ALGORITMOS DE PLANIFICACIÓN DE TRAYECTORIAS BASADOS EN FAST MARCHING SQUARE

AUTOR: JOSE PARDEIRO

TUTOR: RAMÓN BARBER

DIRECTOR: JAVIER V. GÓMEZ

ÍNDICE

1.- Introducción

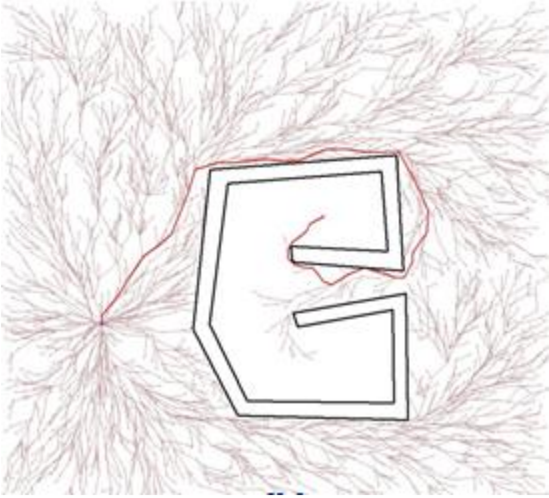
- ¿Qué es planificación de trayectorias?
  - Secuencia de acciones para llevar un sistema de un estado a otro dado un entorno.
  - En robótica se puede aplicar a robots móviles, vehículos aéreos, manipuladores...

- ¿Qué características debe reunir un planificador de trayectorias?
  - Suavidad.
  - Eficiencia computacional.
  - Seguridad.
  - Rapidez.
  - Óptimo.

1.- Introducción

Contexto y motivaciones

- Existen multitud de métodos que resultan adecuados para tareas concretas.
- Fallan en problemas muy concretos.
- En tareas complejas conlleva variar el método de planificación según el contexto.



1.- Introducción

Desarrollo de algoritmos de planificación de trayectorias para propósitos generales y adaptables a entornos reales.

Objetivos

- Desarrollar variaciones de Fast Marching Square (FM^2) que mejoren sus características.
- Integrar los algoritmos en ROS de forma eficiente.
- Comprobar su rendimiento tanto en entorno de simulación como en entorno real.

ÍNDICE

2.- FAST MARCHING SQUARE
Fast Marching Method (FMM)

- Método propuesto por J. A. Sethian en 1996.
- Basado en la expansión de una onda en un fluido.
- La onda se propaga en todas las direcciones con velocidad no negativa.
- FMM calcula el tiempo de llegada de la onda a cada punto del espacio.

2.- FAST MARCHING SQUARE
Fast Marching Method (FMM)

- Matemáticamente el tiempo de llegada se calcula tal que:

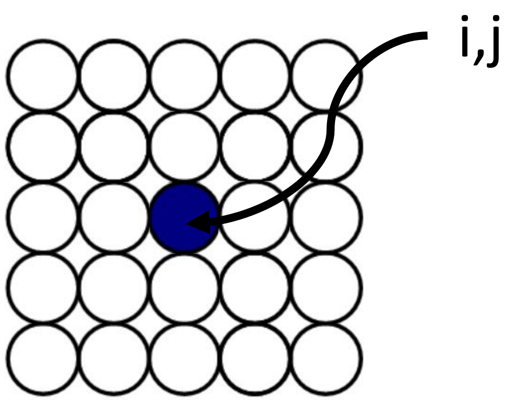
$$\left(\frac{T_{i,j} - T_1}{\Delta x}\right)^2 + \left(\frac{T_{i,j} - T_2}{\Delta y}\right)^2 = \frac{1}{F_{i,j}^2}$$

$$T_1 = \min(T_{i-1,j}, T_{i+1,j})$$

$$T_2 = \min(T_{i,j-1}, T_{i,j+1})$$

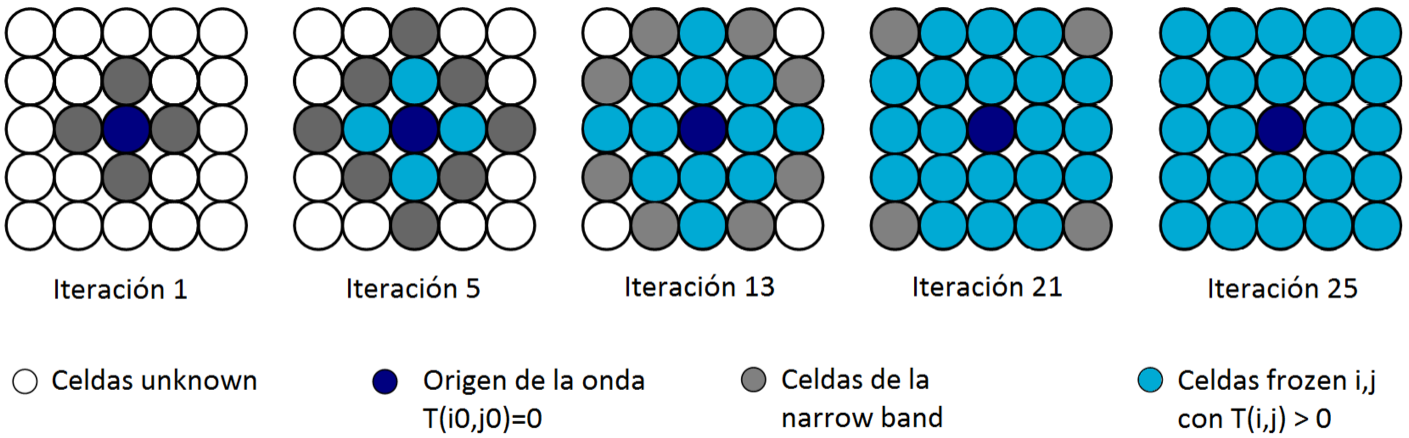
donde:

- Δx, Δy: Distancia entre vecinos.
- $F_{i,j}$: Velocidad del punto a evaluar.
- $T_{i,j}$: Punto a evaluar.

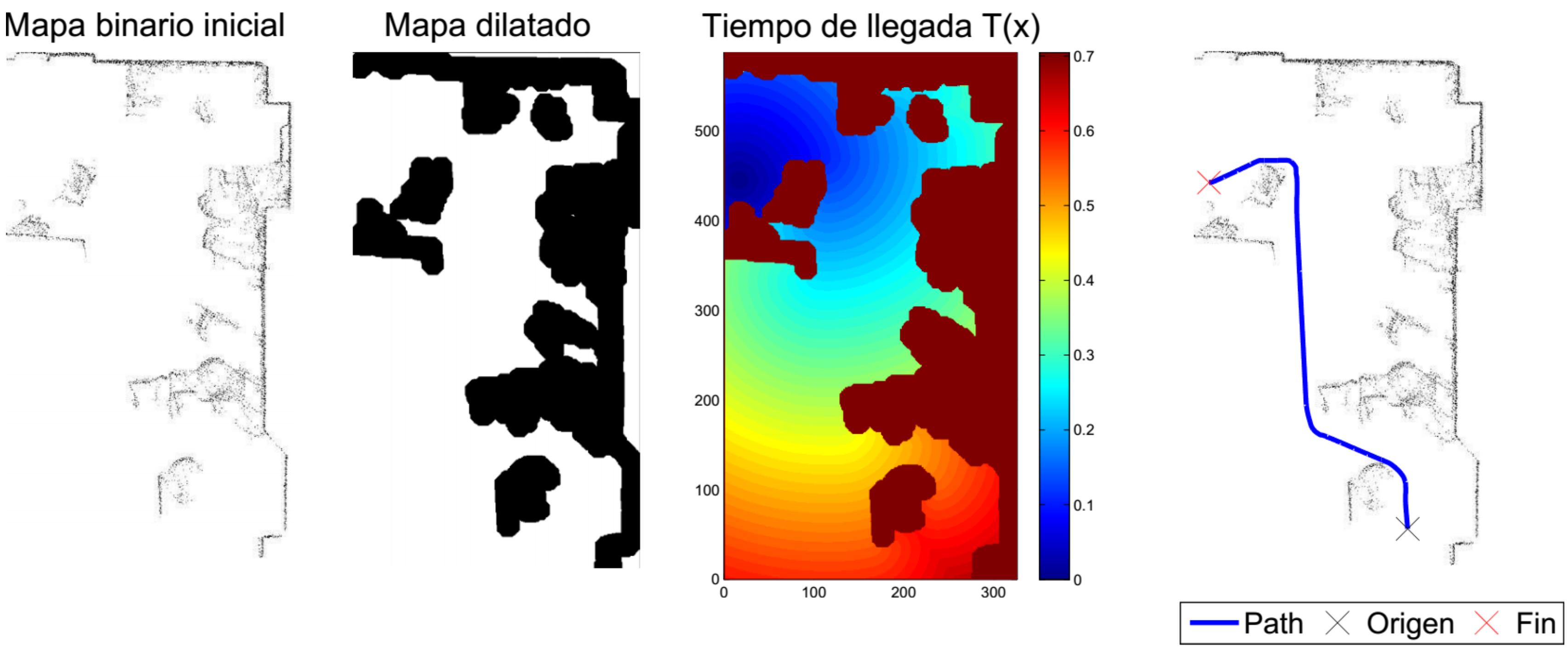


2.- FAST MARCHING SQUARE
Fast Marching Method (FMM)

- La expansión de la onda calcula el tiempo de llegada desde el origen hasta cada celda de la rejilla.
- Las celdas tienen tres posibles estados:
  - Unknown.
  - Narrow Band.
  - Frozen.

