Universidad Zaragoza
1542

Trabajo Fin de Máster

ESTRATEGIAS MULTI-ROBOT DE DESPLIEGUE Y COBERTURA CON MANTENIMIENTO DE LA CONECTIVIDAD

MULTI-ROBOT ALGORITHMS FOR ADAPTIVE COVERAGE AND GLOBAL CONNECTIVITY MAINTENANCE

Autor

Javier Tardós Ibarra

Directores

Rosario Aragüés Muñoz
Carlos Sagüés Blázquiz

Escuela de Ingeniería y Arquitectura

2017

Estrategias Multi-Robot de Despliegue y Cobertura con Mantenimiento de la Conectividad

Resumen

Los sistemas multi-robot son un foco de investigación en la sociedad actual debido a las numerosas ventajas que presentan. Son sistemas robustos, escalables al tamaño del problema y que permiten una especialización de los individuos.

El principal tema que se aborda en este trabajo es la cobertura de una zona determinada por parte de un sistema multi-robot. Se trata de desarrollar un algoritmo de seguimiento de objetivos móviles partiendo de un algoritmo de cobertura existente. El algoritmo del que se parte realiza las labores de cobertura mediante divisiones de Voronoi iterativas, y mantiene la conectividad entre sus agentes por el método del Minimum Spanning Tree.

A la hora de extender sus capacidades al seguimiento de objetivos móviles, se plantean dos alternativas. Una consiste en aplicar funciones de importancia con centro en los objetivos para que el cálculo ponderado de las divisiones de Voronoi acerque a la flota hacia su meta. La otra, por su parte, consiste en modificar los límites de la zona de trabajo en función de la posición de los objetivos y de los propios agentes del sistema. Una comparación entre las dos alternativas permite concluir que el primer método es más rápido y más adecuado para casos en los que hay objetivos sueltos, mientras que el segundo es más adecuado para casos en los que hay concentraciones de objetivos.

También se realiza una implementación del sistema sobre el simulador Gazebo. Los robots se controlan mediante ROS y el algoritmo se ejecuta desde MATLAB. Además se desarrolla un sistema de visualización en Gazebo que permite comprender con mayor claridad el movimiento de cada uno de los robots del conjunto.

Finalmente, se realizan varios experimentos sobre esta implementación en Gazebo para comprobar el comportamiento del algoritmo. También se realiza un estudio paramétrico acerca de un experimento de seguimiento de una formación. En él se varían factores como la velocidad de la formación, el radio de los sensores de los robots o el número de agentes. Se destaca la importancia de realizar estudios como éste con carácter previo a una implementación real, dado que puede evitar derrochar recursos en robots y dispositivos de visión.

Agradecimientos

Quiero aprovechar estas líneas para agradecer la colaboración de todas las personas que han hecho posible llevar a cabo este trabajo.

A Rosario Aragüés:

Por toda la ayuda y el buen trato recibidos de su parte. Por enseñarme todos sus conocimientos en el ámbito del trabajo y por su dedicación e implicación en el mismo. Sin su ayuda el resultado no habría sido el mismo.

A Carlos Sagüés:

Por brindarme la oportunidad de realizar este trabajo en su grupo de investigación. Por aportar su amplia experiencia en los sistemas multi-robot y llevar el trabajo por buen cauce.

A Luis Riazuelo:

Por prestarme su ayuda desinteresada en el momento que más lo necesitaba, aportando sus amplios conocimientos en la materia.

Y por último a todos los miembros del Área de Ingeniería de Sistemas y Automática y, sobre todo, a los compañeros del laboratorio que hicieron de mi estancia una experiencia más amena y más completa.

Índice

Lista de símbolos

C_V	Centro de masa de la región V
$\mathcal{H}$	Función de coste
$J_{V,p}$	Momento polar de inercia de la región V respecto al punto p
M_V	Masa asociada a la región V
O	Centro de una formación de objetivos
o	Posición de los objetivos
P	Conjunto de posiciones de los agentes del sistema
p_i	Posición de los agentes del sistema
Q	Área de trabajo de la función de cobertura
q	Puntos del área de trabajo
s	Radio de observación de los sensores
$\mathcal{V}$	Partición de Voronoi
V_i	Región de Voronoi asociada al robot i
v	Velocidad de la formación
$\mathcal{W}$	Partición del área de trabajo
W_i	Región del área de trabajo
ϕ	Función de densidad sobre el área de trabajo

Lista de figuras

Capítulo 1

Introducción

1.1. Motivación y objetivos

Los robots, antaño considerados como meros elementos de ciencia ficción, siempre han sido un foco de investigación para la humanidad moderna. La ambición que muestra la sociedad en la robótica es debida a la gran capacidad de estos sistemas para facilitar tareas sistemáticas, costosas o peligrosas para el ser humano. Gracias a ello se están integrando progresivamente en la vida diaria y en procesos industriales. Lo hacen en formas más simples, como las aspiradoras autónomas de uso doméstico o los brazos robóticos de una línea de montaje, y en formas más complejas en desarrollo, como robots humanoides que saben interactuar con personas.

Un campo de desarrollo de la robótica actual consiste en la coordinación de sistemas en los que varios robots interaccionan entre sí, en adelante sistemas multi-robot. Esto plantea nuevos problemas a la par que presenta grandes ventajas. Uno de los problemas que surgen consiste en que la flota necesita mantener la conectividad en cuanto a las comunicaciones. De otro modo, el conjunto perdería la capacidad de realizar tareas de cooperación, intercambiar datos o aceptar órdenes. Por otra parte, como ventajas destacan la robustez del sistema, la escalabilidad al tamaño del problema con sólo variar el número de nodos, y la posibilidad de especialización de los robots al poder realizar cada uno tareas distintas.

Entre las problemáticas que se suelen abordar en sistemas multi-robot destacan el control de formaciones, evitación de obstáculos, cobertura de una zona determinada, tareas de búsqueda y rescate, transporte cooperativo, etc. En este Trabajo Fin de Máster se va a desarrollar un algoritmo de cobertura y seguimiento de un conjunto de objetivos móviles. Este tipo de algoritmos se implementan, por ejemplo, para que una flota de cuadrotores realice tareas de vigilancia y monitorización, para que una flota terrestre ofrezca funciones de apoyo a otra aérea, y otras finalidades similares.

Para el desarrollo del algoritmo se partirá de un método previo de despliegue de la flota de robots a posiciones fijas [1] y se modificará de dos maneras para que realice la tarea de seguimiento. La primera consiste en dotar de funciones de importancia a los objetivos móviles para que la flota tienda a dirigirse hacia ellos. La segunda, por otra parte, se basa en modificar los límites ficticios de la región a cubrir en función de la posición de los objetivos. Habrá que tener en cuenta que siempre se ha de mantener un árbol de comunicaciones que garantice la conectividad entre los agentes. Estos algoritmos se desarrollarán en MATLAB [2].

Además se presentará una implementación del algoritmo en el software de simulación Gazebo [3]. Se trata de una plataforma donde se puede diseñar robots, probar su comportamiento, generar escenarios realistas y, sobre todo, simularlos con su motor físico. Se reproducirá el caso en el que

un grupo de agentes terrestres sirven de apoyo a una flota de cuadrotores, ya que es más sencillo controlar los robots terrestres. En este caso se empleará el Turtlebot, que es un robot de cinemática diferencial con una cámara Kinect integrada. Para controlar los robots y para comunicar la simulación de Gazebo con el algoritmo de MATLAB se empleará el software ROS (Robot Operating System [4]), que dispone de paquetes, controladores, etc. para estas funcionalidades.

Así pues, los objetivos del presente Trabajo Fin de Máster son:

- Desarrollar un algoritmo de cobertura con mantenimiento de la conectividad para que un sistema multi-robot realice tareas de seguimiento.
- Implementar el sistema y el algoritmo sobre una plataforma de simulación realista.
- Comprobar el funcionamiento del algoritmo mediante casos concretos y un estudio paramétrico.

1.2. Estado del arte

En la temática de los sistemas multi-robot destacan dos ramas bastante diferenciadas a la hora de diseñar algoritmos. Una es la ejecución centralizada del algoritmo, en la que los agentes envían los datos de sus sensores hasta cierto robot o hasta otro dispositivo central. Éste es el encargado de calcular el algoritmo y enviar de vuelta las órdenes adecuadas a todos los robots. La otra rama consiste en diseñar los algoritmos de forma distribuida, esto es, que todos los agentes en conjunto se comunican y dan órdenes entre sí. Unos mensajes van dando lugar a otros y el algoritmo se completa sin necesidad de un nodo central que conozca todos los datos. Este segundo método de diseño es bastante más complejo y suele requerir un número muy elevado de mensajes entre los nodos del sistema. Por estos motivos, en este trabajo se va a diseñar el algoritmo de forma centralizada, dejando para un trabajo futuro el desempeño de una versión puramente distribuida.

Centrándose en los algoritmos de cobertura, el problema que se presenta es en qué posición colocar los nodos para desempeñar de manera más eficaz la labor de cobertura de una región. Cada robot tiene un rango de visibilidad determinado, de tal manera que no es capaz de observar correctamente puntos del área de trabajo que se encuentren muy alejados. Por tanto, hay que conseguir que los nodos estén distribuidos de la mejor manera posible, minimizando las distancias con cualquier punto del área de trabajo. Este problema es conocido como optimización de la localización y está presente en muchísimas disciplinas, desde la situación en una ciudad de los buzones de correo, hasta el estudio de la distribución de los territorios animales en la naturaleza.

En lo que a la robótica se refiere, existen diferentes métodos para llevar a cabo esta optimización de la localización. Como se recoge en [5], se emplean divisiones de Voronoi, modelos probabilísticos o campos potenciales artificiales, entre otros. En este trabajo se empleará un método basado en dividir iterativamente el espacio de trabajo en regiones de Voronoi, que dará pie a las alternativas que se proponen para realizar la labor de seguimiento de unos objetivos móviles.

A menudo, en la literatura se olvida la necesidad de mantener los robots a cierta distancia para no perder la conectividad entre ellos. En este trabajo se empleará uno de los posibles métodos para este fin, el denominado MST (Minimum Spanning Tree, traducido sería algo como árbol abarcador mínimo). Existen versiones descentralizadas para implementar este método [6], pero se diseñará de forma centralizada dado que el resto del algoritmo se ejecuta así. Este formato es bastante sencillo y consiste en construir el mínimo árbol de comunicaciones que permita la mayor libertad de movimiento posible al conjunto. El árbol se habrá de ir actualizando a medida que los agentes se mueven.

1.3. Organización de la memoria

La memoria se divide en seis capítulos cuyo contenido se explica a continuación.

Tras la presente introducción, en el Capítulo 2 se describe el funcionamiento del algoritmo de cobertura con mantenimiento de la conectividad del cual se parte para elaborar las propuestas de seguimiento. Una primera parte dedicada a la cobertura por divisiones de Voronoi iterativas, y otra parte dedicada al mantenimiento de la conectividad mediante el árbol mínimo de comunicaciones.

A continuación, en el Capítulo 3 se presentan dos alternativas para que el algoritmo de cobertura realice las tareas de seguimiento. Se explica su funcionamiento básico y se ilustra mediante algunos ejemplos. En base a estos ejemplos también se realiza una comparación de su comportamiento.

El Capítulo 4 abarca el proceso de implementación del sistema y el algoritmo sobre una plataforma de simulación realista. Se describen brevemente los softwares empleados y el procedimiento para obtener un conjunto de robots funcionales. También se incluye un sencillo sistema de visualización para el simulador.

En el Capítulo 5 se expone una serie de simulaciones con el fin de estudiar el comportamiento del algoritmo. Por un lado, se simulan varios casos muy concretos sobre la plataforma de simulación. Por otro, se realiza un estudio paramétrico de un experimento de seguimiento de una formación.

Finalmente, el Capítulo 6 contiene las conclusiones desprendidas del trabajo y las posibles líneas futuras de investigación.

Capítulo 2

Descripción del algoritmo de cobertura

En este apartado se explican las bases del algoritmo de cobertura con mantenimiento de la conectividad del cual se parte para, en posteriores capítulos, establecer diferentes modos de realizar tareas de seguimiento.

El algoritmo se divide en dos partes fundamentales. La primera, la labor de despliegue y cobertura llevada a cabo mediante particiones de Voronoi. La segunda, el método de mantenimiento de la conectividad entre los agentes del sistema multi-robot, denominado MST.

2.1. Cobertura por divisiones de Voronoi

Como se ha mencionado en la introducción, en la robótica existen diferentes métodos para resolver el problema de optimización de la localización que plantea el algoritmo de cobertura. Los nodos se deben situar de la manera más distribuida posible para minimizar las distancias entre todos los puntos de la zona de interés y alguno de los robots. En este trabajo se escoge la resolución por particiones de Voronoi, un método intuitivo mediante el cual se consigue la distribución uniforme de la flota.

Se trata de un método iterativo en el que se calculan las nuevas posiciones de los robots a partir de sus posiciones actuales. Por tanto, la solución que ofrece este método es una solución local, que depende de las posiciones iniciales de los nodos. A continuación se procede a explicar el proceso matemático que conlleva el método de las divisiones de Voronoi.

El área de trabajo, Q, es un polígono convexo sobre el que se desea desplegar n agentes. La función $f(\|q - p_i\|)$ describe el rendimiento del sensor de los robots, que depende de la distancia. Sea $\phi(q)$ una función de densidad en Q y $P = \{p_1, ..., p_n\}$ las posiciones de los robots, el problema de optimización de la localización consiste en minimizar la función de coste [1]

$$\mathcal{H}(P, \mathcal{W}) = \sum_{i=1}^{n} \int_{W_i} f(\|q - p_i\|)\, \phi(q)\, dq, \tag{2.1}$$

donde $\mathcal{W} = \{W_1, ..., W_n\}$ es una partición del área de trabajo cuya unión es Q. Si los robots están fijos entre iteraciones, $\mathcal{W}$ se corresponde con la partición de Voronoi, $\mathcal{V} = \{V_1, ..., V_n\}$. La región de Voronoi asociada a un robot es el conjunto de puntos que se encuentran a una distancia menor de él que del resto de agentes:

$$V_i = \{q \in Q \mid \|q - p_i\| \leq \|q - p_j\|,\ \forall j \neq i\} \tag{2.2}$$

Para hacerse una idea rápida de lo que son las regiones de Voronoi antes de continuar explicando el algoritmo, en la Figura 2.1 se representa el resultado de aplicar el método en una zona cuadrada de 30 metros de lado cubierta por 7 robots y donde no hay funciones de densidad. Se observa que la distribución es uniforme y abarca toda el área de interés.

