mathemathik,* vol. 1, pp. 269-271, 1959.

[17] Z. Zhang y Z. Zhao, «A Multiple Mobile Robots Path planning Algorithm Based on A-star and Dijkstra Algorithm,» *International Journal of Smart Home,* vol. 8, nº 3, pp. 75-86, 2014.

[18] N. O. Eraghi, F. López-Colino, A. de Castro y J. Garrido, «Path Length Comparison in Grid Maps of Planning Algorithms: HCTNav, A* and Dijkstra,» *Design of Circuits and Integrated Systems,* pp. 1-6, 2014.

[19] A. Ollero Baturone, Robótica Manipuladores y robots móviles, España: Marcombo, 2001.

[20] R. Siegwart y I. R. Nourbakhsh, Introduction to Autonomous Mobile Robots, London, 2004.

[21] T. Bräunl, Embedded Robotics Mobile Robot Design and Aplications with Embedded Systems, Perth: Springer, 2006.

[22] D. O. Barragán Guerrero, «Repositorio de ESPOL,» 25 Mayo 2008. [En línea]. Available: https://www.dspace.espol.edu.ec/bitstream/123456789/10740/11/MATLAB_GUIDE.pdf.

[23] MathWorks, Inc, «Matlab Primer,» Marzo 2017. [En línea]. Available: http://www.mathworks.com/help/index.html.

[24] A. M. Cueva, *Generación global de trayectorias para robots móviles, basada en curvas betaspline,* Sevilla, 2014.

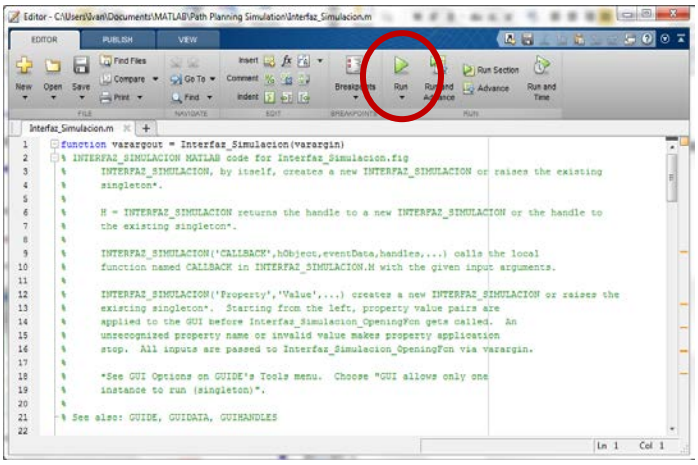
[25] W. Shu-xi y Z. Xing-qui, «The Improved Dijkstra's Shortest Path Algorithm,» *International Conference on Natural Computation,* pp. 2313-2316, 2011.

ANEXOS

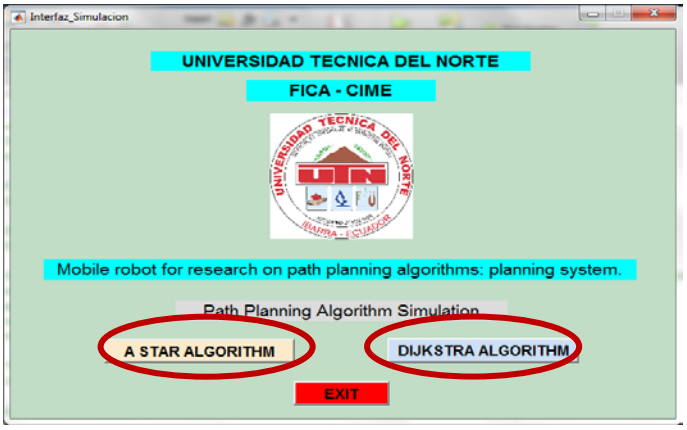
ANEXO 1

Manual de usuario del simulador de planeación de trayectorias.