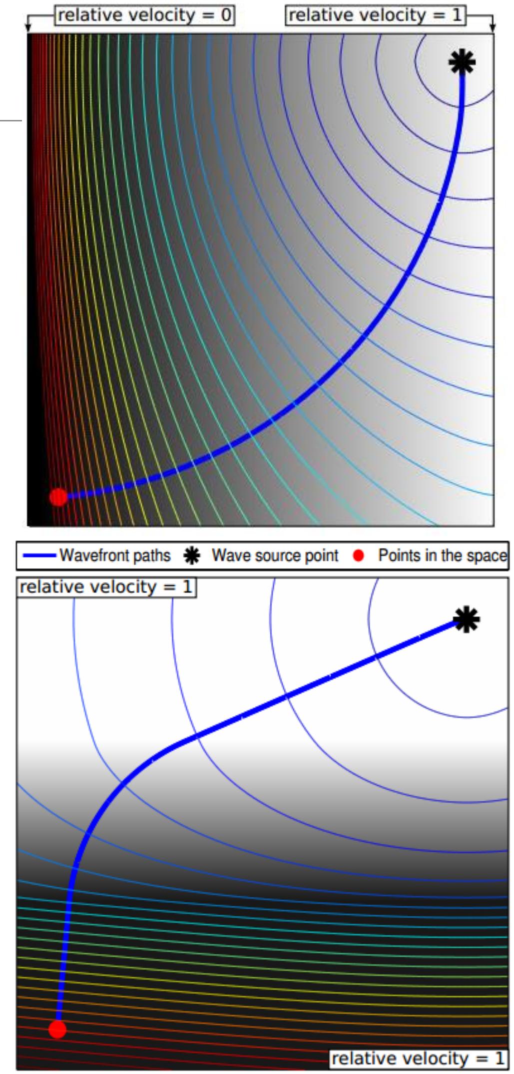


2.- FAST MARCHING SQUARE
Fast Marching Method (FMM)



2.- FAST MARCHING SQUARE

Fast Marching Method (FMM)



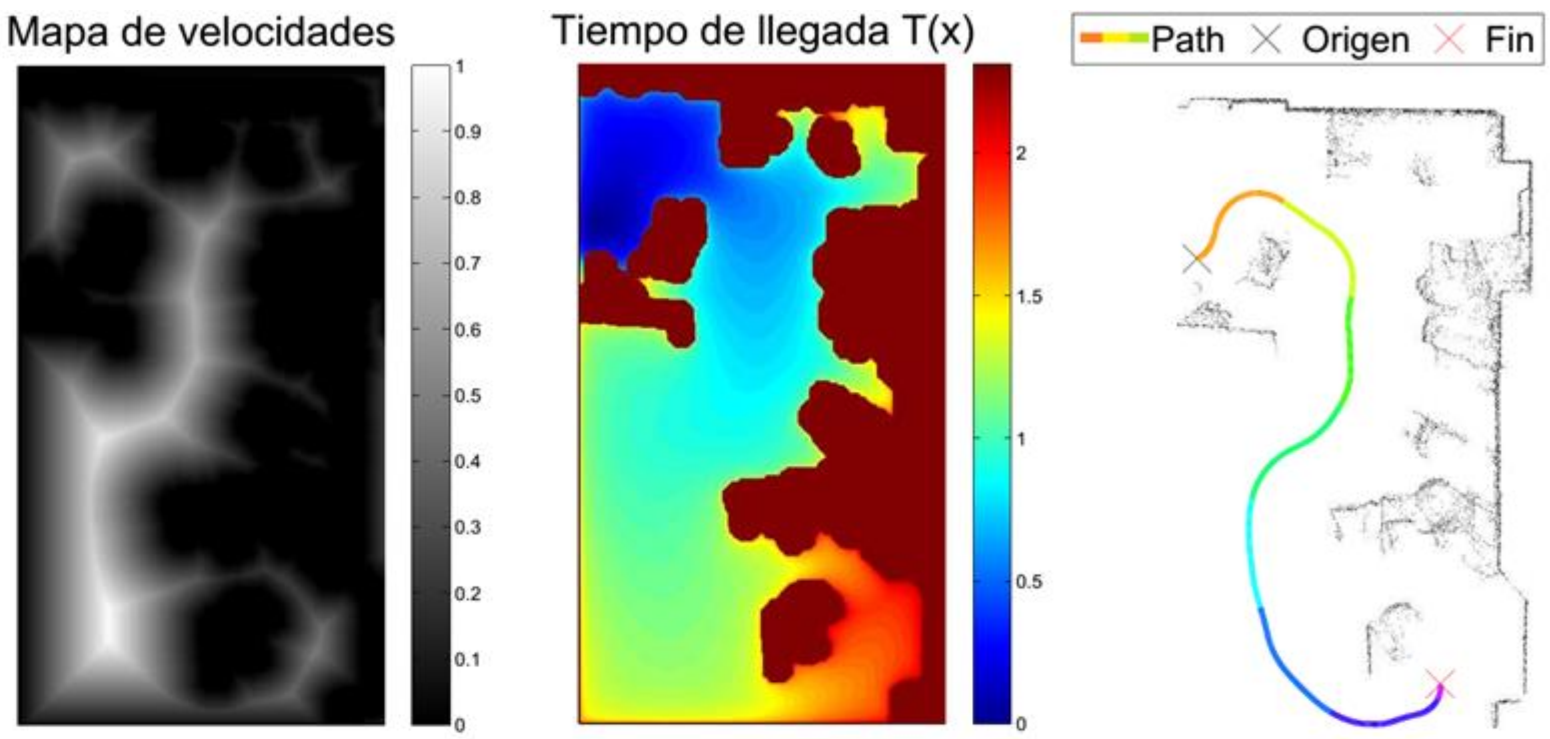
- FMM ofrece trayectorias óptimas acordes al criterio del tiempo mínimo de recorrido.
  - Esto implica trayectorias bruscas y próximas a los obstáculos.
  - La cinemática y la dinámica de los robots no pueden seguir esas trayectorias.
  - Los errores en la generación de los mapas hacen que se necesite una distancia de seguridad mínima a los obstáculos.
  - Es necesario suavizar los resultados.

2.- FAST MARCHING SQUARE FM^2

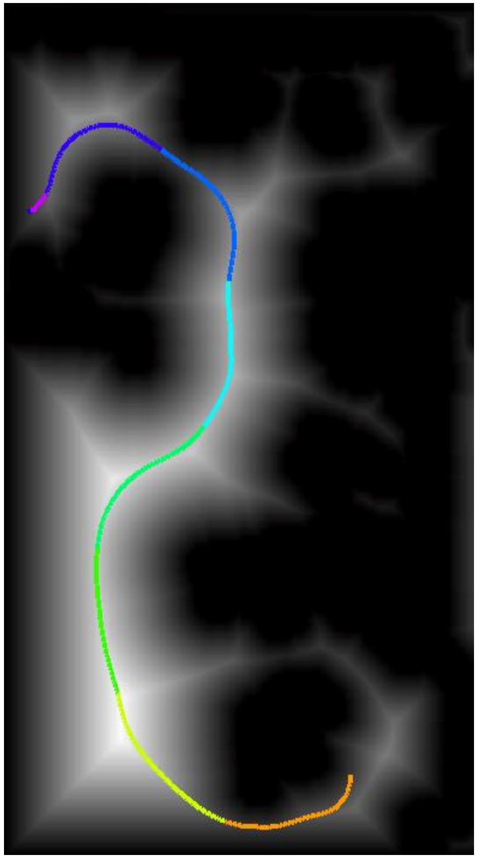
- Con FM^2 se propone un método basado en FMM que solucione las limitaciones anteriores utilizando mapas de velocidades.
  - La onda se expande a diferente velocidad en cada punto del espacio.
  - La velocidad aumentará a medida que la celda se aleje de los obstáculos.
  - Los mapas de velocidad se generan utilizando FMM.
  - Se obtienen resultados suaves y seguros para el robot.

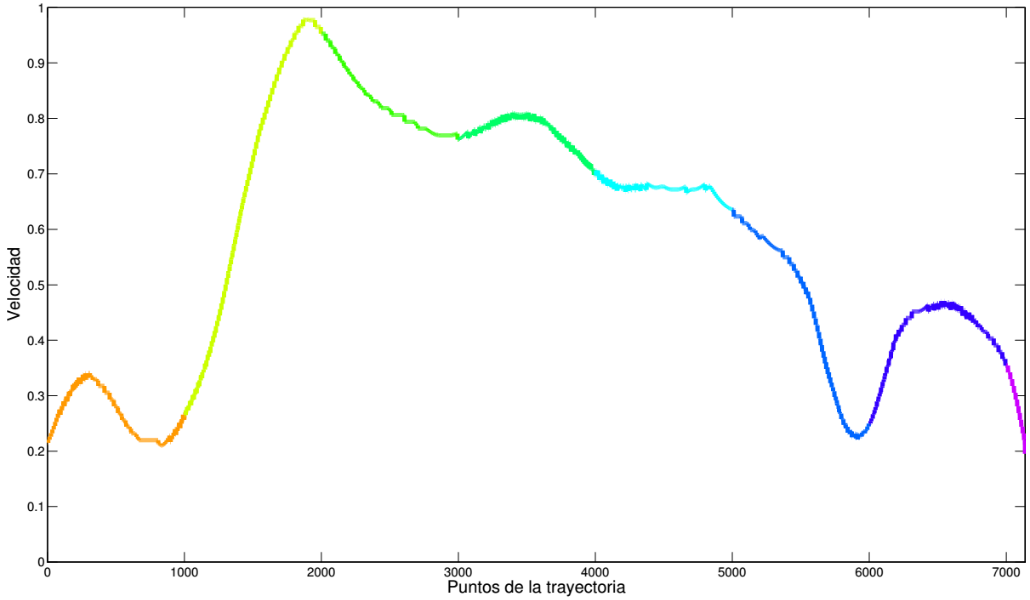
2.- FAST MARCHING SQUARE

Algoritmo



2.- FAST MARCHING SQUARE
Resultado





2.- FAST MARCHING SQUARE

Conclusiones

- Ventajas:
  - Soluciones seguras y suaves.
  - Sin mínimos locales.
  - Proporciona un perfil de velocidades.
  - Óptimo en términos de tiempo.

- Inconvenientes:
  - Costoso computacionalmente.
  - El mapa de velocidades se precalcula.
    - Puede generar trayectorias poco intuitivas en cuanto a velocidad y distancia a los obstáculos.

ÍNDICE

1. INTRODUCCIÓN
2. FAST MARCHING SQUARE
3. **FAST MARCHING SQUARE STAR**
4. FAST MARCHING SQUARE DIRECTIONAL
5. PRUEBAS EN TURTLEBOT
6. CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO

3.- FAST MARCHING SQUARE STAR FM^{2*}

- FM^{2*} busca rebajar el coste computacional de FM^2 sin perder sus características.

- Se orienta la onda de un modo análogo a A* añadiendo una heurística al tiempo de llegada.

- Existen dos aproximaciones:
  - FM^{2*} Clásico: desarrollado por Alberto Valero.

  - FM^{2*} Propuesto : desarrollado en esta tesis.

3.- FAST MARCHING SQUARE STAR

FM^{2*} Clásico

$$T' = T + heuristic$$

donde *T* es el tiempo de llegada calculado por FM^2 y *heuristic* es el valor de la heurística calculado como:

$$heuristic = \frac{d\_E(start, goal)}{robot\_max\_speed}$$

donde *d_E(start, goal)* es la distancia euclídea entre la celda por la que se expande la onda (*start*) y el objetivo (*goal*), *robot_max_speed* es la velocidad máxima permitida por el robot.