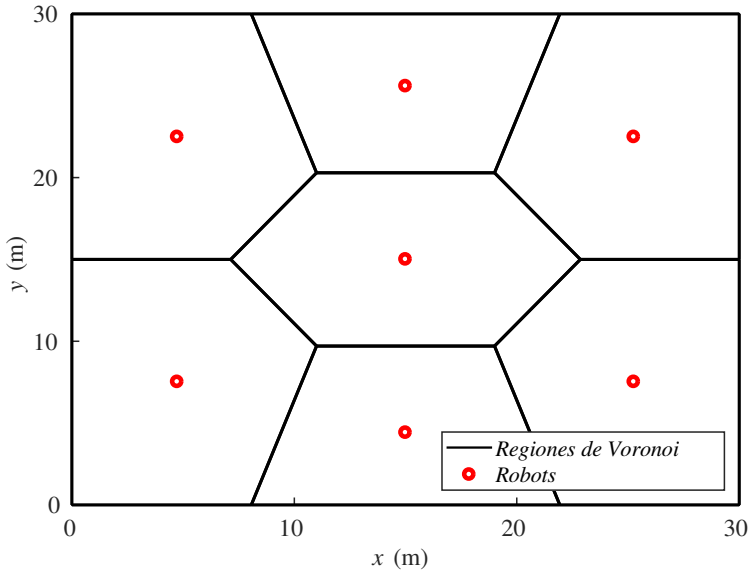


Figura 2.1: Divisiones de Voronoi de 7 robots sin funciones de densidad

Recordando algunas nociones básicas sobre las regiones, dada una región, V, y una función de densidad, ϕ, la masa, el centroide y el momento polar de inercia se calculan, respectivamente, como:

$$M_V = \int_V \phi(q)\, dq \tag{2.3}$$

$$C_V = \frac{1}{M_V} \int_V q\ \phi(q)\, dq \tag{2.4}$$

$$J_{V,p} = \int_V \|q - p\|^2\, \phi(q)\, dq \tag{2.5}$$

Considerando ahora la función de coste aplicada sobre la partición de Voronoi

$$\mathcal{H}_{\mathcal{V}}(P) = \sum_{i=1}^{n} \int_{V_i} \|q - p\|^2\, \phi(q)\, dq, \tag{2.6}$$

donde se sustituye la función f por $\|q - p\|^2$, y teniendo en cuenta el teorema de los ejes paralelos por el cual

$$J_{V,p} = J_{V,C_V} + M_V \|p - C_V\|^2, \tag{2.7}$$

se alcanza la siguiente expresión:

$$\mathcal{H}_{\mathcal{V}}(P) = \sum_{i=1}^{n} J_{V_i,C_{V_i}} + \sum_{i=1}^{n} M_{V_i} \|p_i - C_{V_i}\|^2 \tag{2.8}$$

De esta fórmula se deduce que los óptimos locales para la localización de los nodos en el área de trabajo son los centroides, C_{V_i}, de las divisiones de Voronoi desprendidas por las posiciones actuales

de los agentes. Por tanto, el proceso que ha de ejecutar iterativamente el algoritmo se recoge en la siguiente tabla:

Algoritmo de cobertura por divisiones de Voronoi	
1:	Obtener las posiciones actuales de los nodos.
2:	Dividir el espacio de trabajo por el método de Voronoi en base a las posiciones actuales.
3:	Calcular los centroides C_{V_i} de las regiones de Voronoi.
4:	Mandar mover los agentes a los centroides calculados.

En la Figura 2.2 se representa el mismo caso que en el ejemplo de la Figura 2.1, pero con tres focos de importancia colocados en posiciones aleatorias. Se observa que ahora la distribución no es uniforme, sino que los robots se aglomeran en los centros de las funciones de densidad.

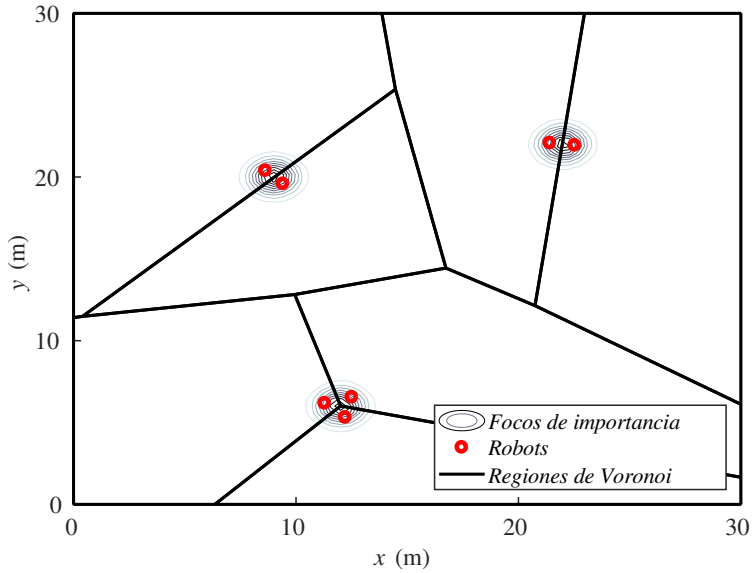


Figura 2.2: Divisiones de Voronoi de 7 robots con tres focos de importancia

En el caso en que la función de densidad, $\phi(q)$, sea constante, los centroides de las regiones de Voronoi coinciden con sus centroides geométricos. Si la región de Voronoi, V_i, es un polígono convexo con N_i vértices notados como $\{(x_0, y_0), ..., (x_{N_i-1}, y_{N_i-1})\}$, su centroide geométrico se calcula según las fórmulas:

$$\begin{aligned} C_{V_i,x} &= \frac{1}{6M_{V_i}} \sum_{k=0}^{N_i-1} (x_k + x_{k+1})(x_k y_{k+1} - x_{k+1} y_k) \\ C_{V_i,y} &= \frac{1}{6M_{V_i}} \sum_{k=0}^{N_i-1} (y_k + y_{k+1})(x_k y_{k+1} - x_{k+1} y_k), \end{aligned} \tag{2.9}$$

donde

$$M_{V_i} = \frac{1}{2} \sum_{k=0}^{N_i-1} (x_k y_{k+1} - x_{k+1} y_k). \tag{2.10}$$

Pero las regiones asociadas a cada agente no son sencillamente las regiones de Voronoi. Las capacidades de observación de los robots están limitadas a un determinado radio, s. Por tanto, la función de rendimiento de los sensores, $f(\|q - p_i\|)$, adopta los siguientes valores:

$$f(\|q - p_i\|) = \begin{cases} \|q - p_i\|^2, & si\ \|q - p_i\| \leq s \\ s^2, & en\ cualquier\ otro\ caso \end{cases}$$

De este modo, ahora las regiones de cobertura de cada robot serán una intersección entre los límites del área de trabajo, Q, las divisiones de Voronoi, V_i, y un círculo con centro en el agente y radio s, $C(p_i, s)$. La función de coste correspondiente es

$$\mathcal{H}(P) = \sum_{i=1}^{n} \int_{V_i \cap C(p_i,s)} f(\|q - p_i\|)\, \phi(q)\, dq \tag{2.11}$$

Las Ecuaciones 2.9 son igualmente aplicables a las regiones formadas por la intersección mencionada. Tan sólo hay que segmentar el círculo, $C(p_i, s)$, en pequeños tramos.

En la Figura 2.3 se representa el mismo caso anterior con tres funciones de importancia, pero con la configuración descrita ahora. El radio de percepción de los sensores es de 3 metros. Se observa que los robots van directos a los focos de importancia y cubren tan sólo las zonas a su alrededor. Esta desconexión entre los agentes es la que motiva la introducción de un algoritmo de mantenimiento de la conectividad.

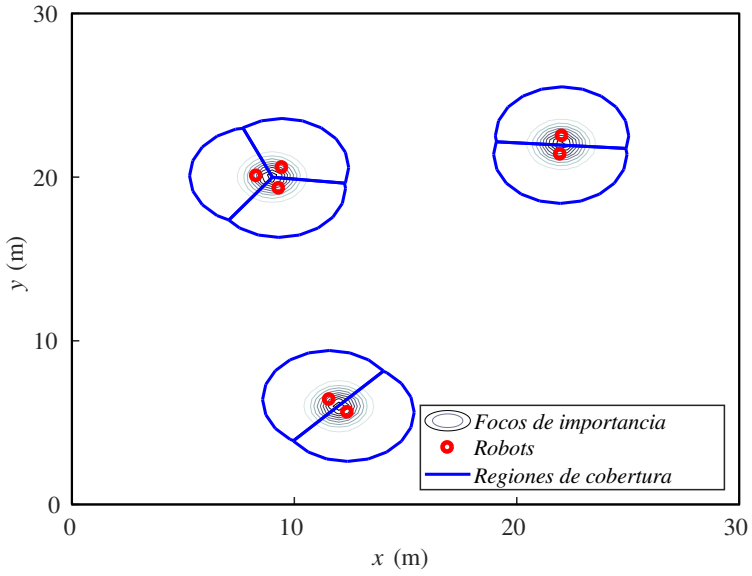


Figura 2.3: Regiones de cobertura de 7 robots con tres focos de importancia

2.2. Mantenimiento de la conectividad

En la introducción se ha destacado la importancia de mantener la conectividad entre los agentes de un sistema multi-robot. Éstos tienen una distancia máxima a la cual se pueden comunicar con otros dispositivos. Si el algoritmo sitúa un robot más allá de dicha distancia, quedará aislado y no podrá recibir órdenes o datos. El algoritmo de despliegue y cobertura que se presenta en este trabajo tiende a hacer que la flota se distribuya y se pierda la comunicación entre los nodos. Como más adelante se verá, esto ocurre en los casos en que la zona a cubrir es demasiado amplia o los objetivos a seguir se separan mucho. Por ello es necesario implementar un método para conservar la conectividad entre los agentes.

El método que se emplea en este trabajo es conocido en la literatura como Minimum Spanning Tree (MST). Consiste en construir un árbol de conexiones que abarque a todos los componentes del sistema minimizando los costes de comunicación. Para ello, si se dispone de un sistema con n nodos, el árbol tendrá tan sólo n-1 arcos (Fig. 2.4).

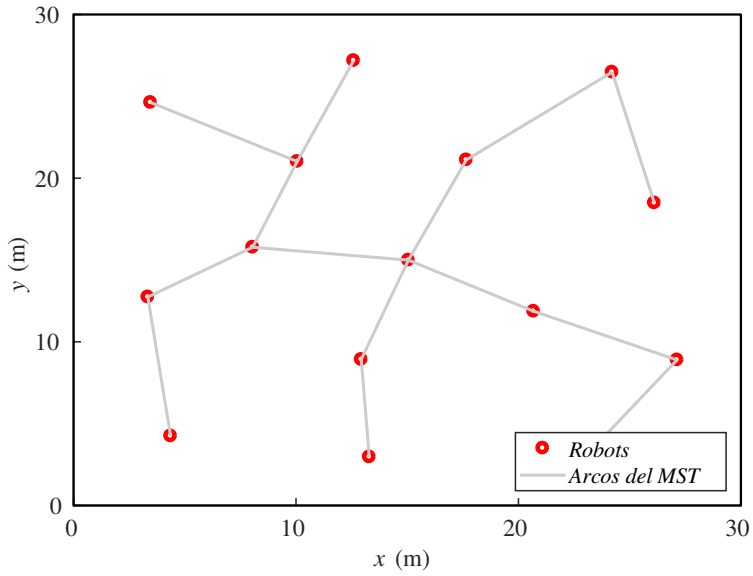


Figura 2.4: Árbol de comunicaciones MST de 15 robots en posiciones aleatorias.

Cada posible arco que une dos robots tiene asociado un peso. Los arcos elegidos para conformar el árbol de comunicaciones son aquéllos que minimizan el sumatorio de sus n-1 pesos. Si como peso se asigna la longitud del enlace, el árbol resultante tendrá la mínima distancia entre sus nodos. La razón para elegir esta variable como peso del arco no es otra que garantizar la mayor libertad de movimiento posible para los agentes, ya que esa es la principal limitación que impone el mantenimiento de la conectividad.

Siguiendo esta descripción, el proceso para construir el MST es muy sencillo. Comenzando por un nodo cualquiera, se compara la distancia a todos los vecinos que se encuentren dentro del radio de comunicaciones y se escoge el más cercano. Para ello, es necesario que en la configuración inicial ningún robot o grupo de robots esté aislado del resto. A continuación se repite el mismo proceso para el resto de nodos, obviando los que ya están incluidos en el árbol. El valor del radio de comunicaciones es un punto de controversia en la literatura [7]. En este caso, es igual al doble del radio de observación de los sensores de los robots, de modo que no quedan espacios intermedios sin cubrir.

El caso mostrado anteriormente en la Figura 2.1, al aplicar la restricción del mantenimiento de la conectividad, queda como se representa en la Figura 2.5. El radio de comunicaciones es de

6 metros y el de los sensores, de 3 metros. La expansión de los agentes es mucho menor que sin el MST, como es lógico. Todos los arcos están estirados al máximo, por lo que cada uno mide 6 metros.