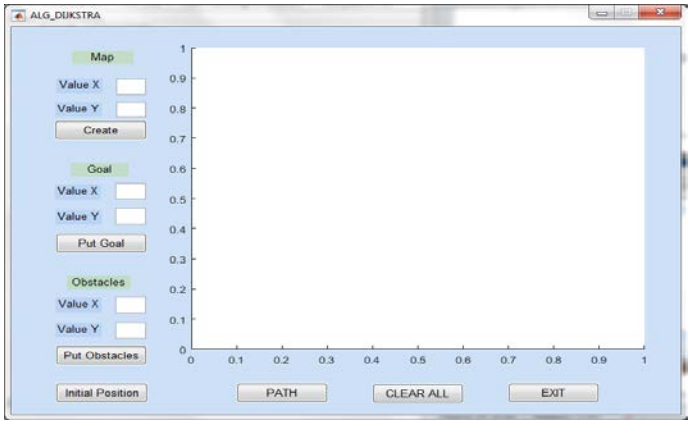
1. Abrir el archivo **“Interfaz_Simulacion.m”**.
2. Ejecutar el programa del simulador aplastando Run.



3. Seleccionar el tipo de algoritmo que se desee simular.

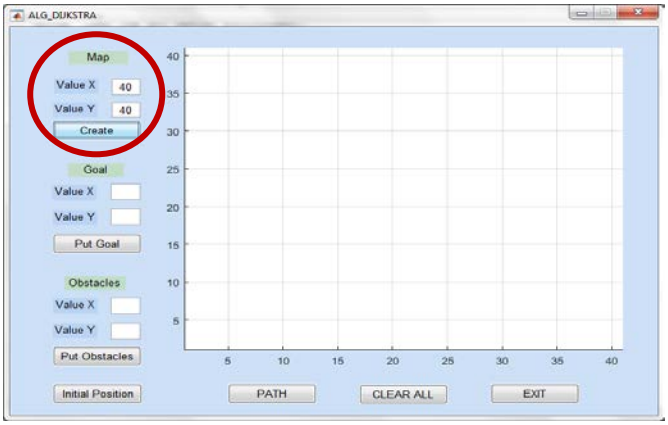


4. Se abre la ventana del algoritmo que se seleccionó.

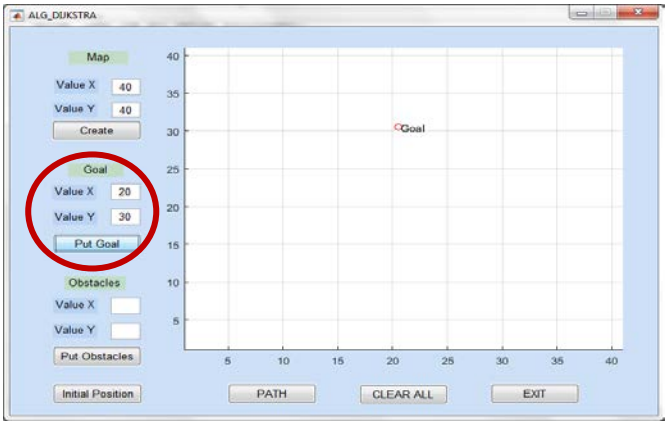


Nota: Para cualquiera de los dos algoritmos el procedimiento de funcionamiento es similar.

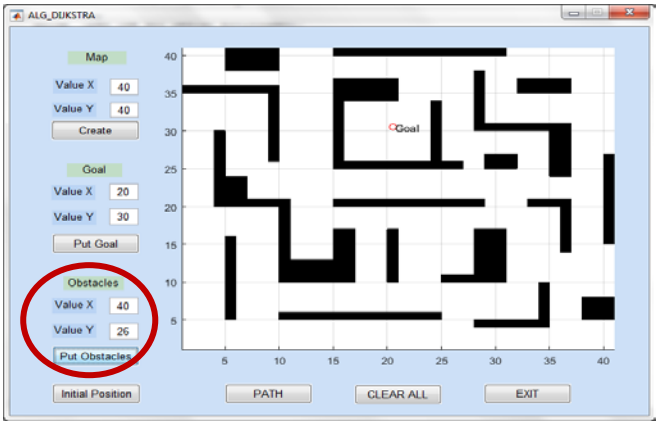
5. Ingresar el tamaño del mapa y crearlo.