3.- FAST MARCHING SQUARE STAR

FM^{2*} Clásico

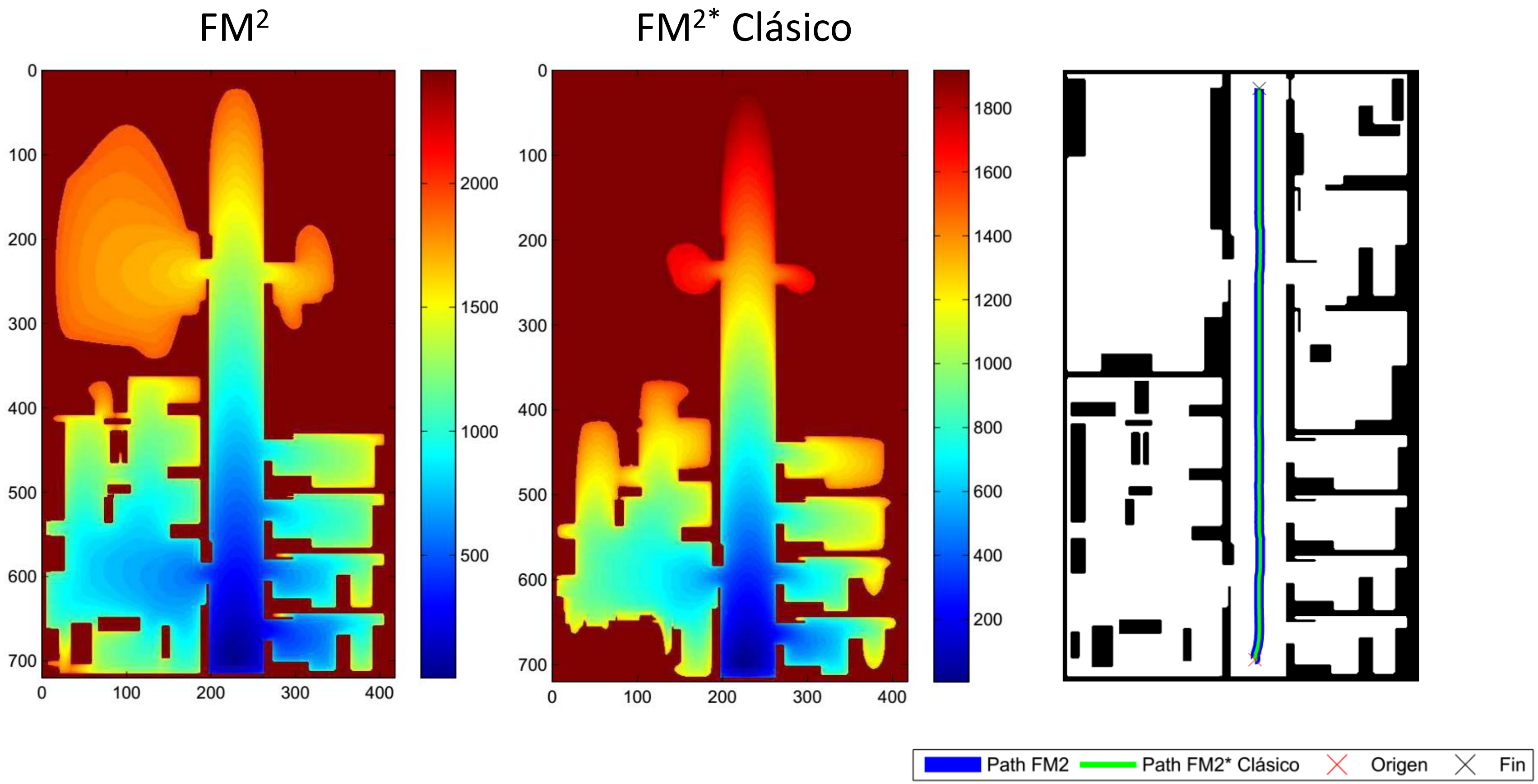
- El tiempo de llegada T’ es más favorable para el vecino Von Neumann más cercano al objetivo.
  - La onda se expande orientada al objetivo.

- La heurística no tiene en cuenta la distancia a los obstáculos, por lo que se pueden perder características de FM^2.

- Se almacenan dos tiempos de llegada:
  - *T’*: se utiliza para expandir la onda.
  - *T*: se utiliza para obtener la trayectoria.

3.- FAST MARCHING SQUARE STAR

FM^{2*} Clásico



3.- FAST MARCHING SQUARE STAR

FM^{2*} Clásico

- Conclusiones:
  - Trayectorias prácticamente idénticas a las de FM^2.
  - Mejora la carga computacional.
  - Se almacena una variable más → Mayor consumo de memoria.

3.- FAST MARCHING SQUARE STAR

FM^{2*} Propuesto

$$heuristic = \frac{d\_E(start, goal)}{cell\_speed}$$

donde *d_E(start, goal)* es la distancia euclídea entre la celda por la que se expande la onda (*start*) y el objetivo (*goal*), *cell_speed* es la velocidad en la celda según el mapa de velocidades.

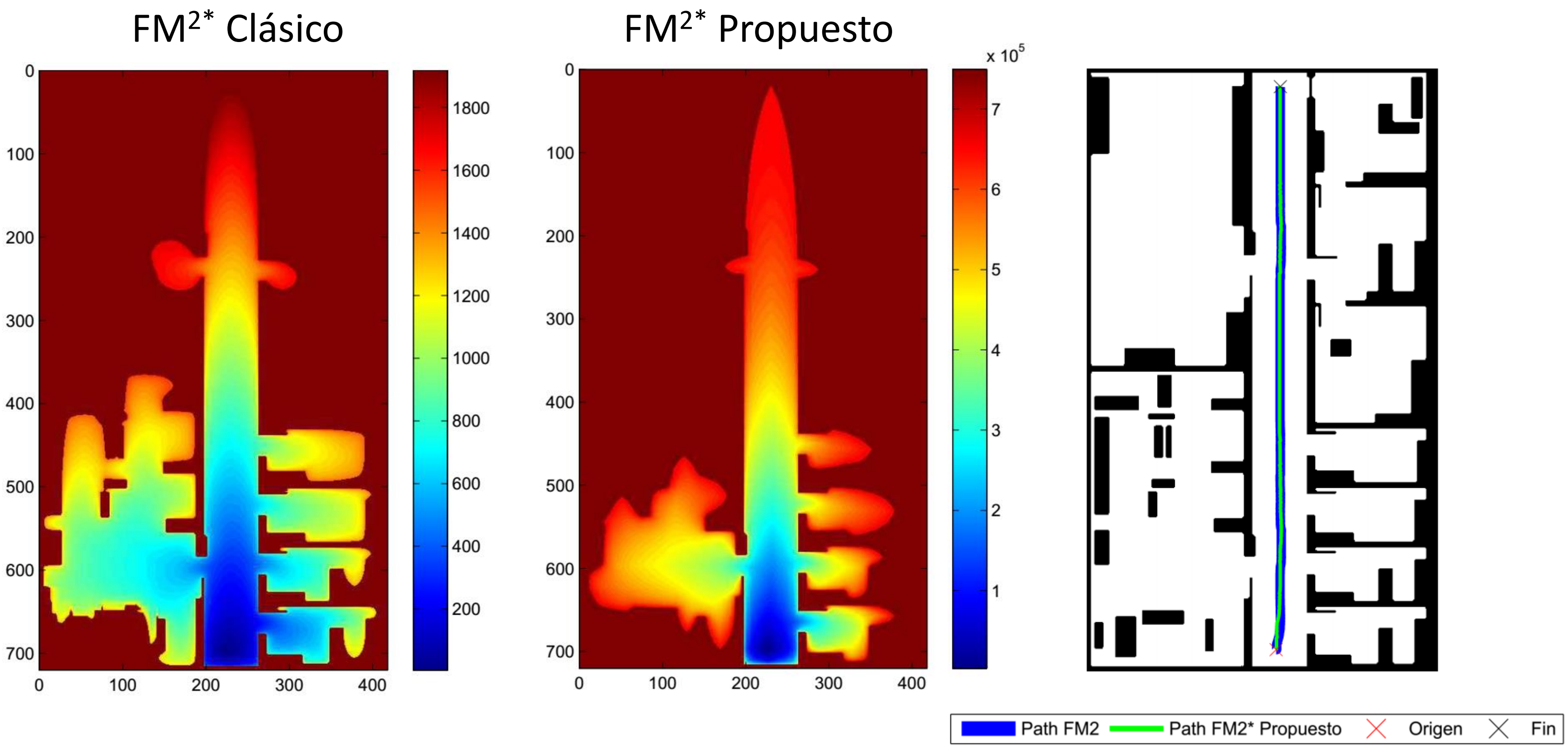
3.- FAST MARCHING SQUARE STAR

FM^{2*} Propuesto

- Tiene en cuenta la distancia a los obstáculos.
- Compromiso entre distancia hasta el punto de destino y distancia a los obstáculos.
- Se mantienen las características de FM^2.
- Sobre el valor T' se expande la onda y se aplica el gradiente.