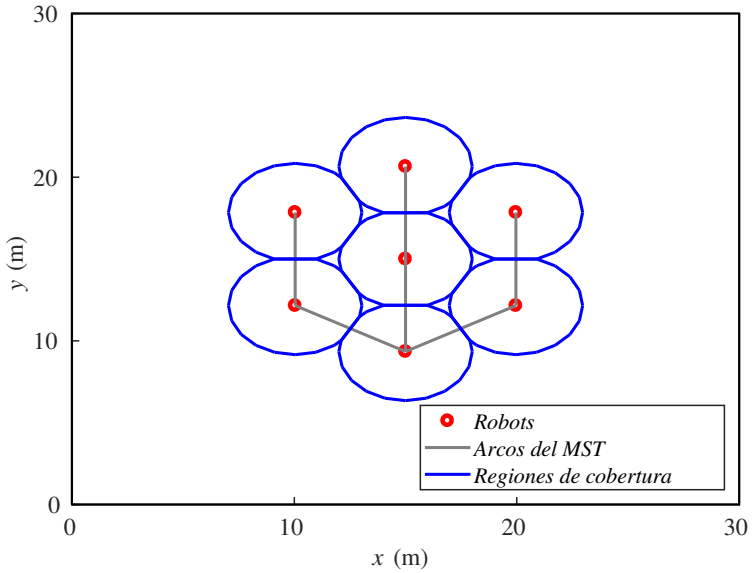
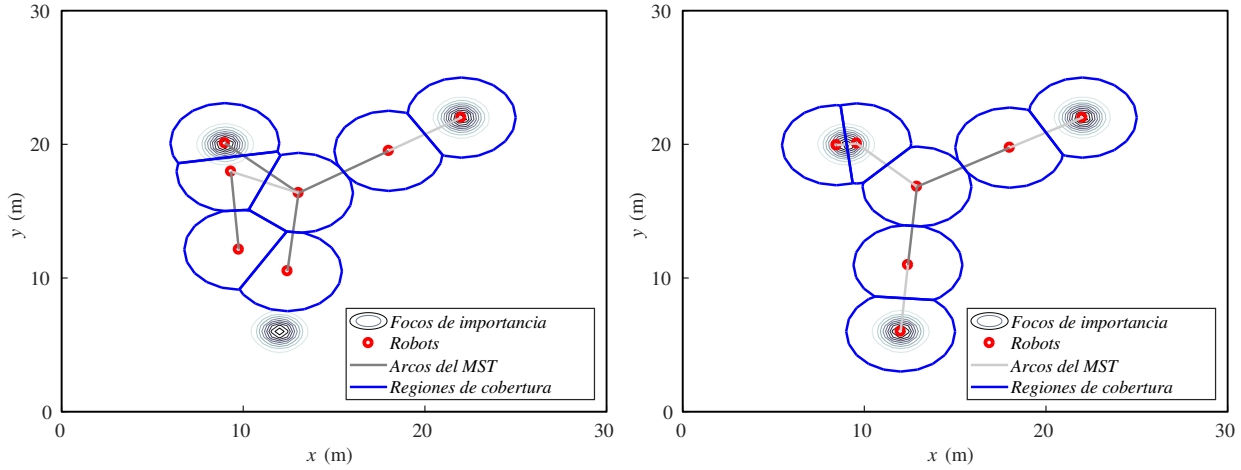


Figura 2.5: Despliegue de 7 robots con mantenimiento de la conectividad, sin funciones de importancia

El algoritmo de seguimiento hace que los agentes se vayan desplazando, y por tanto es necesario ir actualizando el árbol de comunicaciones. Si el árbol se mantiene fijo desde el inicio, es muy probable que no sea el idóneo en algún punto del movimiento de los robots, de modo que quedarían parcialmente bloqueados por la restricción impuesta. Pese a que existen métodos para detectar cuándo es necesario un cambio del árbol [8], en este trabajo se calculará el árbol óptimo en cada paso de simulación, ya que el cálculo del MST en el algoritmo centralizado supone un coste computacional bastante bajo. En la Figura 2.6 se muestra la diferencia entre un árbol fijo y un árbol variable para el caso de los tres focos de importancia. Los arcos del MST se representan en un color más oscuro cuando están cerca de su longitud máxima y comienzan a restringir el movimiento de los agentes.



(a) Árbol inicial fijo.

(b) Árbol variable en cada paso.

Figura 2.6: Despliegue de 7 robots con mantenimiento de la conectividad con tres focos de importancia

Se aprecia que en el caso del árbol variable los robots se distribuyen mucho mejor, de una manera más natural. En el caso del árbol fijo queda uno de los focos sin cubrir, por tanto no desempeña correctamente su función.

Concluyendo, el proceso que ejecuta el algoritmo cada vez que los robots se desplazan se recoge en la siguiente tabla:

Algoritmo de mantenimiento de la conectividad	
1:	Obtener las posiciones de los nodos.
2:	Calcular las distancias entre ellos.
3:	Para cada agente, descartar los que están fuera de su radio de comunicaciones.
4:	Comenzando por un robot, encontrar el más cercano a él y unirlos.
5:	Continuar uniendo los robots más cercanos que no se encuentren ya en el árbol.
6:	Una vez que se han calculado los centroides de las regiones, limitar el movimiento para que ningún agente quede fuera del radio de comunicaciones de algún otro.

Capítulo 3

Propuestas para la labor de seguimiento

Una vez conocido el funcionamiento básico del algoritmo de cobertura con mantenimiento de la conectividad, en este capítulo se procede a explicar la extensión del mismo para realizar labores de seguimiento.

Uno de los objetivos de este Trabajo Fin de Máster consiste en desarrollar un algoritmo para que un conjunto de robots terrestres desempeñen una labor de apoyo a una flota aérea. Para ello, la cuestión que se plantea es cómo conseguir que la sigan para poder realizar dichas tareas. A continuación se presentan dos alternativas, una adjudicando funciones de importancia a los objetivos, y otra modificando los límites del área de trabajo en función de la posición de los objetivos.

3.1. Aplicación de funciones de importancia

Esta primera alternativa consiste en situar funciones de importancia con centro en los objetivos para que, como se ha visto en el capítulo anterior, los robots tiendan a desplazarse hacia ellos.

Las funciones de densidad aplicadas tienen la forma:

$$\phi\left(q\right) = e^{-\|q-o\|^2}, \tag{3.1}$$

donde q es el punto en consideración y o es el centro de la función de importancia, situado en el objetivo correspondiente. En la Tabla 3.1 se muestran algunos valores de ϕ para diferentes distancias entre un punto cualquiera y el centro de una función como la descrita.

Tabla 3.1: Valores de la función de importancia para algunas distancias.

$\|q-o\|$ (m)	0.1	0.5	1	2	3	5
$\phi(q)$	0.9900	0.7788	0.3679	0.0183	0.0001	$1{\cdot}10^{-11}$

Se observa que si el punto está muy cerca el valor tiende a 1, y conforme se va alejando decrece rápidamente. A los dos metros el valor ya es unas 50 veces menor, por lo que los puntos a partir de esa distancia apenas tienen influencia.

Al ponderar el cálculo de los centroides de las regiones de cobertura de cada robot con las funciones de importancia (Ec. 2.4), quedan desplazados hacia el centro de la función, es decir, hacia el objetivo. Si sólo existe una función, los agentes se juntarán en torno a su centro. En caso de que se

implanten más funciones, los agentes que se encuentren en la zona más cercana a cada centro se acercarán, cumpliendo siempre con las restricciones impuestas por el mantenimiento de la conectividad.

En la Figura 3.1 se representan diferentes instantes de tiempo de una simulación realizada con la alternativa propuesta. Un grupo de 5 robots se despliega para cubrir tres objetivos que no se mueven. Se contempla como los agentes van avanzando hacia los objetivos hasta cubrirlos todos.

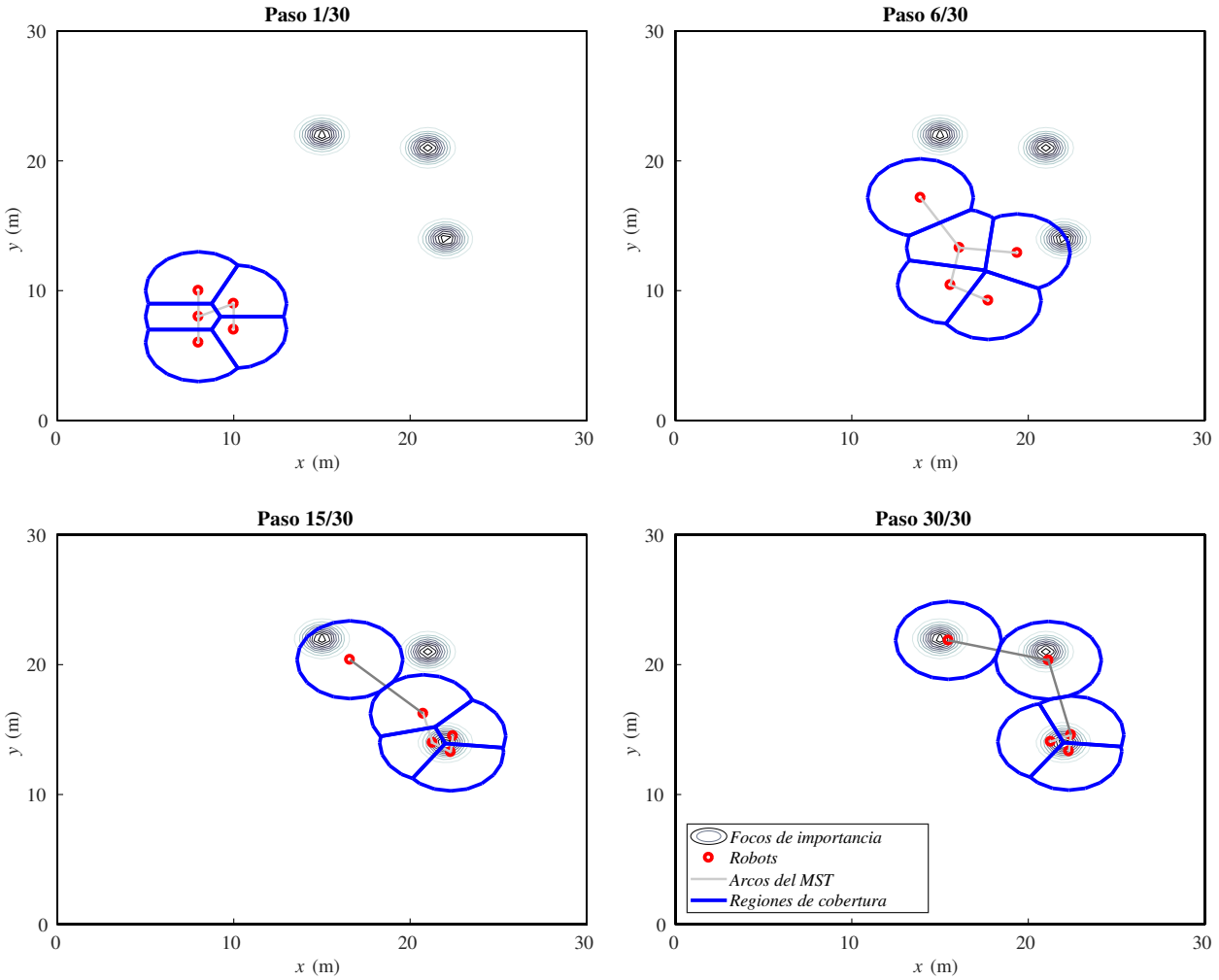


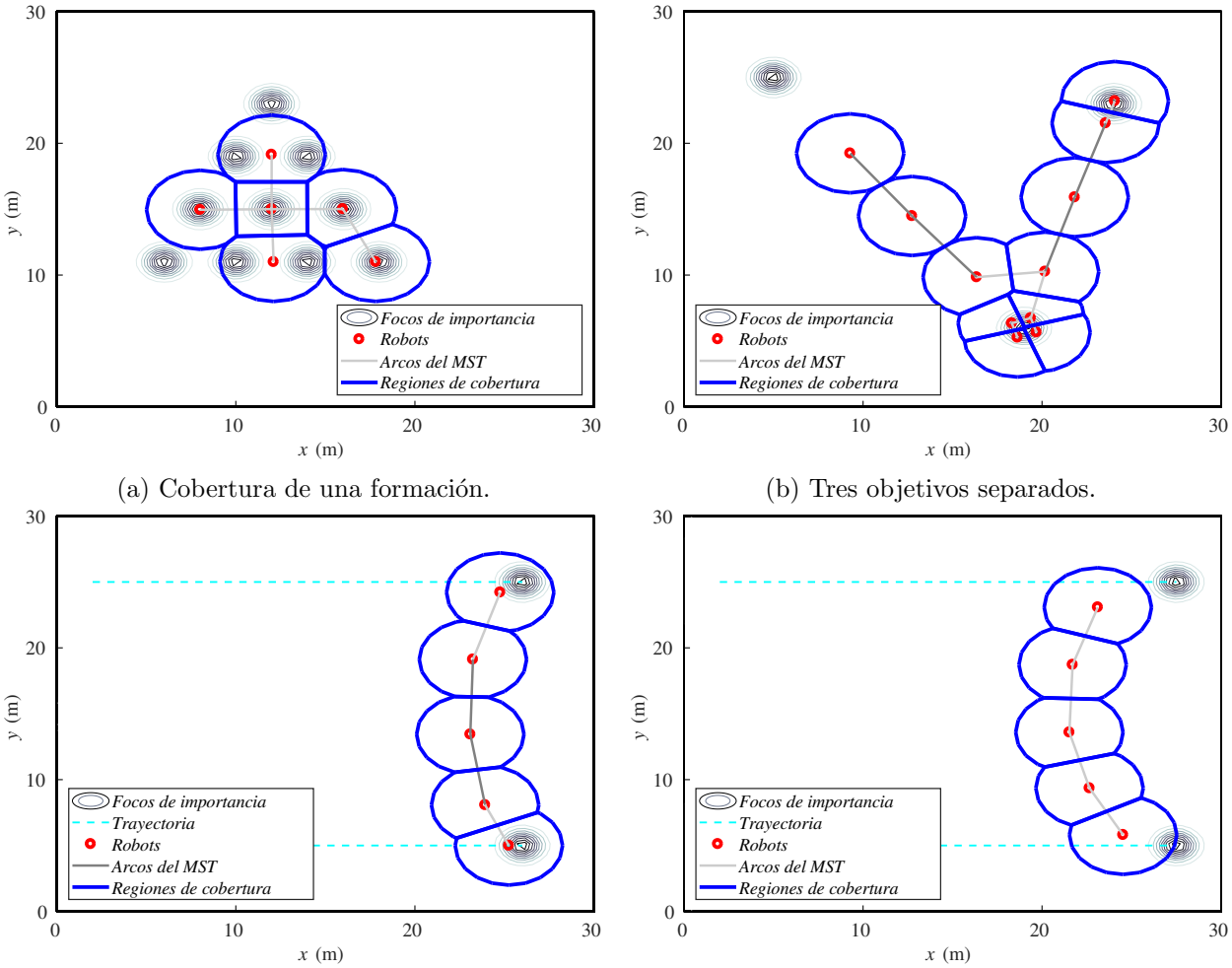
Figura 3.1: Instantáneas de las posiciones de los robots a lo largo de la simulación

La ventaja de esta alternativa es que los robots se dirigen directa y rápidamente hacia los objetivos simplemente con el cálculo de los centroides. Como inconveniente destaca la fuerte dependencia con la posición inicial de los agentes con respecto a las zonas de importancia, que puede hacer que alguno de los objetivos quede sin cubrir por el hecho de que inicialmente estaba más lejos que los demás.

Con el objetivo de ilustrar el comportamiento del algoritmo, en la Figura 3.2 se muestran algunos ejemplos de sistemas a cubrir. En todos los casos, los robots parten aproximadamente desde el centro de la zona mostrada en el gráfico.