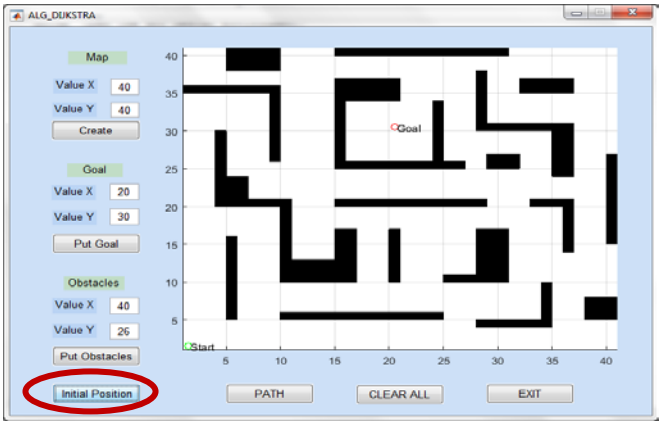
6. Ingresar la posición de llegada del robot.



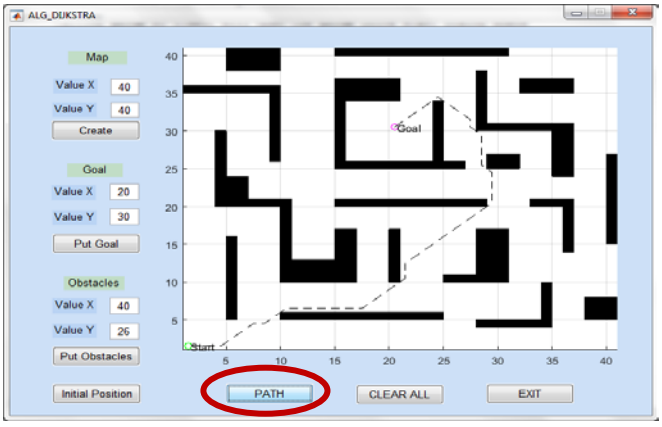
7. Ingresar los obstáculos.



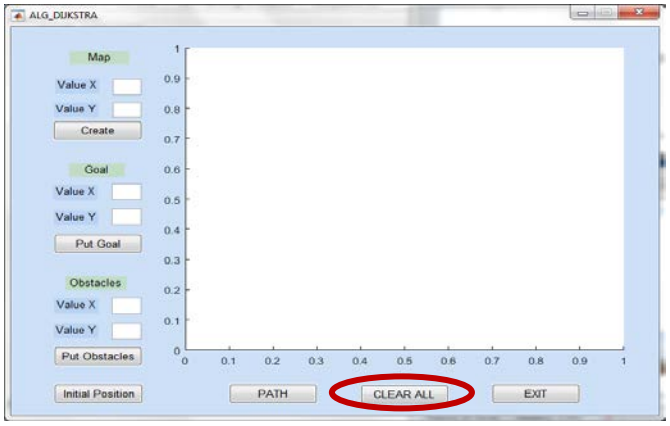
8. Ingresar la posición de inicio del robot que será en la coordenada (1,1).



9. Aplastar PATH para que se grafique la ruta calculada por el algoritmo.



10. Si se desea realizar otra simulación aplastar CLEAR ALL para limpiar el mapa.



11. Para salir de la ventana del simulador del algoritmo aplastar EXIT.

12. Para cerrar todo el programa del simulador aplastar EXIT en la ventana principal.