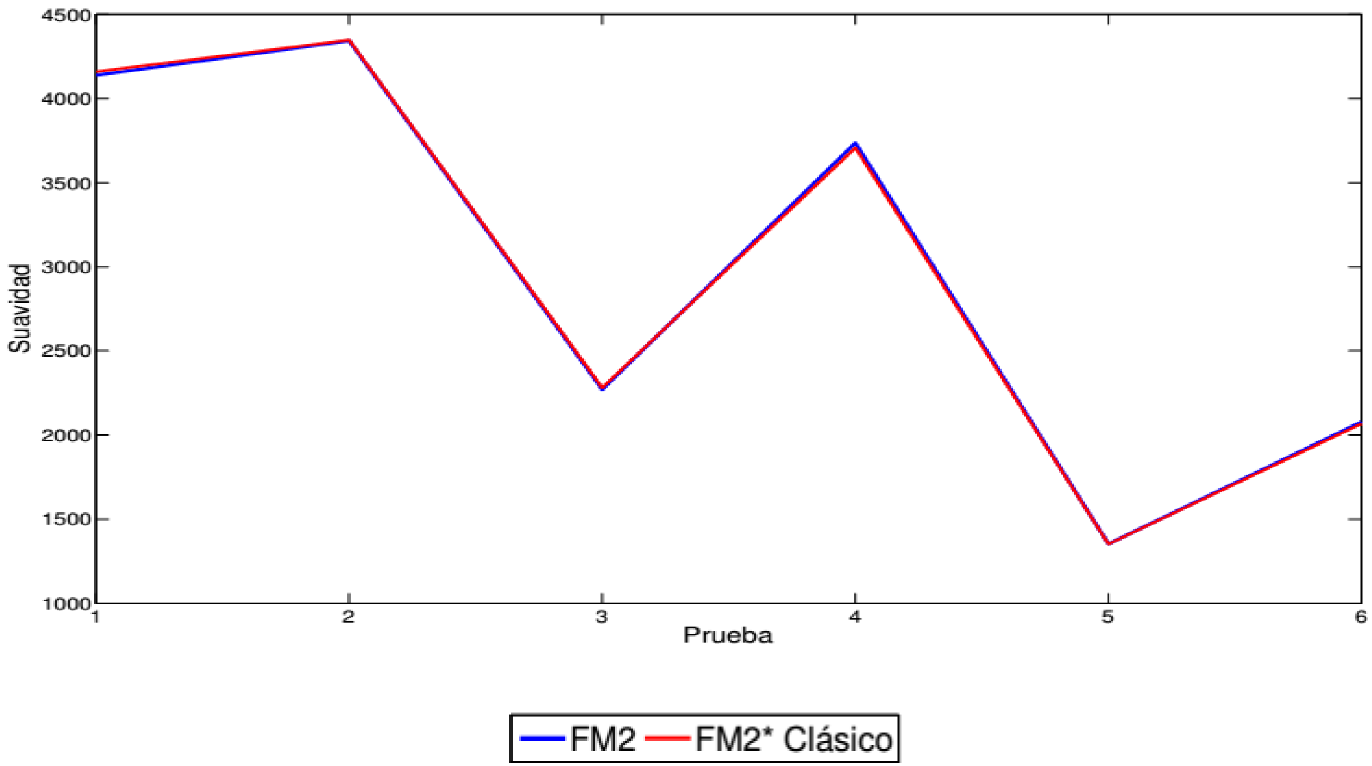
3.- FAST MARCHING SQUARE STAR

FM^{2*} Propuesto



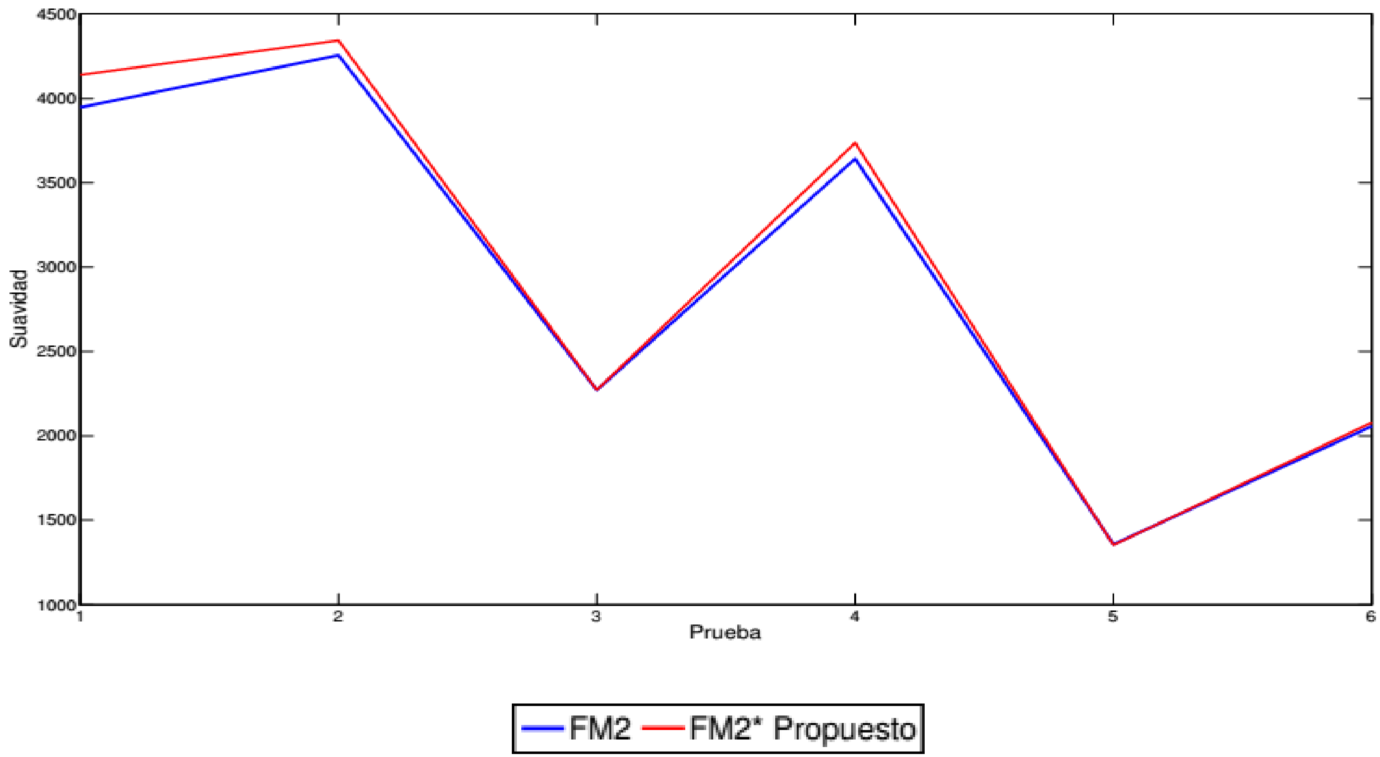
3.- FAST MARCHING SQUARE STAR
Comparativa de resultados

Suavidad de las trayectorias de FM^{2*} Clásico frente a FM^{2}.



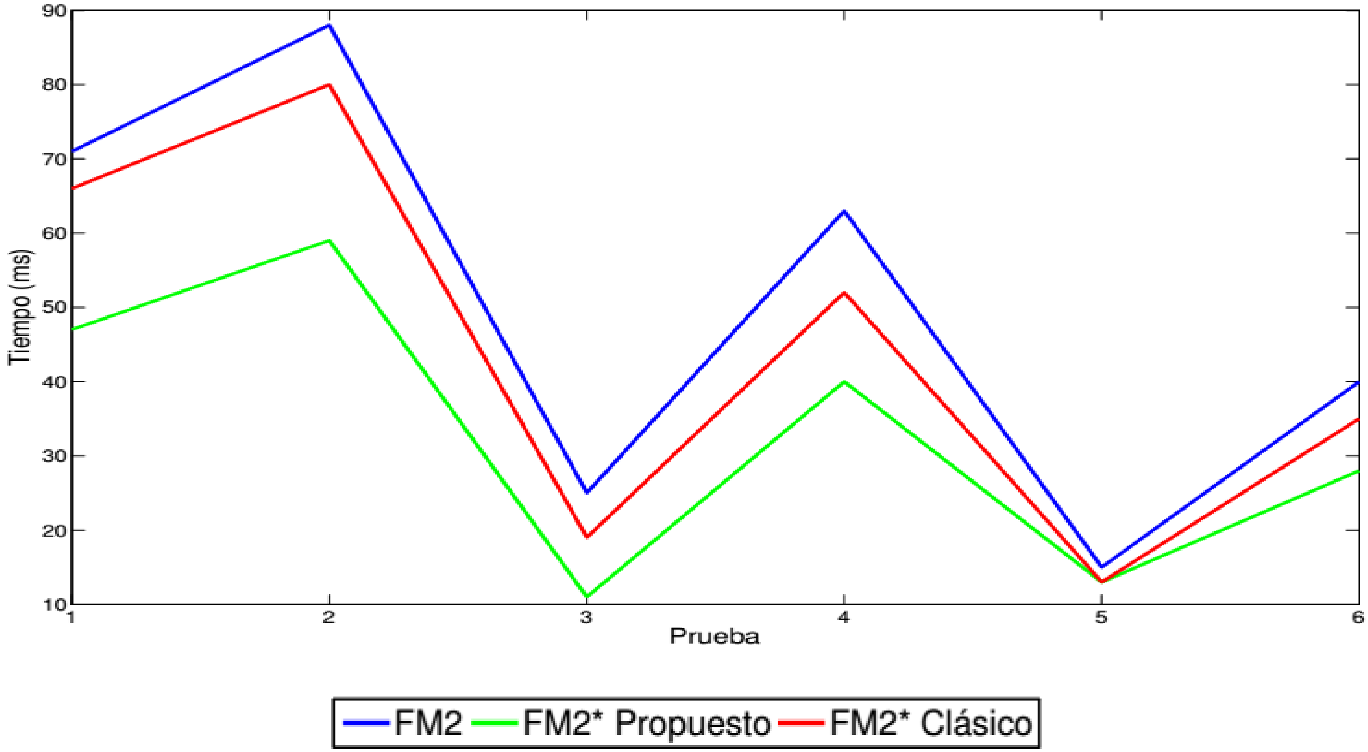
3.- FAST MARCHING SQUARE STAR
Comparativa de resultados

Suavidad de las trayectorias de FM^{2*} Propuesto frente a FM^2.



3.- FAST MARCHING SQUARE STAR
Comparativa de resultados

Tiempo de ejecución de los tres métodos.



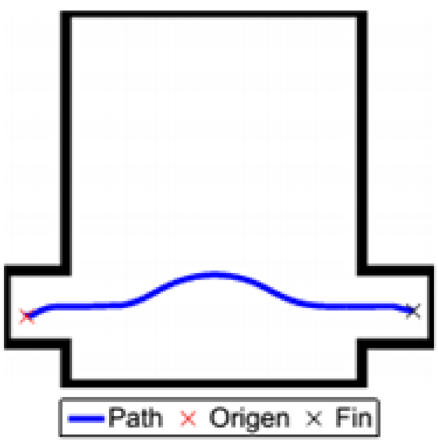
3.- FAST MARCHING SQUARE STAR
Conclusiones

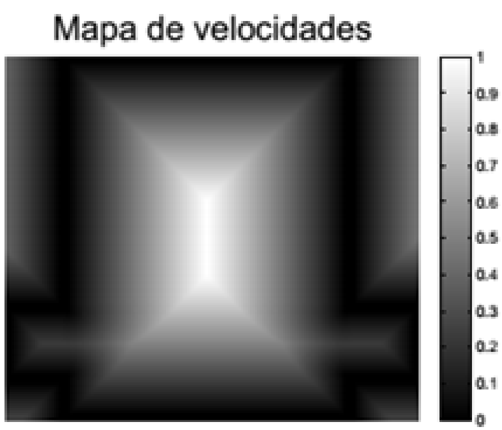
- Ventajas:
  - Los resultados del método desarrollado en esta tesis consiguen una orientación mayor de la onda.
  - Es computacionalmente más eficiente al utilizar un único valor de tiempo para expandir la onda y calcular el gradiente.
- Inconvenientes:
  - Las trayectorias obtenidas son muy similares a las obtenidas mediante FM^2, pero no idénticas.
    - El descenso de suavidad con respecto a FM^2 es mínimo.

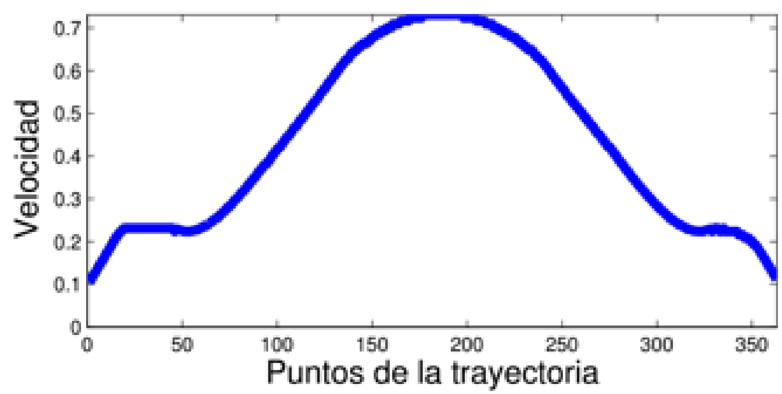
ÍNDICE

4.- FAST MARCHING SQUARE DIRECTIONAL
FM^2 Directional

- FM^2 propone trayectorias suaves para los robots.
- Se basa en mapas de velocidades, calculados previamente.
  - No tiene en cuenta los puntos de inicio y final.
  - Puede ofrecer trayectorias que recorren mayor espacio y con un perfil de velocidades más lento del que se espera de forma intuitiva.







4.- FAST MARCHING SQUARE DIRECTIONAL
FM^2 Directional

- FM^2 Directional modifica el mapa de velocidades durante la expansión de la onda.

- En cada instante se analiza como se expande la onda respecto a los obstáculos.
  - Si la onda se aleja → Se expande a máxima velocidad.