Antes de empezar con los dos primeros casos, hay que destacar que se han simulado suficientes pasos como para que la solución haya alcanzado el comportamiento en régimen permanente. El primer caso (Fig. 3.2a) consiste en realizar la cobertura de una flota que está en una formación

inmóvil. Se aprecia que algunos objetivos quedan sin cubrir, los más alejados de la posición inicial de los agentes. En el segundo caso (Fig. 3.2b) se lleva el algoritmo al límite colocando tres objetivos muy distanciados. Los agentes se aglomeran en torno a los objetivos, quedando uno de ellos sin cubrir. Esto se debe a la posición inicial de los nodos, ya que si alguno de los cuatro que se sitúan en el objetivo inferior se colocase en el extremo izquierdo, los tres objetivos quedarían cubiertos.



(a) Cobertura de una formación.

(b) Tres objetivos separados.

(c) Seguimiento de dos objetivos a velocidad baja.

(d) Seguimiento de dos objetivos a velocidad alta.

Figura 3.2: Diferentes ejemplos de aplicación del algoritmo con zonas de importancia.

Por último, el tercer y cuarto caso representan el último paso de simulación de una serie en la que los objetivos se han ido moviendo según la trayectoria horizontal mostrada. En la Figura 3.2c los objetivos llevan una velocidad de 0,75 m/paso, y el sistema es capaz de seguirlos. En la Figura 3.2d la velocidad es de 1,5 m/paso, y los agentes ya no alcanzan a seguirlos.

En el siguiente apartado se comparará el comportamiento de los agentes con la otra alternativa propuesta, empleando siempre el mismo número de robots y partiendo desde la misma posición inicial.

3.2. Modificación de los límites de la zona

Esta otra alternativa consiste en modificar dinámicamente los límites de la zona de trabajo en función de la posición de los objetivos móviles.

A la hora de modificar la zona de trabajo, se ha escogido envolver los objetivos con el rectángulo más pequeño posible. El rectángulo es la forma más sencilla de implementar en el algoritmo, ya que solamente es necesario conocer el máximo y el mínimo de las coordenadas de los objetivos. Para que el algoritmo funcione correctamente, los agentes también tendrán que inscribirse con el rectángulo. Por tanto, sus coordenadas también han de incluirse en el cálculo del máximo y el mínimo.

Al modificar la región como se ha descrito, y al no existir funciones de importancia, los robots se distribuyen uniformemente alrededor de los objetivos. De esta manera, la cobertura que ofrece este método se centra más en el entorno de los objetivos, y no tanto en los objetivos en sí.

Al igual que en el apartado anterior, en la Figura 3.3 se muestran diferentes instantes de tiempo (los mismos que antes) de una simulación realizada con esta otra alternativa. Se comprueba que el avance de los agentes hacia los objetivos es más lento, pero la cobertura de la zona de interés es más amplia.

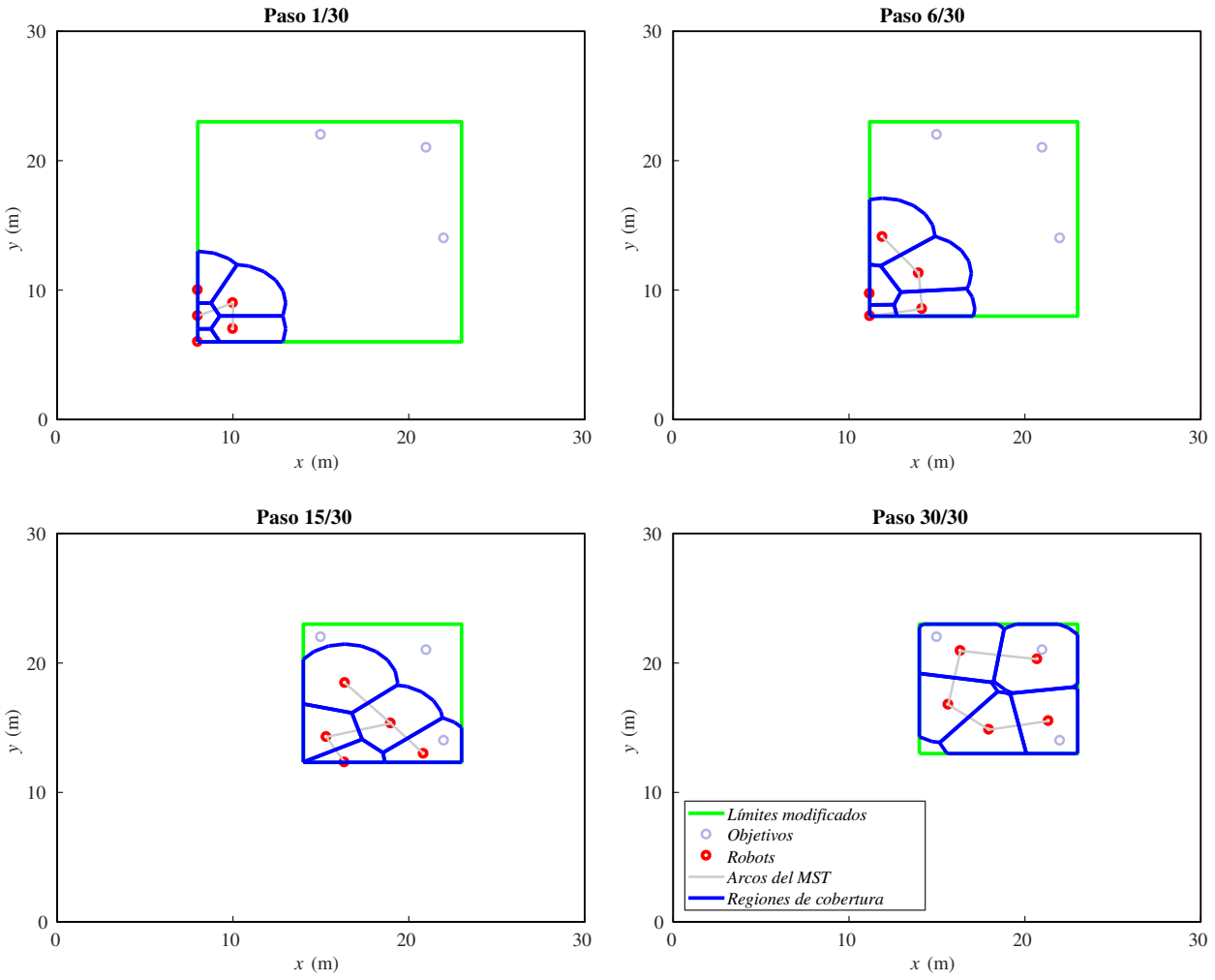
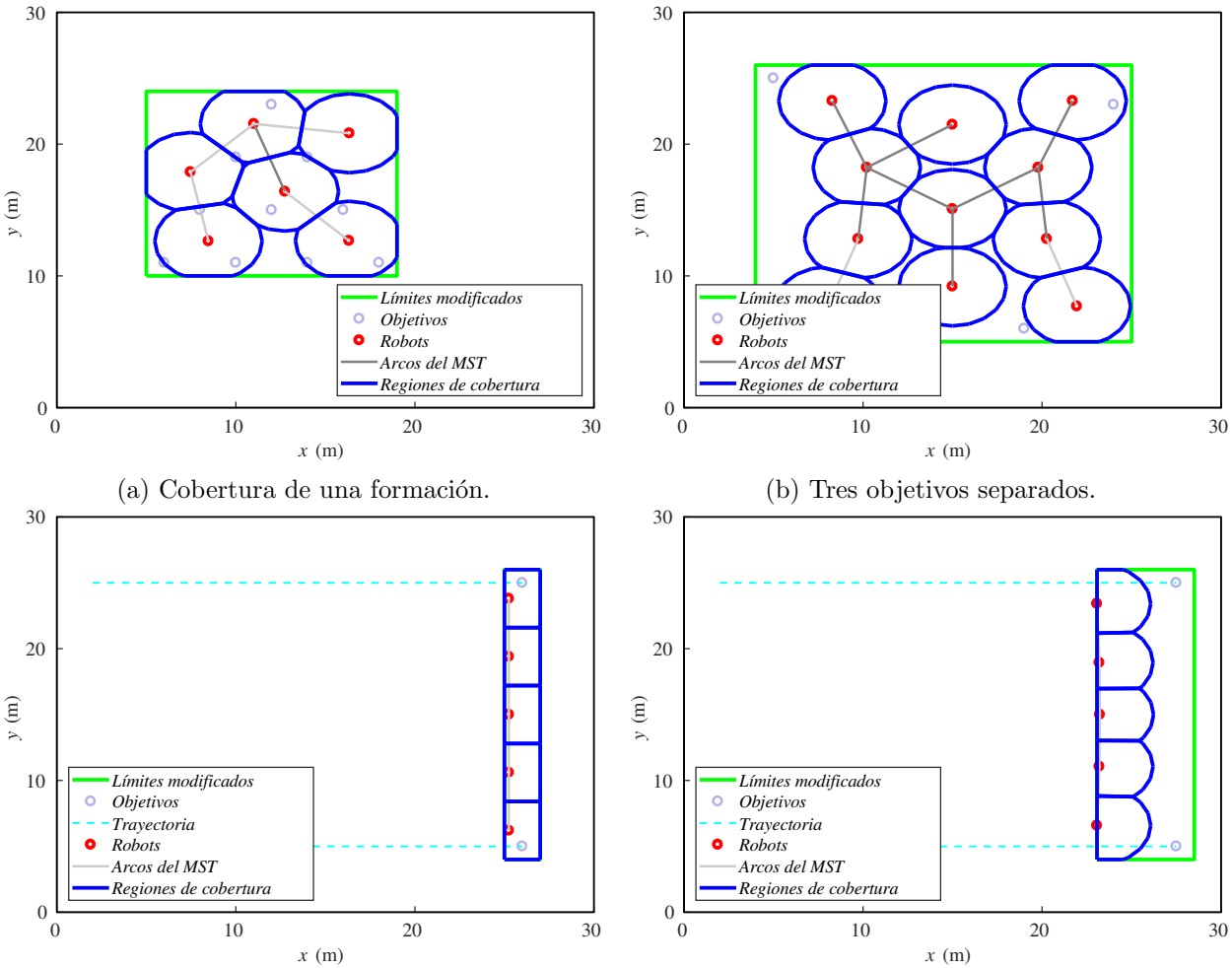


Figura 3.3: Instantáneas de las posiciones de los robots a lo largo de la simulación

Las ventajas que presenta esta alternativa son la uniformidad que ofrece en la zona de los objetivos y que ninguno de ellos queda fuera, a no ser que la zona sea demasiado grande para el número de agentes. También tiene varios inconvenientes. Al definir un rectángulo como zona de trabajo, es probable que se le otorgue la misma importancia a una zona vacía y relativamente alejada de los puntos de interés que a esos propios puntos. Otro caso en el que desempeña un mal papel es cuando los objetivos se mueven demasiado deprisa. Si los agentes no son capaces de distribuirse con suficiente celeridad, el método pierde la ventaja que tenía con respecto a la alternativa anterior.

En la Figura 3.4 se muestran los mismos ejemplos que en el apartado anterior, aplicando en este caso la alternativa de modificar los límites del área de trabajo.



(a) Cobertura de una formación.

(b) Tres objetivos separados.

(c) Seguimiento de dos objetivos a velocidad baja.

(d) Seguimiento de dos objetivos a velocidad alta.

Figura 3.4: Diferentes ejemplos de aplicación del algoritmo con modificación de los límites.

En la Figura 3.4a, la cobertura conseguida es mejor que con el método de las zonas de importancia. Todos los objetivos se encuentran dentro del rango de los sensores. Por otra parte, en la Figura 3.4b los agentes se distribuyen uniformemente y, al no haber un número suficiente para la zona resultante, los objetivos quedan mal cubiertos. El algoritmo presenta un mal comportamiento para este caso con las dos alternativas, ya que es un ejemplo muy extremo.

Pasando a los objetivos móviles, en la Figura 3.4c la cobertura conseguida es un poco mejor que con el método anterior. Los agentes avanzan en línea recta y los extremos son equidistantes a los dos objetivos. En la Figura 3.4d, sin embargo, el comportamiento es igual que con las zonas de importancia, ya que los agentes no son capaces de seguir a los objetivos debido a su velocidad.

Con todo ello, viendo los resultados obtenidos en estos ejemplos con las dos alternativas, se desprenden las conclusiones que se exponen a continuación. Cabe remarcar de nuevo que estos cuatro ejemplos son para contemplar el comportamiento del algoritmo en diferentes casos muy concretos y realizar una comparación entre las alternativas. En el Capítulo 5 se llevarán a cabo nuevas simulaciones con diferentes motivaciones.

En primer lugar, el método de las zonas de importancia es bueno para objetivos sueltos pero falla a la hora de cubrir formaciones y objetivos muy distantes a diferentes distancias de la posición inicial. En segundo lugar, el método de los límites variantes es idóneo para la cobertura de formaciones, pero si los objetivos están muy alejados hay demasiados agentes que quedan en un terreno sin interés. Por último, se demuestra la existencia de una velocidad límite de los objetivos para que los agentes puedan seguirlos. Ambos métodos fallan cuando se supera cierta velocidad.

Capítulo 4

Implementación sobre una plataforma de simulación

Otro de los objetivos de este Trabajo Fin de Máster consiste en implementar el algoritmo de cobertura y seguimiento sobre una plataforma de simulación realista, en la que se encuentren problemas muy similares a los que aparecerían en una implementación real.

En este capítulo se realiza una pequeña introducción al software empleado, se describe el proceso de implementación sobre un robot, y su extensión a varios robots.

4.1. Presentación del software

Los programas que se van a emplear tienen diferentes funcionalidades. En primer lugar está Gazebo, que es la plataforma que simula con su motor físico el movimiento de los robots. También es necesario un sistema operativo para controlar los robots. En este caso se emplea ROS (Robot Operating System), que dispone de paquetes específicos para el robot elegido. Por último, el programa MATLAB (abreviatura de MATrix LABoratory) sirve de base para ejecutar el algoritmo. Las versiones empleadas en este trabajo son Gazebo 7.1 y ROS Kinetic en Ubuntu 16.04.