  - Si la onda se acerca → Se expande a la velocidad que marca el mapa de velocidades.

4.- FAST MARCHING SQUARE DIRECTIONAL

FM^2 Directional

- La velocidad utilizada se definirá tal que:

$$v_{exp} = \begin{cases} 1, & if\ v_{source} > v_{cell} \\ v_{cell}, & otherwise \end{cases}$$

donde:

v_{source}: Velocidad en la celda desde la que se expande.

v_{cell}: Velocidad en la celda a la que se va a expandir.

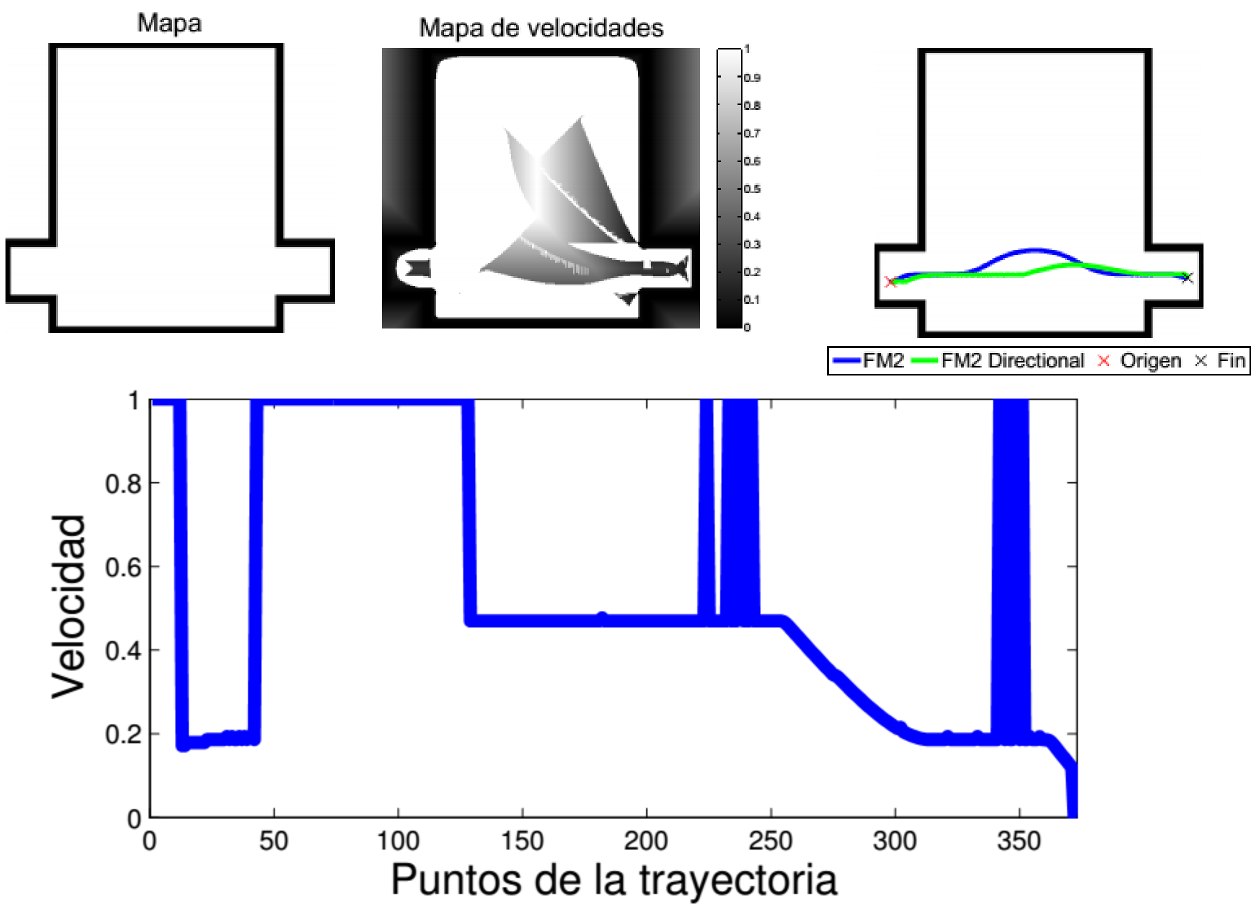
- Es necesario tener en cuenta que la onda se expande en el sentido contrario al camino resultante final.

4.- FAST MARCHING SQUARE DIRECTIONAL
FM^2 Directional

- La onda se expande de forma inesperada.
- Se almacenan dos tiempos de llegada:
  - *T*: se utiliza para expandir la onda.
  - *T'*: se utiliza para obtener la trayectoria y el perfil de velocidad.