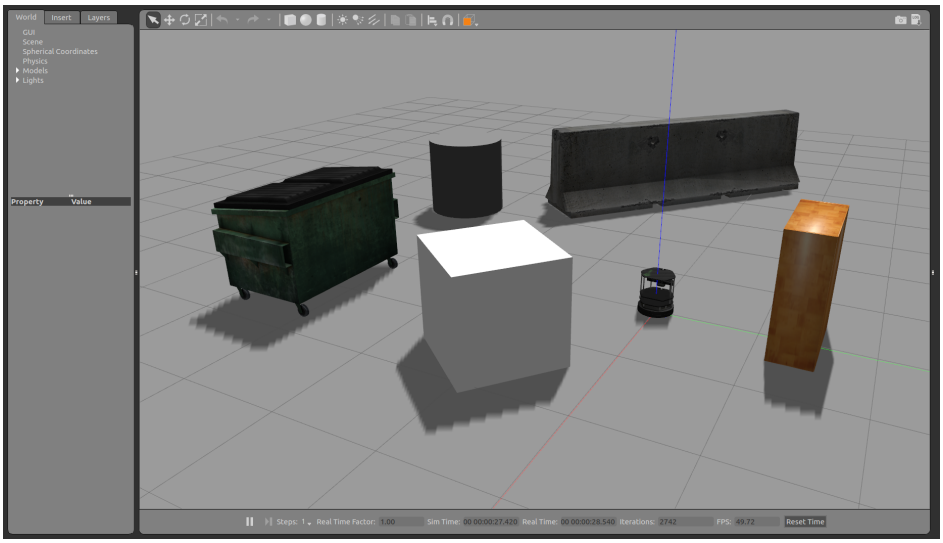


Figura 4.1: Escenario de simulación en Gazebo con un Turtlebot y varios objetos.

Se ha escogido este software debido a que la interacción entre los tres programas está bastante desarrollada. Existen métodos para que MATLAB se comunique tanto con ROS como con Gazebo. También la integración de Gazebo en el sistema de ROS es cada vez mayor, ya que, en las fases de

investigación previas a una implementación real, es conveniente probar los algorimos en un entorno de simulación controlado. A continuación se describe brevemente cada uno de los programas.

Gazebo es un simulador de entornos 3D especializado en robótica (Fig. 4.1). Permite diseñar robots personalizados, aunque en este trabajo se empleará el Turtlebot, ampliamente utilizado en ámbitos de investigación. También permite crear mundos virtuales o importar modelos de otras plataformas CAD.

Para este trabajo, Gazebo tendrá dos funciones principales. Como plataforma de simulación, soportará la ejecución de los algoritmos lanzados desde MATLAB, y como elemento principal de visualización, incluirá elementos visuales para una comprensión más inmediata del comportamiento de los robots.

ROS, por su parte, es un marco para el desarrollo de software para robots. Se trata de un software libre compuesto por dos partes, el sistema operativo en sí, y una serie de paquetes aportados por los usuarios. Es por ello que constituye un foco de desarrollo para investigadores de todo el mundo.

Su funcionamiento se basa en una arquitectura de grafos, en la que los nodos son los encargados del procesamiento. Existen dos métodos de comunicación entre los nodos (Fig. 4.2). Uno consiste en que un nodo publica continuamente mensajes en un asunto, al cual otro nodo se suscribe para obtener la información. El otro consiste en que un nodo solicita un servicio en un momento determinado a otro nodo. Enlazando estos mensajes de diversa índole, el sistema acaba cumpliendo con su funcionalidad.

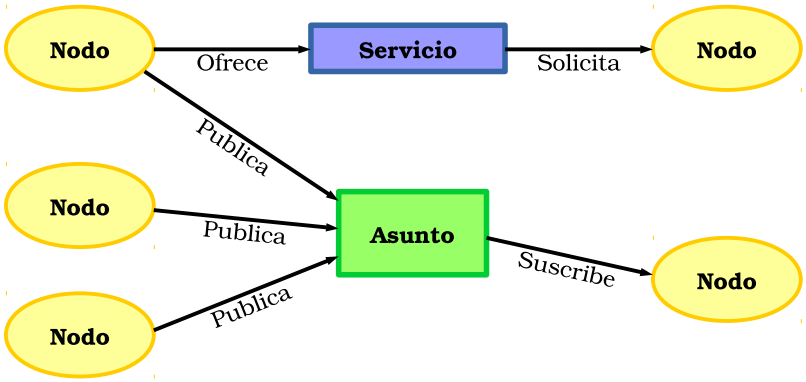


Figura 4.2: Esquema de las diferentes formas de comunicación entre los nodos en ROS.

En lo que respecta a este trabajo, se emplearán todos los paquetes necesarios para la comunicación con Gazebo y MATLAB, y todos aquéllos que sirvan para controlar el Turtlebot en el entorno de trabajo planteado. Entre estos últimos destacan el controlador de la cámara acoplada en el robot, el sistema de navegación o el sistema de localización.

Por último, MATLAB es una herramienta de software matemático empleado en casi todos los ámbitos de la ingeniería. Cuenta con un lenguaje de programación propio y es capaz de representar datos, implementar algoritmos y comunicarse con otros programas, entre otras muchas funcionalidades.

Son precisamente estas tres funcionalidades las que serán de utilidad en este trabajo. Las figuras de los Capítulos 2 y 3 están realizadas con el sistema de representación de MATLAB, que permite visualizar y diferenciar los distintos elementos. Y, como ya se ha mencionado, el algoritmo se implementa en este programa y durante las simulaciones se comunicará con Gazebo y ROS gracias a sus funcionalidades.

4.2. Implementación de un robot

Las simulaciones que se van a realizar imitan la situación en la que un grupo de robots terrestres realizan tareas de apoyo a una flota aérea. Se ha seleccionado este caso puesto que los robots que ejecutan el algoritmo son los terrestres, cuyo control con ROS es más sencillo. En las simulaciones se incluirán cuadrotores pero sólo como elementos visuales auxiliares. Los agentes terrestres que se van a emplear son los Turtlebots (Fig. 4.3), unos robots con dos ruedas móviles, de cinemática diferencial, con una cámara Kinect integrada.

(a) En pie. (b) Detalle de las ruedas.

Figura 4.3: Turtlebot generado en Gazebo.

De este modo, el problema planteado en este apartado es implementar un Turtlebot en Gazebo con las funcionalidades necesarias para este trabajo, gracias a ROS y MATLAB. Los nodos se describen en lenguaje XML y se lanzan a Gazebo a través de ROS mediante ficheros con extensión *.launch*. Uno de estos ficheros básicos, incluidos en los tutoriales de Gazebo-ROS [9], consta principalmente de una descripción de la estructura y la representación gráfica del robot, un nodo que transforma las imágenes de la cámara a medidas de un láser, un controlador para las ruedas y un nodo que publica constantemente las transformaciones entre los diferentes sistemas de referencia.

Partiendo de estos ficheros, las funcionalidades que se necesitan implementar para cumplir con los objetivos de este trabajo son un sistema de localización, para conocer la posición del Turtlebot, y un sistema de navegación, que le permita desplazarse a los puntos deseados evitando los obstáculos que se interpongan.

En algunos sistemas reales el sistema de localización se implementa mediante un nodo que ejecuta un AMCL (Adaptive Monte Carlo Localization), generalmente en aquéllos privados de localización por GPS. Este método otorga una aproximación probabilística a la localización del robot. Un filtro de partículas estima la posición del móvil gracias a la comparación entre las medidas del

láser y un mapa conocido del entorno. Sin embargo, el escenario que se va a simular presenta varios inconvenientes para implementar este método. En primer lugar, el escenario estará libre de obstáculos fijos, tan sólo entran en escena los Turtlebots. Esto impide que se puedan tomar referencias fijas para ejecutar el AMCL con un mapa conocido de los obstáculos. Además, se trata de un entorno altamente dinámico, donde los robots se mueven continuamente y crean oclusiones que impiden observar el entorno, lo cual dificultaría la localización en caso de que se considerase integrar algunos obstáculos fijos.

En las librerías de ROS no se ha encontrado ningún sistema de localización que pueda funcionar para este caso particular. Y desarrollar un método específico queda fuera del alcance de este Trabajo Fin de Máster. Es por ello que se recurre a implementar lo que se conoce como un sistema de "falsa localización". En Gazebo se puede conocer perfectamente la posición del robot, por lo que la labor de localización se puede completar recogiendo este dato para entregárselo al algoritmo (Fig. 4.4).

Figura 4.4: Esquema de los nodos del sistema de localización.

El sistema de navegación es el aspecto más complejo de la estructura funcional necesaria para este trabajo. ROS ofrece un paquete de navegación en dos dimensiones mediante el cual, un nodo principal recibe información de odometría, sensores y posición objetivo, y devuelve comandos de velocidad para el controlador de las ruedas del robot. Requiere también de un mapa en el que se distingan los obstáculos que existen en el entorno. Siempre que haya elementos con distintos sistemas de referencia (robot, cámara, mapa, etc) también se hacen necesarias las transformaciones entre ellos.

Por tanto, el grafo de este sistema presenta una estructura como la mostrada en la Figura 4.5. El nodo principal, a partir de los obstáculos reflejados en el mapa, las mediciones de los sensores y la localización del robot es capaz de generar una trayectoria hasta el objetivo fijado. Lo hace estableciendo un mapa de coste bidimensional sobre el suelo, calculando la trayectoria que menor coste implica.

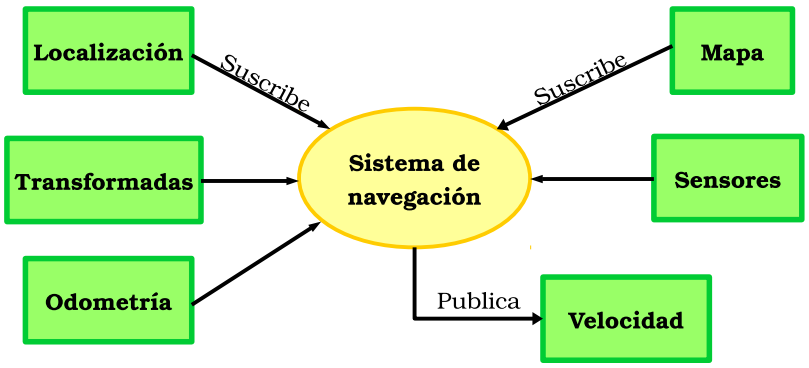


Figura 4.5: Esquema de los nodos del sistema de navegación.

Entrando en detalle, el nodo principal dispone de cinco elementos. El primero es un mapa de coste global, generado a partir del mapa de obstáculos y las medidas de la cámara que se le aportan. El segundo es un planeador de trayectorias global, que establece un camino a seguir a partir del mapa de coste generado anteriormente. También existe un mapa de coste local que tan sólo se fija en los datos que va otorgando la cámara. Con este último mapa y empleando como guía la trayectoria generada por el planeador global, un planeador local, que tiene acceso a las medidas de odometría, produce los comandos de velocidad que se envían al controlador de las ruedas del robot. El quinto elemento de este nodo se añade para establecer comportamientos de recuperación del robot en caso de que se detecte que puede estar atascado.

En el escenario que se va a simular no existen obstáculos, más allá de los propios robots. Por lo tanto, el mapa que se aporta al sistema de navegación, que incluye los obstáculos fijos conocidos, será una imagen blanca. Los robots se evitarán entre sí gracias al mapa de coste local, que irá generando obstáculos cuando el láser detecte la presencia de otro agente.

4.3. Extensión a varios robots

A la hora de extender la implementación a varios robots, Gazebo y ROS disponen de una utilidad muy apropiada. Todos los nodos, mensajes, servicios, sistemas de referencia, etc., se pueden englobar en un mismo grupo, al que se adjudica un espacio de nombres. Estos espacios se caracterizan mediante un prefijo, que se coloca delante de todos los objetos mencionados.

Así pues, creando un espacio por cada robot se consigue replicar la estructura descrita en el apartado anterior el número de veces que se desee. Los únicos elementos que no se replican son el nodo de Gazebo, ya que sólo hay un núcleo de simulación, y el nodo que contiene el mapa del sistema de navegación, para que el mapa conforme un sistema de referencia fijo para todos los robots.

Cuando entran varios robots en escena, se hace necesaria alguna ayuda para comprender el movimiento que está haciendo cada uno de los robots en cada instante. Con dicho fin, se ha diseñado un sencillo sistema de visualización. Mediante comandos en el algoritmo de MATLAB, se mandan aparecer y desaparecer objetos que representan diferentes aspectos del sistema.

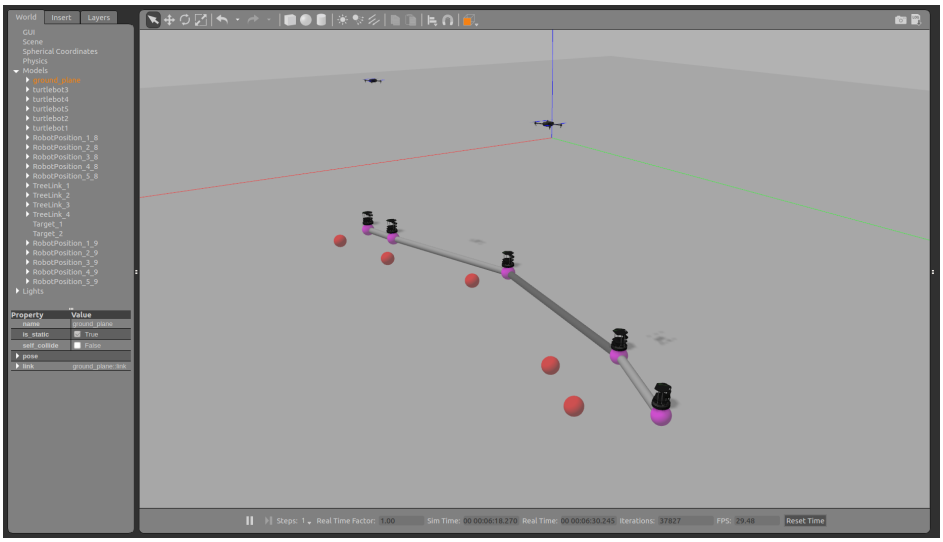


Figura 4.6: Escenario de simulación en Gazebo con 5 Turtlebots y el sistema de visualización.

En la Figura 4.6 se incluye un escenario de simulación de Gazebo con varios Turtlebots ejecutando el algoritmo de seguimiento. Se representan tanto las posiciones actuales de los robots como las posiciones objetivo que se desprenden del algoritmo mediante esferas de colores situadas justo bajo el suelo, al cual se ha aplicado cierto grado de transparencia. Los arcos del árbol de comunicaciones se representan mediante cilindros horizontales que unen las posiciones de los robots. Por último, los objetivos, en este caso aéreos, se representan mediante cuadrotores situados a una cierta altura. Hay que destacar que los cuadrotores no se mueven de manera continua, sino que se cambia su posición de forma discreta.

Capítulo 5

Simulaciones y resultados

Este capítulo tiene dos cometidos, explicar las simulaciones realizadas y analizar los resultados obtenidos. Unas simulaciones irán simplemente destinadas a observar el comportamiento del algoritmo en algunos casos concretos, y otras irán destinadas a analizar los cambios producidos al modificar ciertas variables del problema.

5.1. Simulaciones con Gazebo

En primer lugar, se va a simular cuatro ejemplos en los que se pone a prueba el funcionamiento del algoritmo de cobertura desarrollado en el presente trabajo. Para observar el proceso con mayor detalle, se implementan sobre la plataforma de Gazebo-ROS.

Los casos estudiados son:

1. Primero un despliegue sencillo de una flota de siete robots sin objetivos, de forma que se expanden hasta el límite de sus radios de comunicaciones.
2. Un conjunto de cinco robots que sigue a una flota aérea formada por diez cuadrotores que vuelan en una formación triangular.
3. Un grupo de cinco robots que realiza la cobertura de dos cuadrotores que siguen trayectorias curvilíneas que se cruzan en un punto.
4. Por último, una flota de seis robots que se estira para albergar a dos robots aéreos que se separan una gran distancia.

Estos cuatro casos se proponen con objeto de enseñar el comportamiento del algoritmo en situaciones comunes (expansión sencilla y seguimiento de una formación) y situaciones complicadas (trayectorias cruzadas y separación de los objetivos) que ponen a prueba su validez.

Los vídeos que muestran los casos planteados se recogen en la carpeta TARDOS_IBARRA_JAVIER_647077_COMPLEMENTARIO_TFM de los archivos complementarios entregados con este trabajo. Cada vídeo lleva como nombre "*caso_*" y el número del caso correspondiente según el listado anterior. A continuación se describe el contenido de estos vídeos y se explican los comportamientos observados en ellos.

En el caso 1 se representan tres ventanas. Una corresponde a la visualización de Gazebo (parte derecha de la pantalla) explicada en el Capítulo 4, otra a las gráficas generadas con MATLAB (parte inferior izquierda), como las mostradas en el Capítulo 3, y la última es una herramienta de visualización de ROS (parte superior izquierda), en la que se observan otros aspectos de la

simulación como el mapa de coste local de cada robot. La primera iteración se muestra a velocidad real y el resto se aceleran a velocidad x8. En este caso se observa como la flota se va expandiendo, inicialmente con movimientos más extensos y luego más cortos. El sistema de navegación tiene una tolerancia de llegada al objetivo de algunas decenas de centímetros para que el movimiento sea más fluido. Esto hace que cuando los robots se han de desplazar una distancia inferior a dicha tolerancia, consideran que no se tienen que mover. Por tanto, los agentes dejan de extenderse cuando los arcos del árbol de comunicaciones aún no han llegado a su límite de longitud. La disposición final de los agentes se muestra en la Figura 5.1.

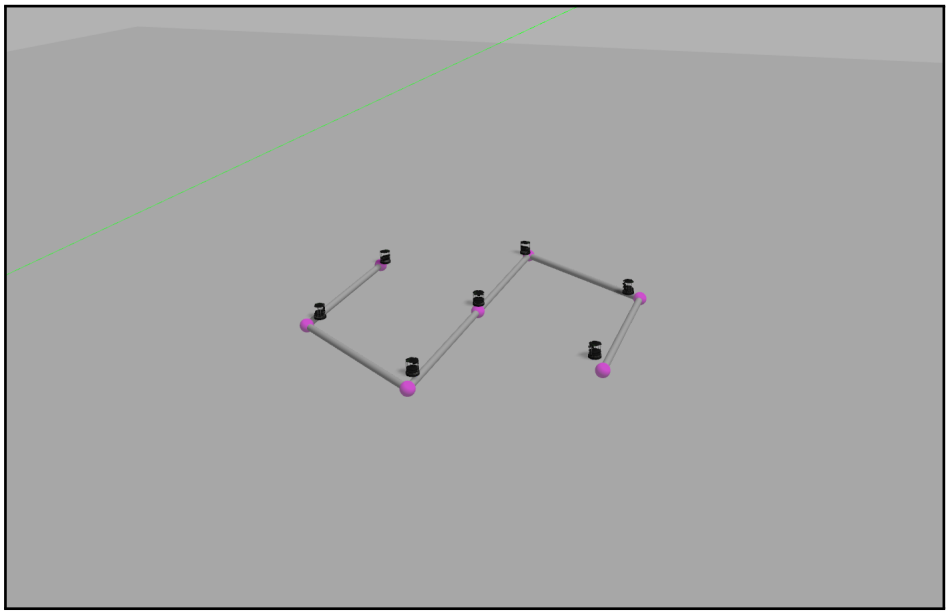


Figura 5.1: Disposición final de los robots en Gazebo en el caso 1.

Para el resto de simulaciones, que son algo más complejas, tan sólo se representan dos ventanas. El visualizador de ROS se quita porque carga demasiado la ejecución, haciéndola muy lenta. En el caso 2 se quiere observar el comportamiento del método de modificación de los límites a la hora de seguir una formación. De nuevo la primera iteración se muestra a velocidad normal y el resto x8.

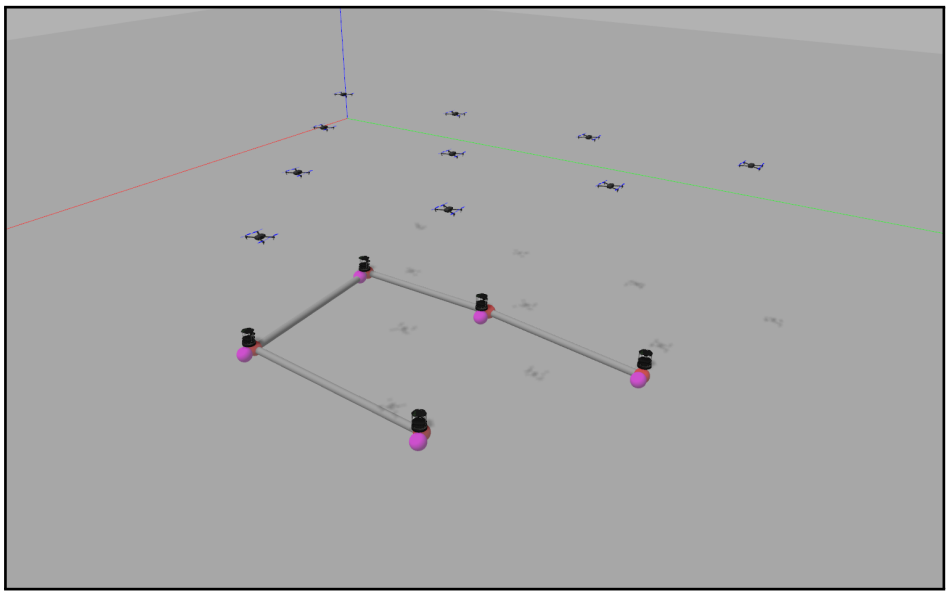


Figura 5.2: Disposición final de los robots en Gazebo en el caso 2.

Se puede ver como los cinco robots se van desplegando para cubrir la formación de cuadrotores y acompañarlos en su avance. En la última iteración (Figura 5.2) todos los robots de la flota aérea quedan cubiertos por alguno de los agentes terrestres.

Para los casos 3 y 4 se emplea el método de las funciones de importancia, ya que se trata de objetivos sueltos. Son simulaciones diseñadas para poner a prueba el algoritmo y comprobar hasta qué punto es fiable. En el caso 3, tras la primera iteración, el resto se muestran a velocidad x12. Los robots inicialmente se despliegan para albergar los dos objetivos, que se van juntando. Tras el cruce de las trayectorias, el redespliegue no es el más adecuado y uno de los objetivos queda finalmente sin cubrir (Figura 5.3). Cabe destacar que si los objetivos llevaran una velocidad más baja, la flota sería capaz de seguirlos, extendiéndose mejor.

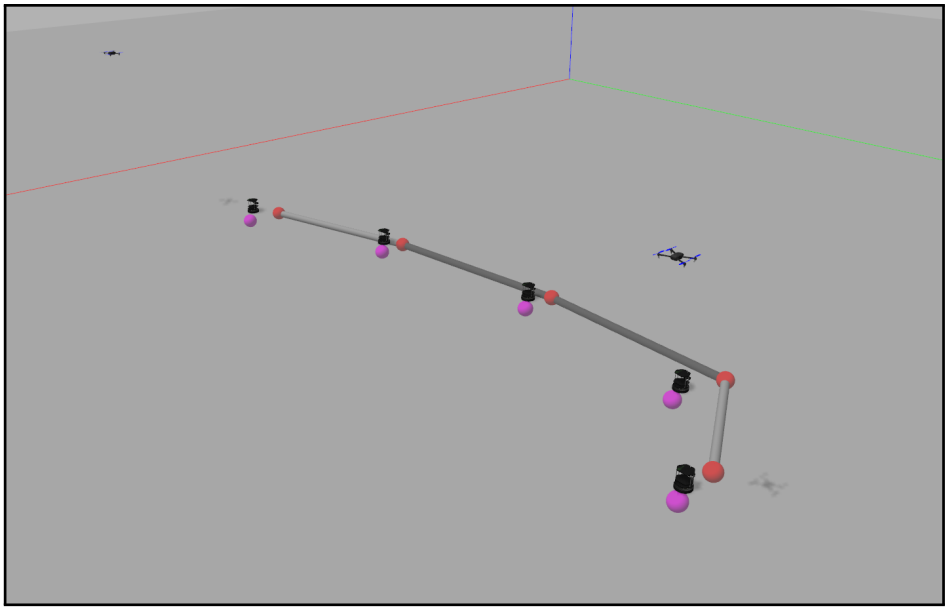


Figura 5.3: Disposición final de los robots en Gazebo en el caso 3.

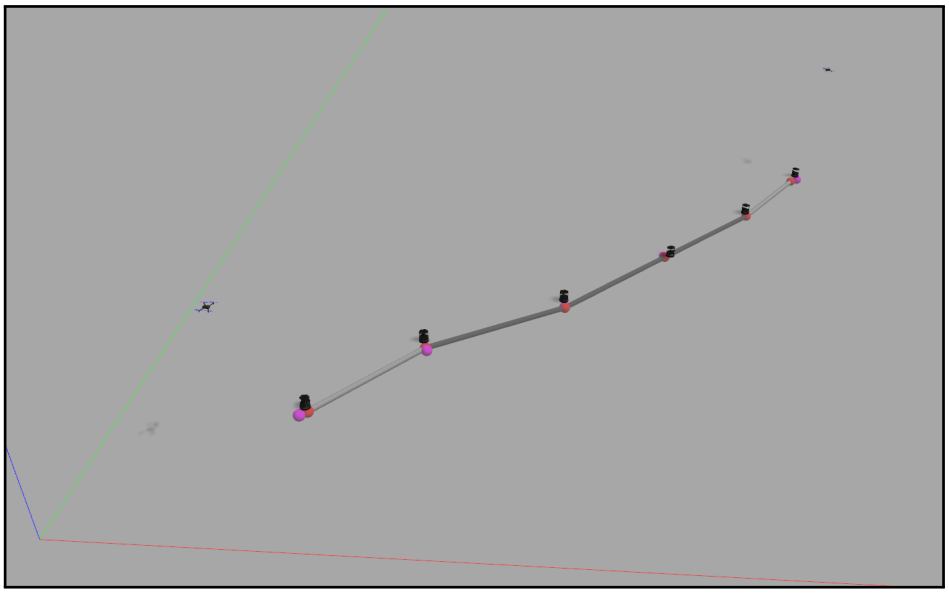


Figura 5.4: Disposición final de los robots en Gazebo en el caso 4.

En el caso 4, la velocidad mostrada es x16. En la primera iteración, los objetivos están tan cerca que los robots se juntan. A pesar de ello, a continuación consiguen estirarse en una línea para alcanzar a cubrir los dos objetivos (Figura 5.4), cumpliendo con la labor de mantenimiento de la conectividad gracias a que la mayoría de los arcos de comunicaciones se estiran hasta el límite.

En líneas generales, el comportamiento mostrado por el algoritmo es satisfactorio. Las únicas anomalías encontradas han sido a la hora de expandirse sin objetivos, que los robots no abarcan todo el área que podrían debido a la tolerancia del sistema de navegación, y cuando las trayectorias se cruzan, ya que los agentes se entorpecen mutuamente antes de volver a desplegarse.

5.2. Estudio paramétrico

En esta sección se sigue un procedimiento diferente. A partir de un experimento modelo, se van a variar ciertos factores estudiando las consecuencias que conllevan en el comportamiento del algoritmo. Además, estas simulaciones se ejecutarán tan solo desde MATLAB, evitando la elevada carga computacional del sistema Gazebo-ROS.

El experimento modelo consiste en una flota de seis robots que siguen a un conjunto de otros doce que marchan en una formación rectangular a razón de 0,3 metros por cada paso de simulación. Cada robot de la flota que realiza las tareas de seguimiento tiene un radio de percepción de 3 metros y, por tanto, un radio de comunicaciones de 6 metros. Además, el método de seguimiento implementado en este experimento base es el que modifica los límites de la zona de trabajo.

Tomando como base este experimento, en este estudio se van a variar cuatro factores de manera individualizada. Estos factores son la velocidad de la formación a seguir, el radio de percepción de los agentes, el número de robots de la flota perseguidora y el método de seguimiento.