4.- FAST MARCHING SQUARE DIRECTIONAL

FM^2 Directional



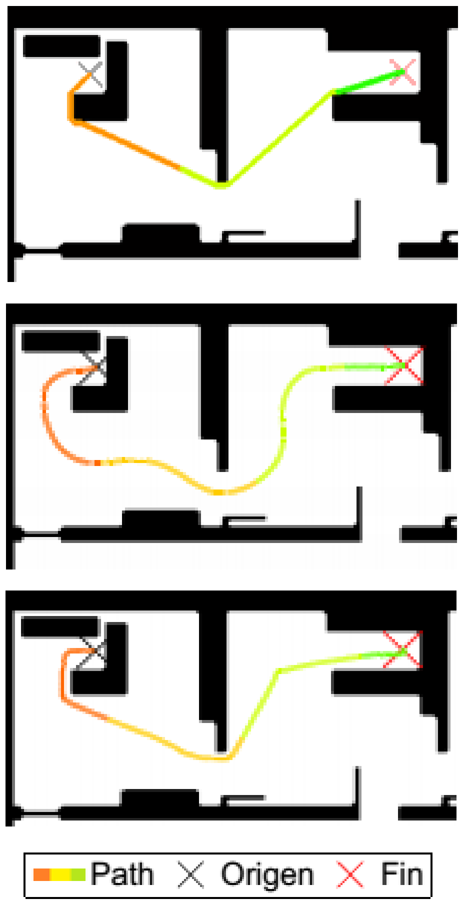
4.- FAST MARCHING SQUARE DIRECTIONAL

Ejemplos

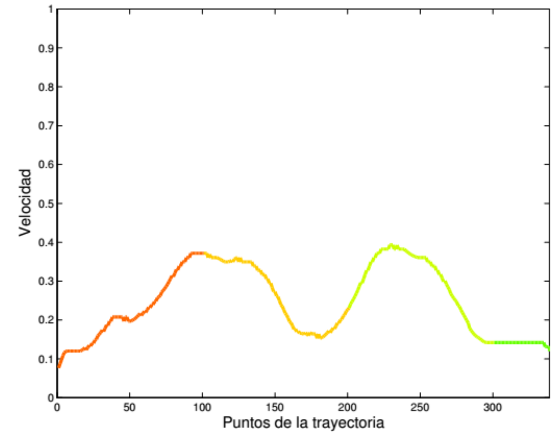
FMM

FM²

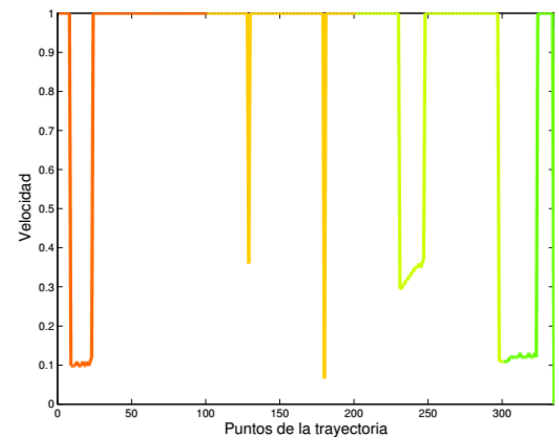
FM² Dir



FM²



FM² Directional

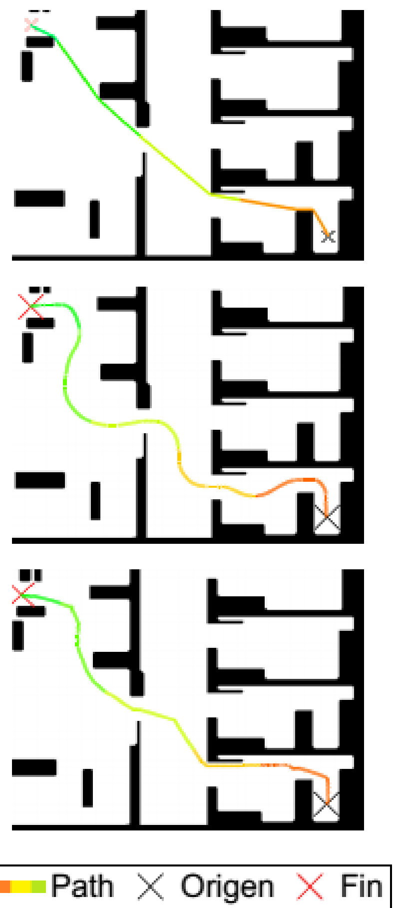


4.- FAST MARCHING SQUARE DIRECTIONAL
Ejemplos

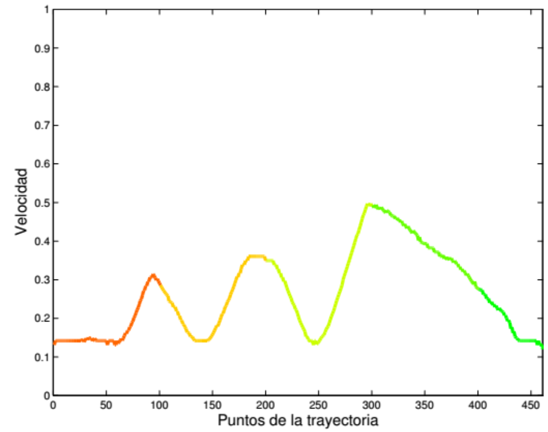
FMM

FM^2

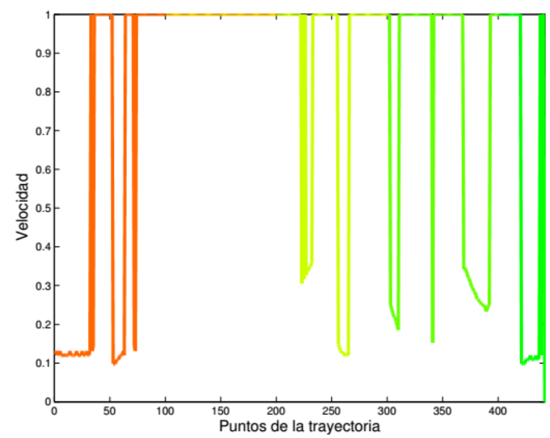
FM^2 Dir



FM^2

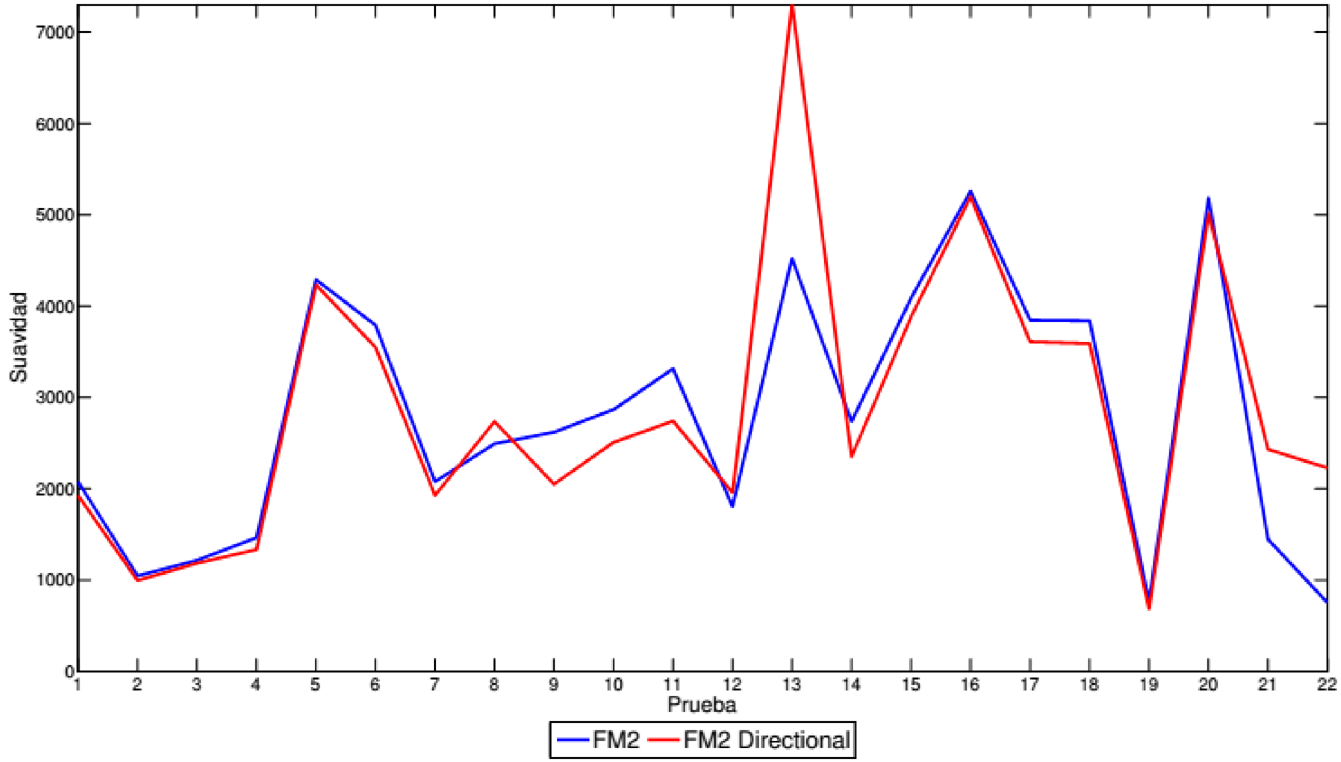


FM^2 Directional



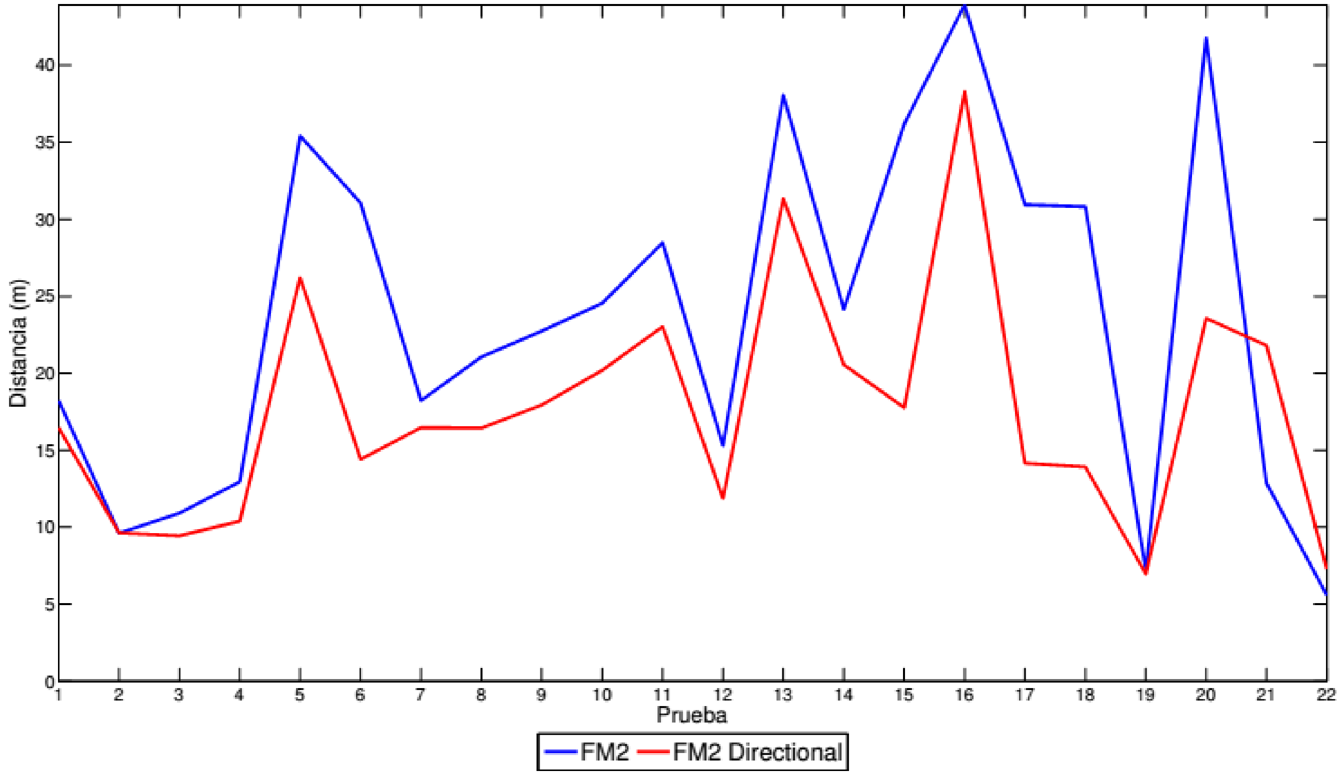
4.- FAST MARCHING SQUARE DIRECTIONAL
Resultados

Suavidad de las trayectorias de FM^2 Directional frente a FM^2.



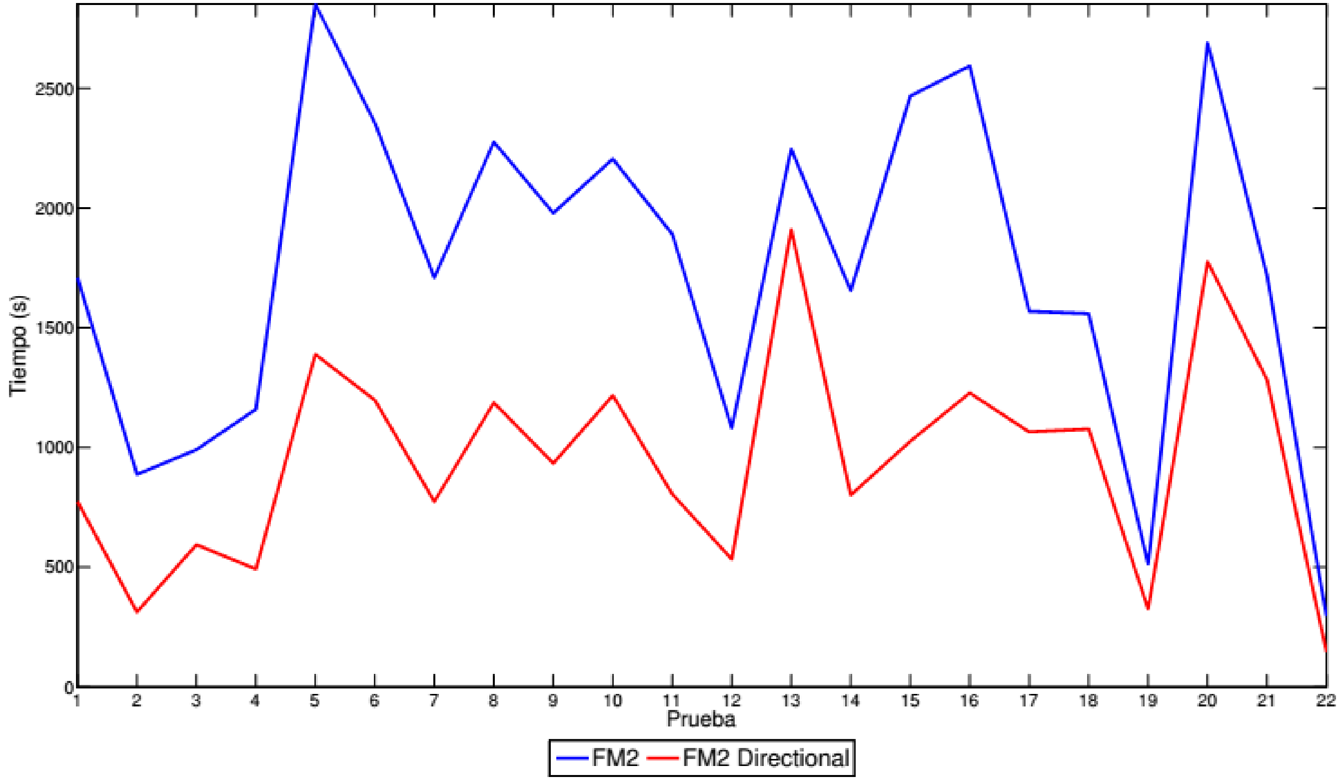
4.- FAST MARCHING SQUARE DIRECTIONAL
Resultados

Distancia recorrida de las trayectorias de FM^2 Directional frente a FM^2.