Para estudiar cómo afecta cada uno de los factores al experimento, se representarán dos variables que ayudan a conocer el grado de cobertura ofrecido en cada caso. Una de ellas es la función de coste (Ec. 2.11) presentada en el Capítulo 2. Como se describió entonces, el problema de cobertura consiste en minimizar esta función de coste. Por lo que cuanto menor sea su valor, mejor será la cobertura de la zona de interés. Por otro lado, se representará otro valor definido como

$$\sum_{i=1}^{n} \|p_i - O\|\,, \tag{5.1}$$

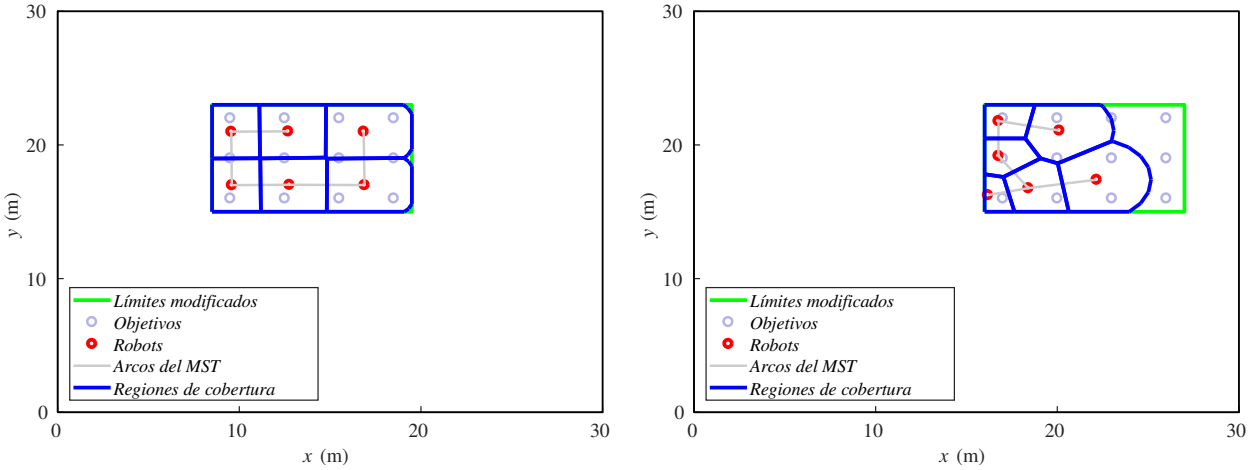
esto es, la suma de las distancias de cada agente al centro de la formación a seguir, O. Este parámetro también permite estimar si la calidad de la cobertura ofrecida es mayor o menor.

En los siguientes apartados se describen los cambios realizados y los resultados obtenidos para cada uno de los factores mencionados.

5.2.1. Velocidad de la formación

El primer factor a estudiar va a ser la velocidad de la formación, la cual, en las imágenes presentadas más adelante, avanza en línea recta de izquierda a derecha. Existe una velocidad a partir de la cual los agentes son incapaces de seguir a la formación objetivo con el algoritmo presentado en este trabajo. El objetivo de este apartado será averiguar el valor de esta velocidad límite para el caso concreto del experimento base.

Cabe destacar que, en todas las simulaciones, tanto los robots perseguidores como los que marchan en formación parten de la misma posición inicial. A su vez, en todas las simulaciones se ejecutan 30 pasos, por lo que la posición final de los objetivos será diferente para cada caso, como se contempla en la Figura 5.5. Sin embargo, son pasos suficientes para que el sistema sobrepase el régimen transitorio y, de este modo, las simulaciones sean comparables.



(a) v=0,25 m/paso (b) v=0,5 m/paso

Figura 5.5: Posiciones finales de los robots en función de la velocidad de la formación.

En la Figura 5.6 se representa la función de coste a lo largo de los pasos de simulación para diferentes valores de velocidad de la formación.

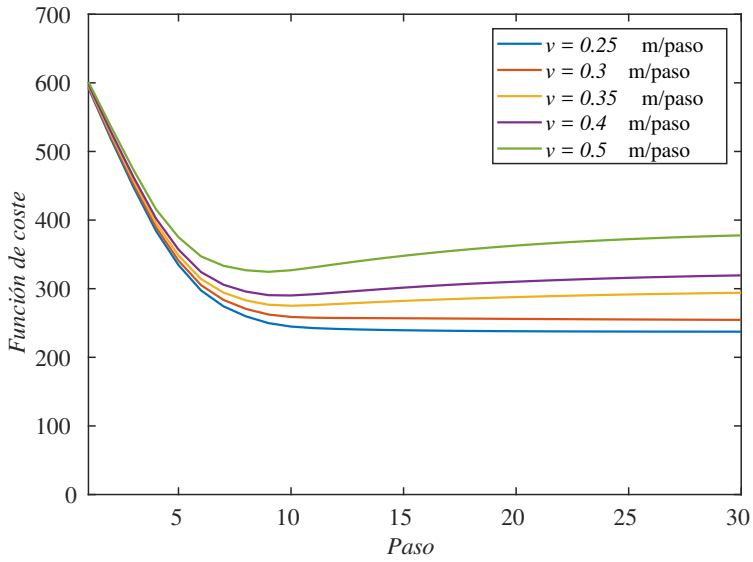


Figura 5.6: Función de coste en función de la velocidad de la formación.

Se observa que todas las gráficas parten del mismo punto, ya que las posiciones iniciales de los robots son iguales en todos los casos. A medida que avanza la simulación, los agentes se van desplegando y la función de coste disminuye. Si la velocidad de la formación es suficientemente baja, la función de coste se estabiliza debido a que los agentes se sitúan en unas posiciones que equilibran su velocidad de avance con la de la formación. Si por el contrario la velocidad de la formación es demasiado elevada, los agentes tienden a quedarse retrasados y la función de coste crece hasta un valor notablemente superior. La Figura 5.5, que representa las posiciones finales de los robots (en el paso 30) para las velocidades de 0,25 y 0,5 m/paso, sirve como ejemplo de estos razonamientos.

En cuanto a la suma de las distancias al centro de la formación, la lectura de la gráfica (Fig. 5.7) es muy similar a la de la función de coste. Inicialmente las distancias son mayores y, conforme se despliegan los robots, se van reduciendo hasta un valor estable en el caso de que la velocidad sea baja. Si la velocidad es demasiado elevada, las distancias crecen hasta un valor superior al quedarse retrasados los agentes.

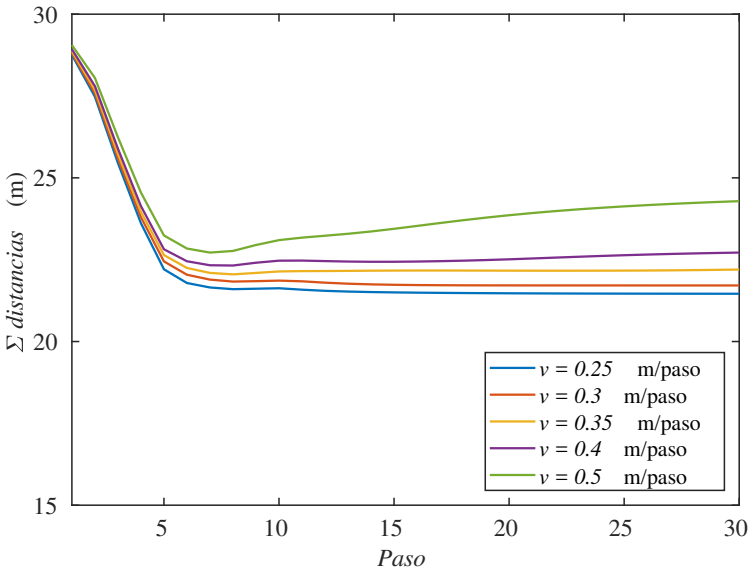


Figura 5.7: Sumatorio de las distancias de los robots al centro de la formación en función de la velocidad de la misma.

Como un apunte crítico, cabe mencionar que podría darse algún caso en el que el dato de esta variable fuese engañoso. Observando la Figura 5.5a, si los dos agentes situados más a la derecha se situaran simétricamente, con respecto al centro de la formación, en el lado izquierdo, el valor obtenido sería el mismo pero la cobertura sería mucho peor. Sin embargo, para realizar una comparativa como la de esta sección en la que se va variando el valor de un sólo factor es una variable perfectamente válida, como se observa en el paralelismo de su comportamiento con el de la función de coste.

Conforme a las premisas expuestas anteriormente, la velocidad máxima que es capaz de seguir la flota de robots que lleva implementado el algoritmo se encuentra en torno a los 0,3 metros por paso de simulación. Estas unidades de velocidad dejan lugar a que un sistema que compute el algoritmo en menos tiempo sea capaz de seguir formaciones más rápidas que otro sistema más lento.

Al observar que la velocidad de 0,3 m/paso es suficientemente baja para el seguimiento de la formación, pero a su vez está cerca del límite, se escoge como valor del experimento base con el objetivo de poder observar con mayor facilidad cómo afectan el resto de factores al sistema.

5.2.2. Radio de percepción

El siguiente factor que se va a evaluar es el radio de percepción de los sensores que incorporan los robots. Se varía el valor del radio en pasos de 0,5 metros en torno al valor establecido en el experimento modelo. El valor más pequeño que se evalúa es de 2 metros, ya que las posiciones iniciales de los robots están separadas más de 3 metros, y la comunicación entre ellos sería imposible al establecer el radio en 1,5 metros. El objetivo es observar la diferencia en el comportamiento del sistema al aumentar o al reducir el radio de los sensores, y con ello determinar si se justifica el implementar sensores de mayor rango o no.

De nuevo, todos los robots parten de la misma posición inicial. En este caso se simulan 40 pasos en cada experimento para observar algunos fenómenos con mayor claridad.

En la Figura 5.8 se representa la función de coste a lo largo de los pasos de simulación para diferentes valores del radio de percepción.

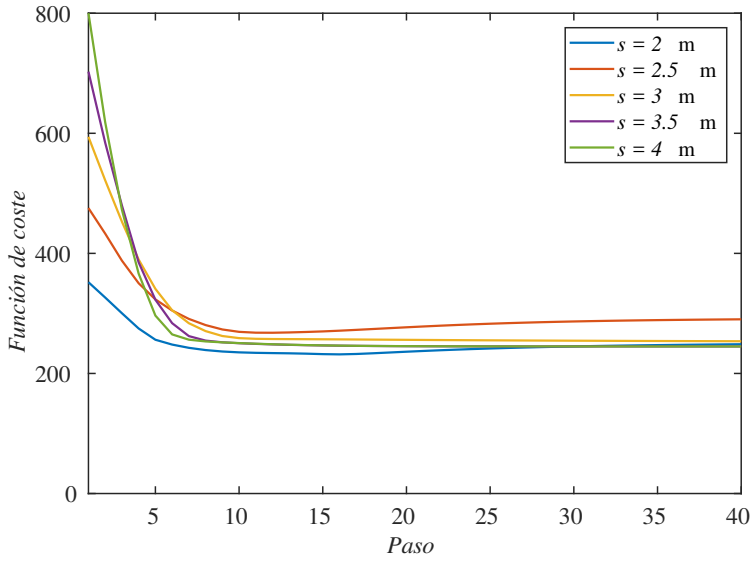
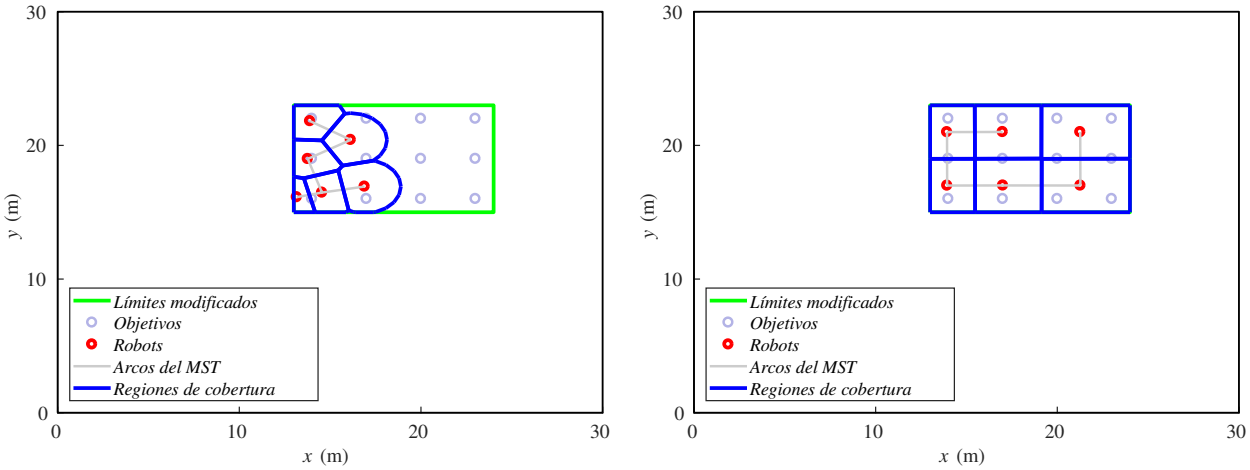


Figura 5.8: Función de coste en función del radio de percepción de los sensores.

En esta ocasión, las curvas no parten del mismo punto, ya que la función de coste se evalúa en las regiones de cada robot y éstas son más pequeñas cuanto menor es el radio. A pesar de ello, se observa que con radios mayores se alcanza un valor de coste inferior que con el más pequeño. De todos modos, la cobertura conseguida con el menor de los radios es mucho peor que con los radios mayores, como se observa en la Figura 5.9. Otro comportamiento que se puede apreciar es que, aunque el radio aumente, la función de coste tiende a un mismo valor. Esto quiere decir que los robots se sitúan en la misma posición, ya que es la que equilibra el avance de los agentes con la velocidad de la formación.



(a) s=2 m (b) s=3.5 m

Figura 5.9: Posiciones finales de los robots en función del radio de percepción.

Con la gráfica de la suma de las distancias al centro de la formación (Fig. 5.10) se corrobora el hecho de que existe una posición de equilibrio para los agentes, que no se modifica aunque se

aumente el radio de los sensores. Además, se observa un comportamiento sobreoscilante cuando los radios son muy bajos, para las posiciones iniciales dadas en este caso en concreto.

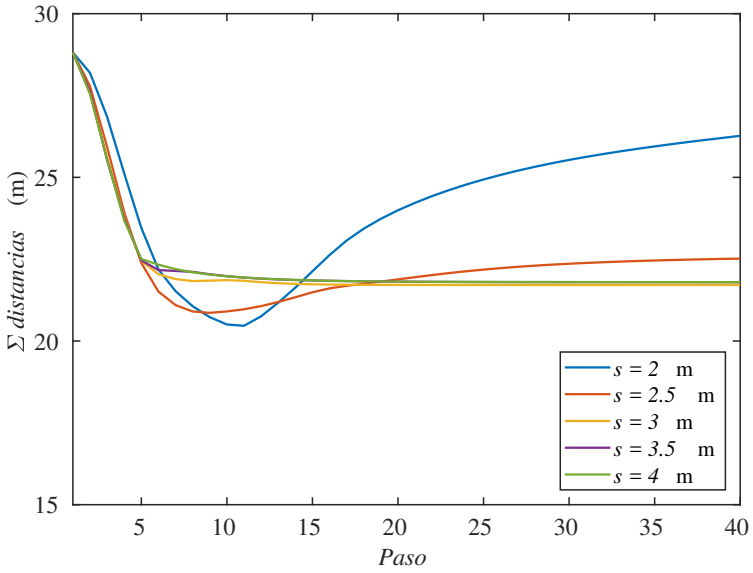


Figura 5.10: Sumatorio de las distancias de los robots al centro de la formación en función del radio de percepción de los sensores.