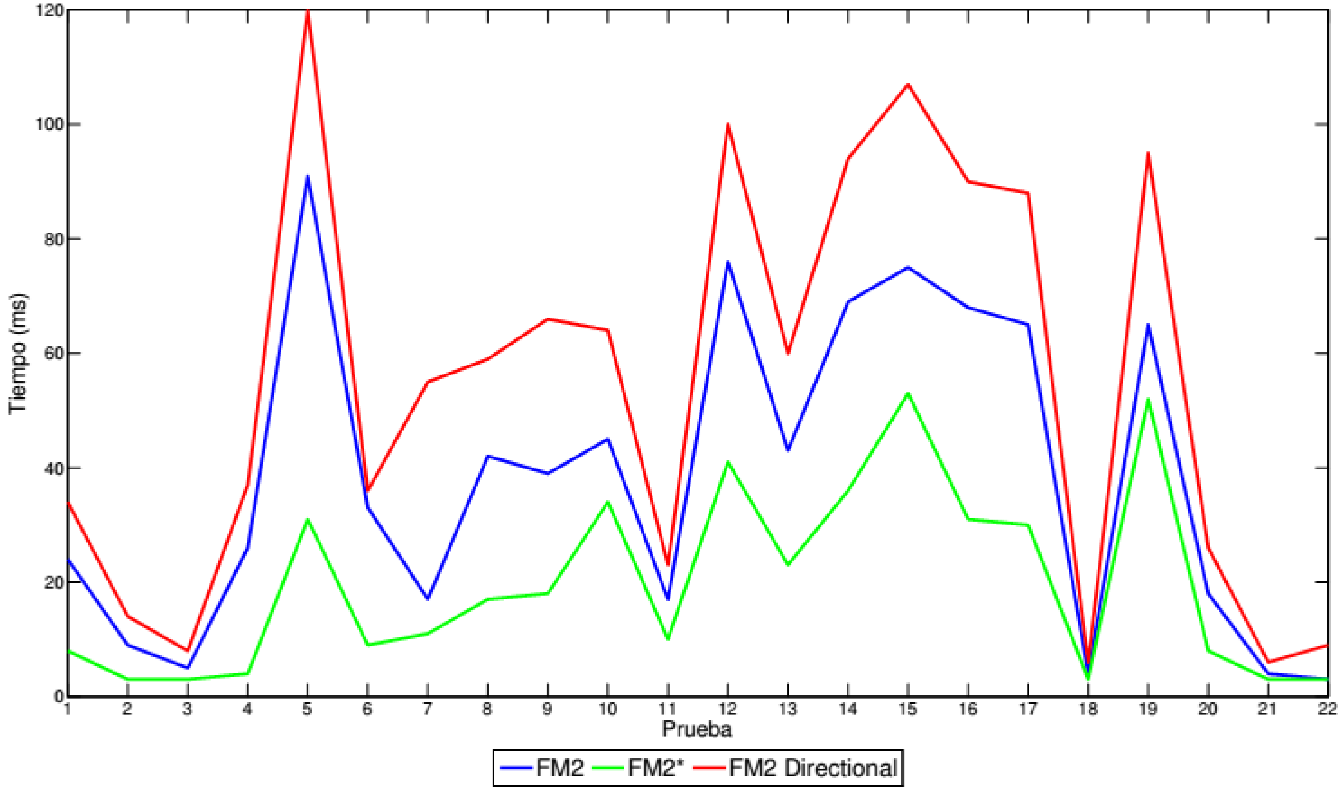
4.- FAST MARCHING SQUARE DIRECTIONAL
Resultados

Duración de las trayectorias de FM^2 Directional frente a FM^2.



4.- FAST MARCHING SQUARE DIRECTIONAL
Resultados

Tiempo de ejecución de FM^2, FM^{2*} y FM^2 Directional.



4.- FAST MARCHING SQUARE DIRECTIONAL
Conclusiones

- FM^2 Directional genera trayectorias y perfiles de velocidad coherentes entre sí.

- Se obtienen trayectorias:
  - Más cortas en distancia.
  - Más rápidas al recorrerlas.
  - Menos suaves.

- Es computacionalmente menos eficiente.

ÍNDICE

5.- PRUEBAS EN TURTLEBOT
Introducción

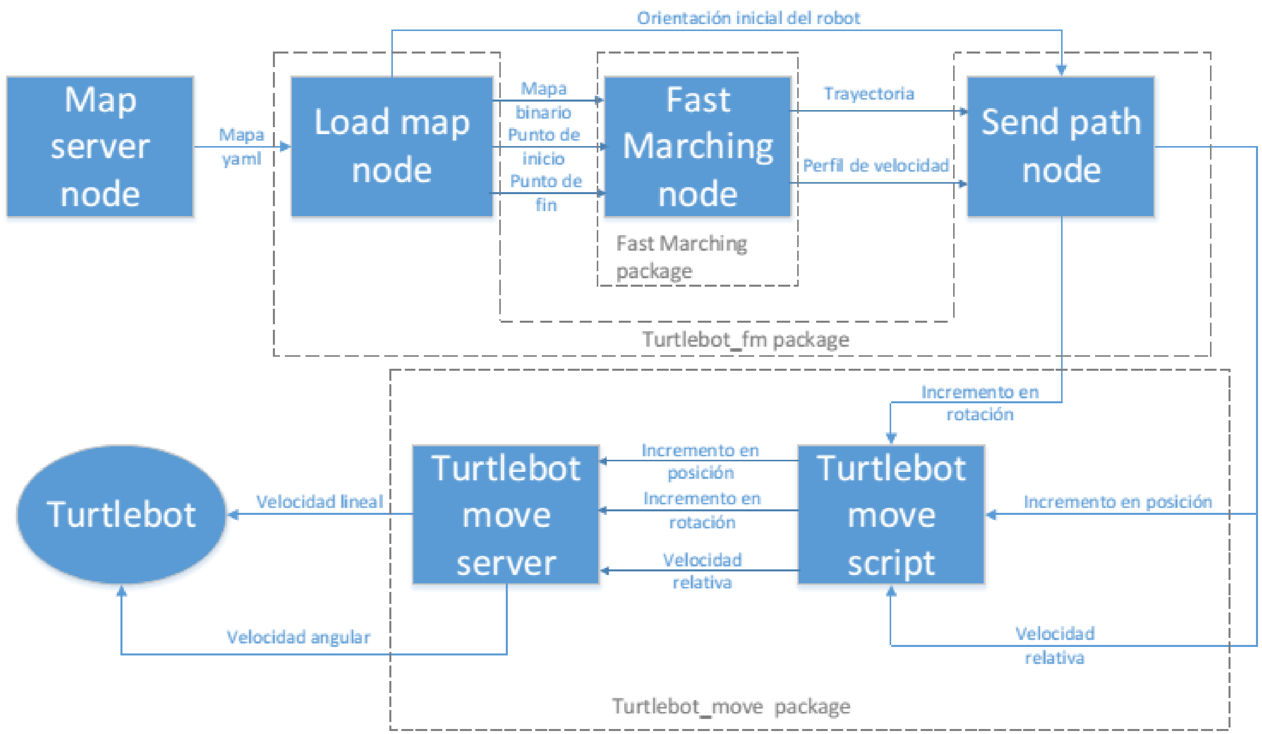
- Se comprobarán los algoritmos sobre una plataforma real.
- Se ha elegido el robot Turtlebot:
  - Robot móvil de dos ruedas.
  - Software y arquitectura de hardware libre.
  - Está formado por una base móvil, un ordenador portátil y una Kinect.
  - Compatible con ROS.
  - Muy extendido en investigación.

5.- PRUEBAS EN TURTLEBOT
ROS

- Framework orientado a la robótica.
- Provee servicios a bajo y alto nivel.
- Está formado por dos capas:
  - Sistema operativo.
  - Paquetes.
- Cada paquete contiene al menos un nodo.
- Funciona como una estructura de grafos:
  - Diversos nodos conectados entre sí.
  - Cada nodo realiza una tarea.
  - Pasos de mensajes entre ellos.
  - No tienen por qué estar en el mismo ordenador.

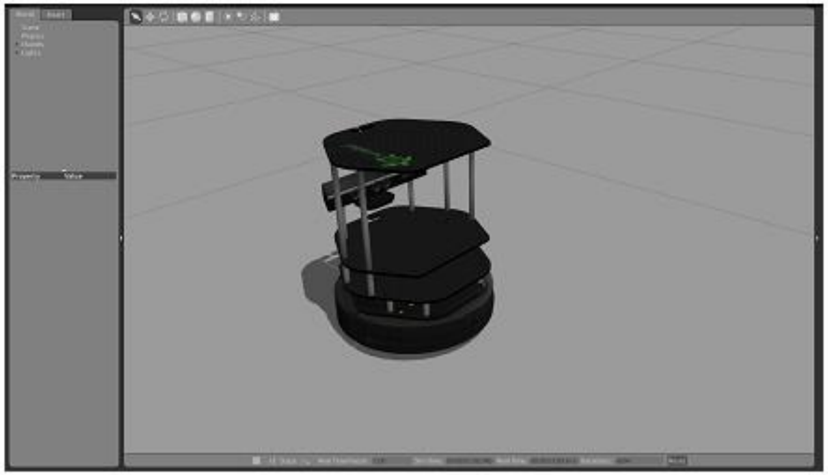
5.- PRUEBAS EN TURTLEBOT
Arquitectura propuesta

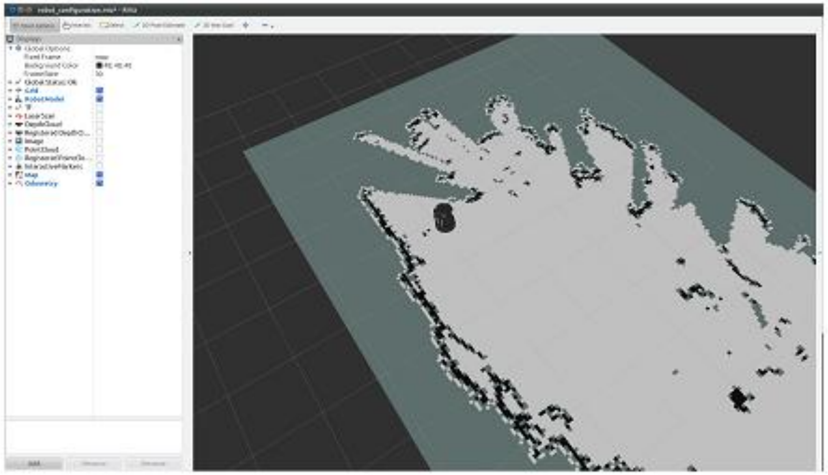
- Turtlebot_fm.
  - Load map.
  - Send path.
- Fast Marching.
  - FM^2.
  - FM^{2*}.
  - FM^2 Directional.
- Turtlebot_move.
  - Turtlebot move script.
  - Turtlebot move server.



5.- PRUEBAS EN TURTLEBOT
Pruebas

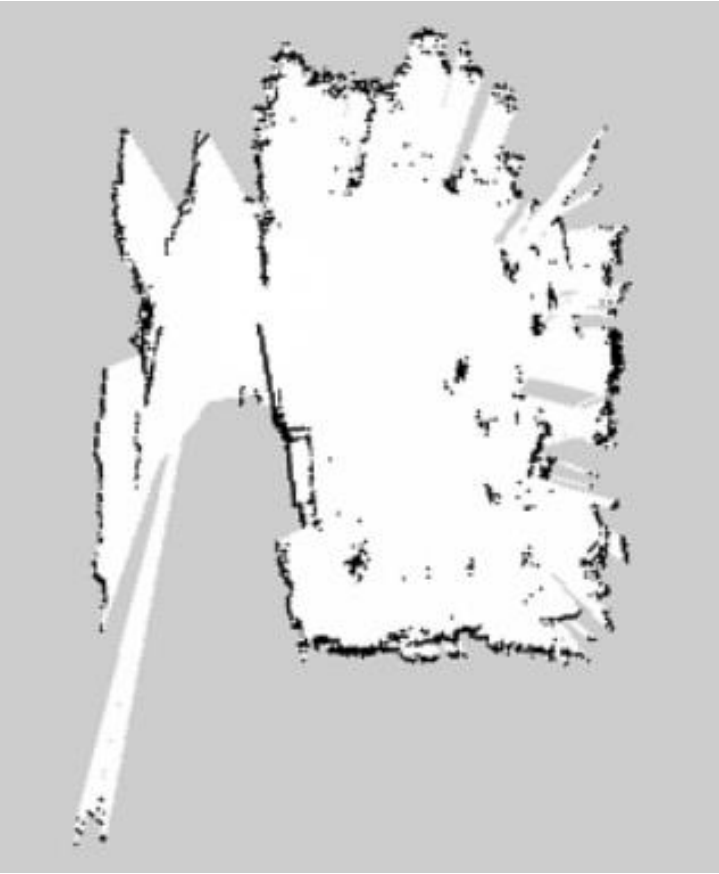
- Pruebas en simulación.





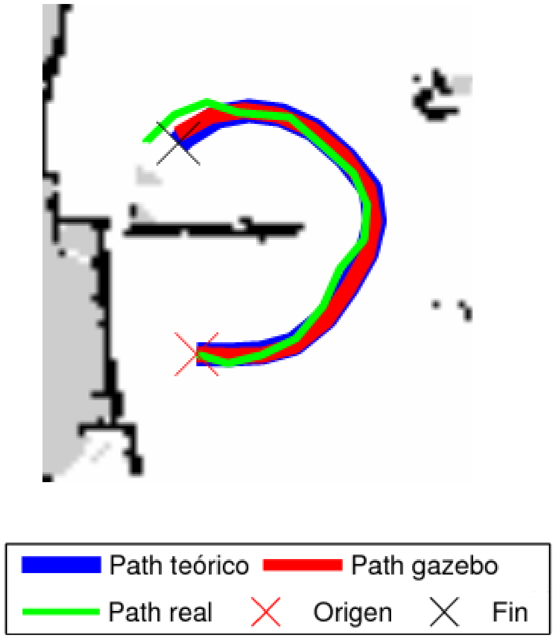
- Pruebas en sistema real.

5.- PRUEBAS EN TURTLEBOT
Pruebas en simulación



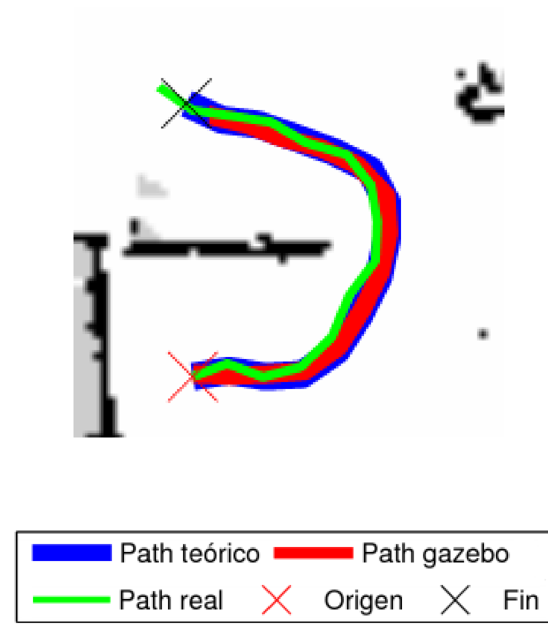
5.- PRUEBAS EN TURTLEBOT
Pruebas en sistema real

- FM^2 en mapa con obstáculos.

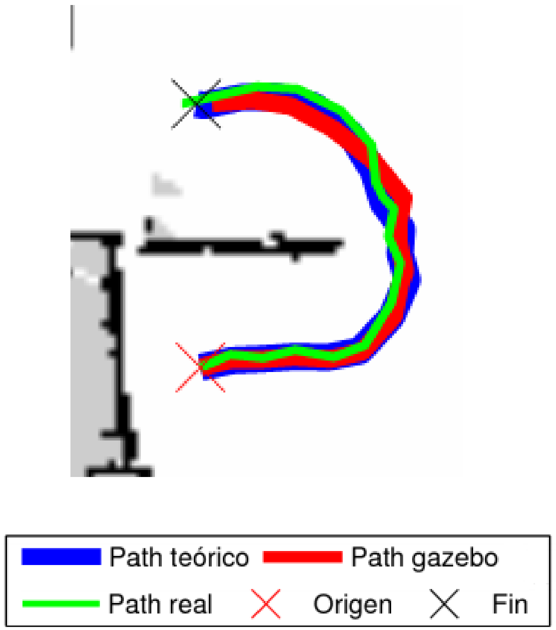
5.- PRUEBAS EN TURTLEBOT
Pruebas en sistema real

- FM^{2*} en mapa con obstáculos.



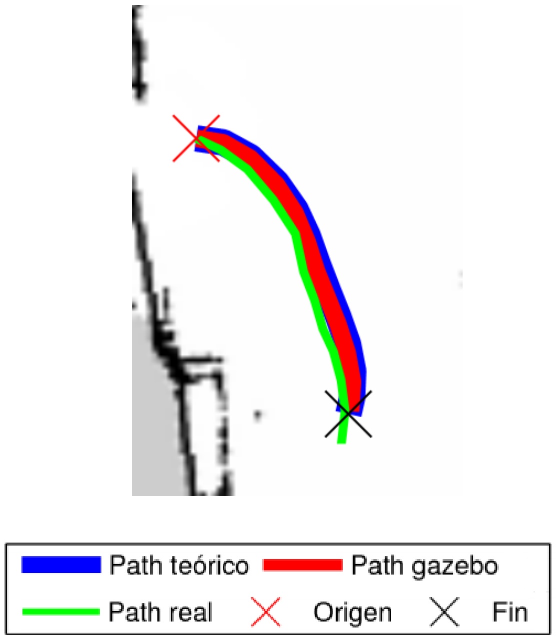
5.- PRUEBAS EN TURTLEBOT
Pruebas en sistema real

- FM^2 Directional en mapa con obstáculos.



5.- PRUEBAS EN TURTLEBOT
Pruebas en sistema real

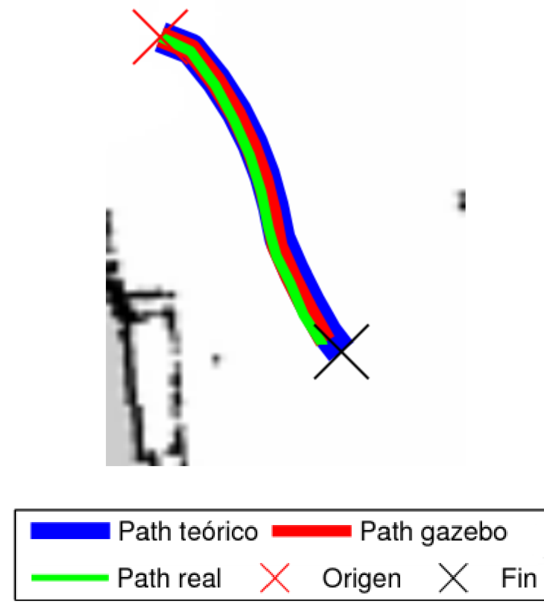
- FM^2 en mapa sin obstáculos.



5.- PRUEBAS EN TURTLEBOT

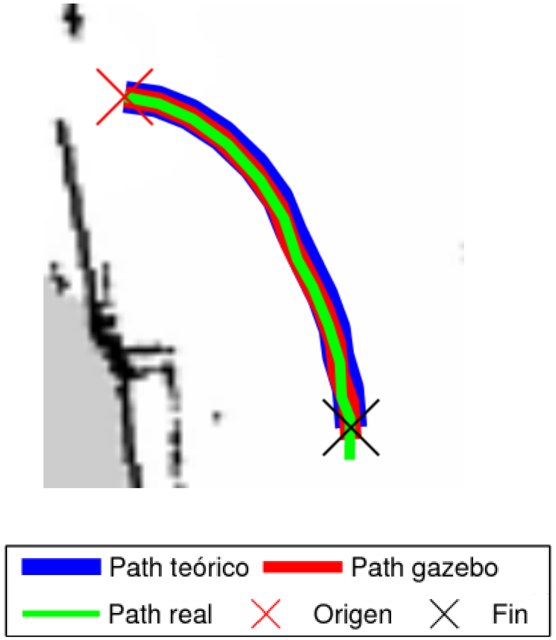
Pruebas en sistema real

- FM^{2*} en mapa sin obstáculos.



5.- PRUEBAS EN TURTLEBOT
Pruebas en sistema real

- FM² Directional en mapa sin obstáculos.



5.- PRUEBAS EN TURTLEBOT
Conclusiones

- Se han analizado los problemas que ocurren al aplicar los algoritmos a un sistema real.
- Resultados obtenidos con cierto grado de precisión.
- Limitaciones inherentes al robot.
  - No es diferencial.
  - Imposibilidad de realizar una trayectoria continua.
  - Baja resolución de los encoder.
- Los problemas se incrementan al pasar de simulación al sistema real.
  - Errores de precisión en los mapas.
  - Errores de precisión de los sensores en general.
  - Errores no sistemáticos externos al robot.
- La arquitectura implementada es válida.

ÍNDICE

6.- CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO

- Conclusiones:
  - Durante la tesis se han desarrollado variaciones de FM^2 que mejoren algunas de sus características:
    - FM^{2*} mejora su eficiencia computacional.
    - FM^2 Directional obtiene trayectorias más cortas y perfiles de velocidad más rápidos.
  - La elección del método depende de la aplicación.
  - La implementación en la plataforma real ha resultado satisfactoria dentro de las limitaciones de la plataforma.

- Trabajo futuro:
  - Probar el nodo propuesto como planificador en diferentes sistemas.
  - Continuar el desarrollo de dos nuevas variaciones de FM^2 propuestas:
    - FM^2 Directional basado en gradientes.
    - FM^{2*} Directional, método que combina FM^{2*} y FM^2 Directional.

Universidad Carlos III de Madrid

¡GRACIAS!

1.- FAST MARCHING METHOD (FMM) Pseudocódigo

```
input : A grid map G of size m × n
input : The set of cells Ori where the wave is originated
output: The grid map G with the T value set for all cells
Initialization;
foreach g_ij ∈ Ori do
    g_ij.T ← 0;
    g_ij.state ← FROZEN;
    foreach g_kl ∈ g_ij.neighbours do
        if g_kl = FROZEN then skip; else
            g_kl.T ← solveEikonal(g_kl);
            if g_kl.state = NARROW BAND then
            narrow_band.update_position(g_kl);
            if g_kl.state = UNKNOWN then
                g_kl.state ← NARROW BAND;
                narrow_band.insert_in_position(g_kl);
            end
        end
    end
    Iterations;
    while narrow_band NOT EMPTY do
        g_ij ← narrow_band.pop_first();
        foreach g_kl ∈ g_ij.neighbours do
            if g_kl = FROZEN then skip; else
                g_kl.T ← solveEikonal(g_kl);
                if g_kl.state = NARROW BAND then
                narrow_band.update_position(g_kl);
                if g_kl.state = UNKNOWN then
                    g_kl.state ← NARROW BAND;
                    narrow_band.insert_in_position(g_kl);
                end
            end
        end
    end
end
```

2.- FAST MARCHING SQUARE DIRECTIONAL
Método de gradiente