A la vista de los resultados obtenidos, una vez conocidas las características del sistema cabe plantearse la necesidad de realizar un estudio de esta índole antes de adquirir los sensores que se incorporarán en los robots de una implementación real. A partir de cierto valor, un sensor con mayor radio de percepción no presentará un comportamiento mejor que otro con menor rango sensitivo. De este modo, se puede ahorrar una importante cantidad de dinero si el sistema consta de muchos robots.

5.2.3. Número de agentes

En esta ocasión, se evalúa cómo afecta al sistema el hecho de variar el número de robots que siguen a la formación. Con objeto de resaltar las diferencias, se va a ir duplicando iterativamente el número de robots desde los 4, que ya puede ser considerado un sistema multi-robot, hasta los 32, un valor exageradamente grande para la zona de trabajo de este experimento. El objetivo es estudiar el comportamiento del sistema al variar el número de robots, comprobando si merece la pena aumentar la inversión o no.

En este caso los robots no pueden partir siempre de la misma posición, ya que su número va variando entre las distintas simulaciones. Partirán desde una posición próxima a la del resto de estudios realizados en este capítulo. La simulación se realiza con 30 pasos.

En la Figura 5.11 se representa la función de coste a lo largo de los pasos de simulación para diferentes cantidades de robots perseguidores. Se observa como las curvas no parten del mismo punto. Cuantos más robots hay en la escena, menor es la función de coste. Esto es debido a que las regiones asociadas a cada robot se hacen más pequeñas al incluir más robots en el mismo espacio. Con el discurrir de la simulación, los agentes se despliegan y la función de coste tiende a un valor constante, como en casos anteriores. Sin embargo, en la curva de 32 robots se observa que finalmente asciende. Este fenómeno se produce porque una gran cantidad de robots se aglomera en la parte posterior de la formación y se entorpecen mutuamente, haciendo que en la parte delantera las regiones de cobertura crezcan.

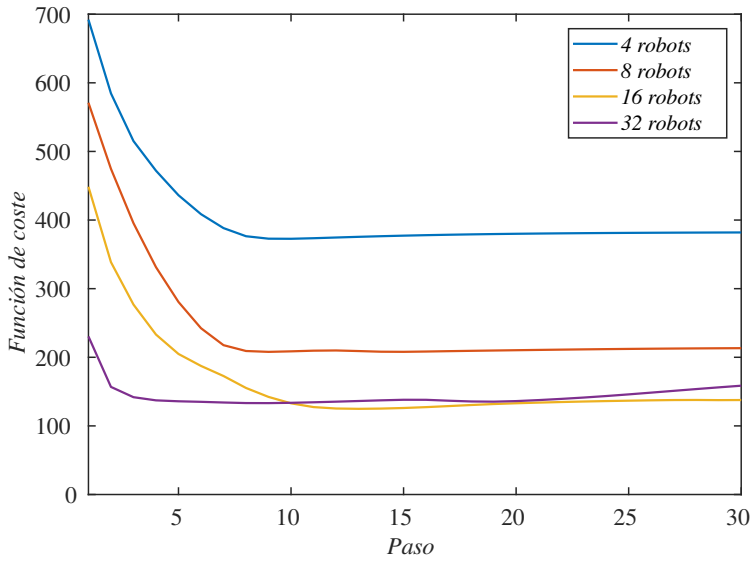
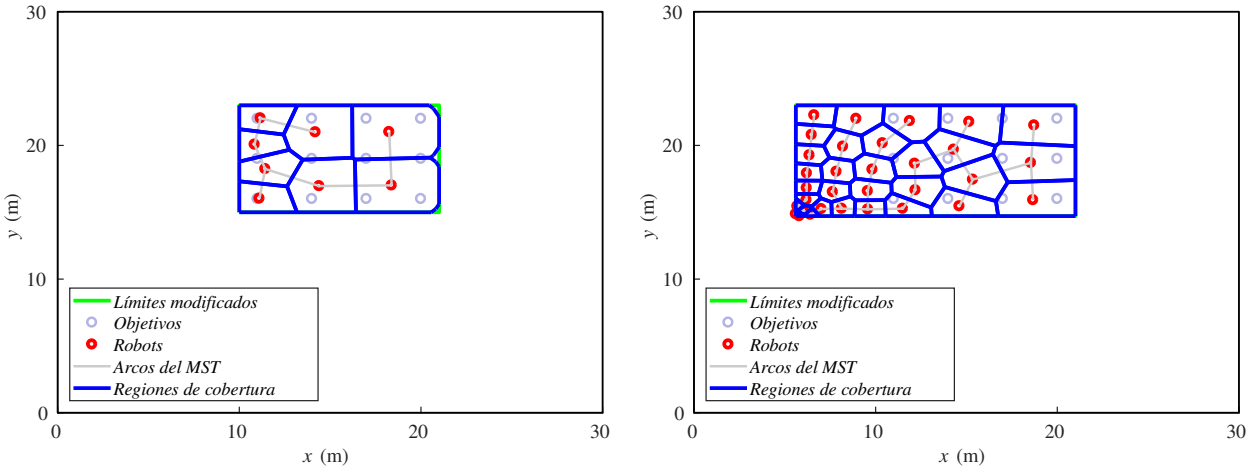


Figura 5.11: Función de coste en función del número de robots.

En la Figura 5.12 se puede comprobar esta situación, comparando la posición final de 8 y de 32 robots. En la Figura 5.12b se observa como los agentes quedan muy retrasados mientras la formación sigue avanzando.



(a) 8 robots

(b) 32 robots

Figura 5.12: Posiciones finales de los robots en función del número de agentes.

A la hora de representar las distancias al centro de la formación, en este caso no se puede tomar el valor de la suma, ya que en cada simulación hay un número diferente de robots. En esta ocasión hay que tomar el valor medio para cada grupo. En la Figura 5.13 se comprueba que todas las curvas tienden a un valor constante a excepción de la de 32 robots, por las razones explicadas anteriormente. Además, el valor de la distancia en estado permanente es menor cuanto menor es el número de robots. Esto se justifica porque el exceso de agentes tiende a acumularse en la parte trasera de la formación y, por tanto, hacen crecer la distancia media.

Con todo ello, se puede concluir que es necesario realizar un estudio previo a una posible implementación real para determinar el número de robots óptimo para el sistema en cuestión. Un defecto de agentes impedirá la cobertura completa de la zona de interés. Por el contrario, un exceso de robots entorpecerá las labores de seguimiento y despliegue. Por tanto, un estudio detallado puede suponer un gran ahorro en robots.

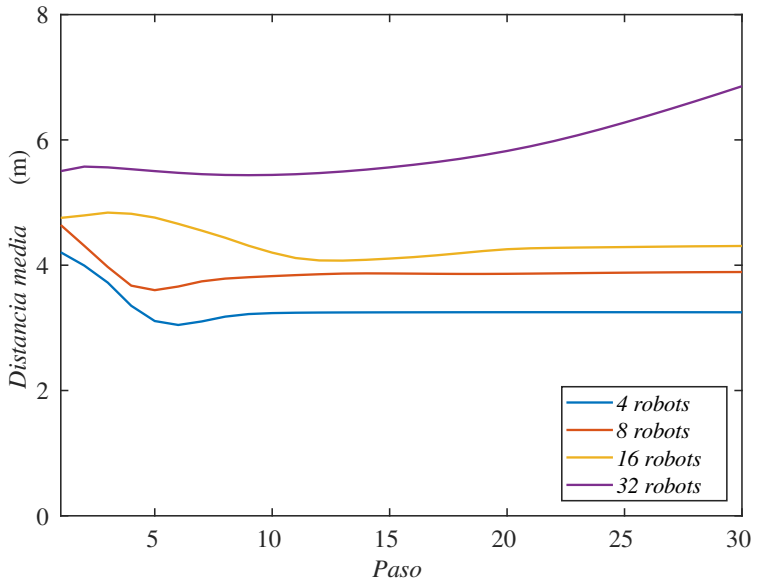


Figura 5.13: Distancia media de los agentes al centro de la formación en función del número de robots.

5.2.4. Método de seguimiento

En último lugar, se evalúa el comportamiento del sistema al variar el método de seguimiento implementado en los agentes. Los dos métodos a probar son los introducidos en el Capítulo 3 de esta memoria.

El método que consiste en modificar los límites ficticios del área de trabajo ha estado implementado en todo el estudio de esta sección. En este apartado se va a tratar de comparar con el método de las funciones de importancia. Para ello no se va a poder emplear la función de coste, ya que, al aplicar las funciones de importancia, su valor se reduce en gran medida (recordar la Tabla 3.1) y no es comparable a los valores obtenidos con el otro método.

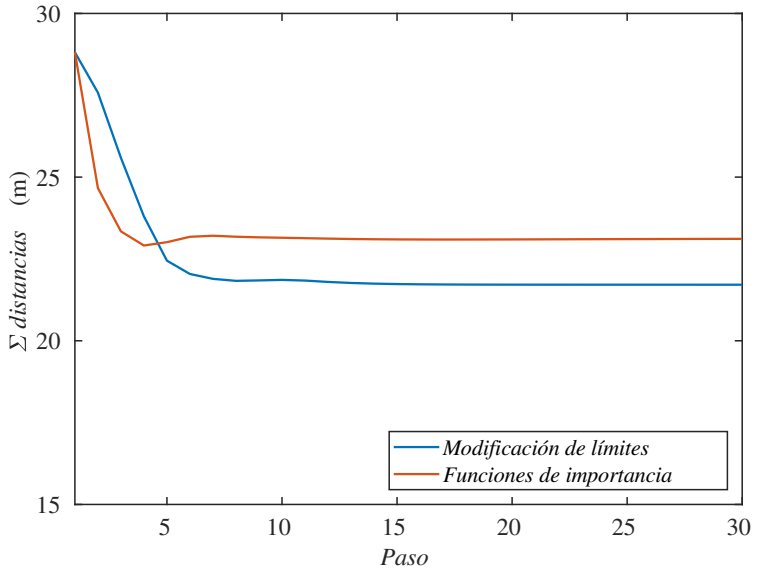
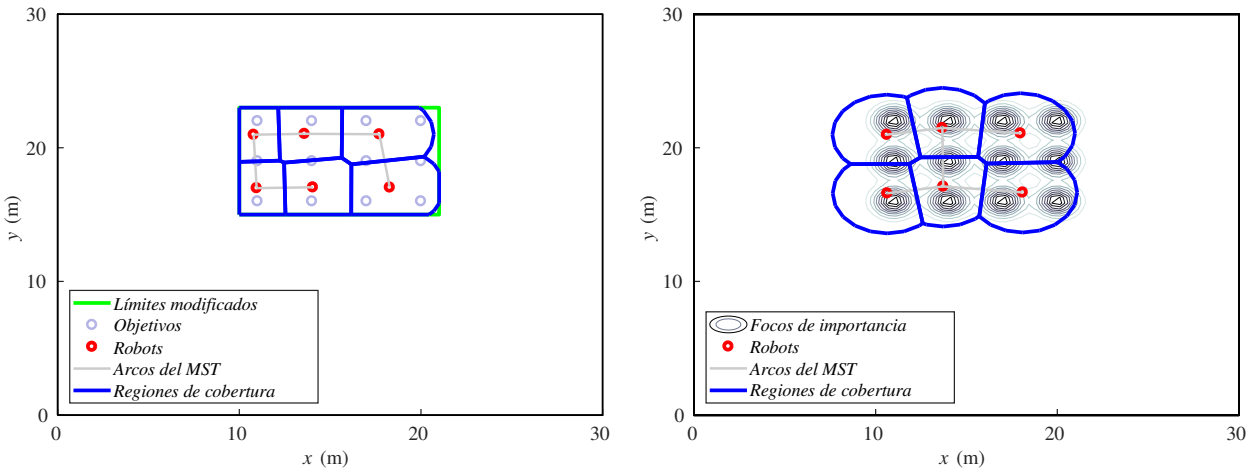


Figura 5.14: Sumatorio de las distancias de los robots al centro de la formación en función del método de seguimiento.

En la Figura 5.14 se muestra la suma de las distancias de los robots al centro de la formación objetivo en función del método empleado. Se puede comprobar que, en este caso, el método de las

funciones de importancia favorece una mejor distribución de los agentes en la zona de interés. Se podría esperar que el método de la modificación de los límites se viera favorecido en el caso de perseguir una formación, sin embargo, sucede lo contrario. El hecho de que el radio de percepción de los sensores sea suficientemente grande para abarcar parte de la función de importancia de los robots contiguos, permite que los agentes no se queden estancados con los primeros objetivos que encuentran en su camino. Si los robots de la formación estuvieran más separados, el mínimo local de un agente se encontraría en el centro de la primera función de importancia que encontrara, y no se moverían más allá de esa posición.

En la Figura 5.15 se recogen las posiciones finales de los robots para las dos configuraciones estudiadas. Se observa que las disposiciones finales son muy similares, quedando todos los robots cubiertos en ambos casos.



(a) Modificación de los límites

(b) Funciones de importancia

Figura 5.15: Posiciones finales de los robots en función del método de seguimiento empleado.

Como conclusión, de nuevo se hace necesario un estudio previo a una implementación real para elegir el método de seguimiento en función de los parámetros del sistema, con especial atención al radio de los sensores y a la distancia entre los robots de la formación objetivo.

Capítulo 6

Conclusiones y líneas futuras

6.1. Conclusiones

Las conclusiones que se desprenden del presente Trabajo Fin de Máster abarcan diferentes ámbitos.

En primer lugar, se ha logrado desarrollar un algoritmo de cobertura con mantenimiento de la conectividad que es capaz de realizar tareas de seguimiento. Para ello se han ideado dos alternativas, que presentan sus ventajas e inconvenientes.

El método de adjudicar funciones de importancia a los objetivos es más adecuado para situaciones en las que hay pocos puntos de interés. Los agentes alcanzan los centros de interés con mayor rapidez pero, en función de la posición inicial relativa, es posible que algún objetivo quede sin cubrir.

Por su parte, el método de modificar los límites de la zona de trabajo es más adecuado para situaciones en las que hay una concentración de puntos de interés. Los agentes se despliegan de una manera más uniforme, lo cual implica que si los objetivos están separados se tiende a cubrir zonas vacías intermedias.

Del estudio paramétrico también se han extraído conclusiones valiosas. Por un lado, se ha demostrado la existencia de una velocidad límite que el método de seguimiento es capaz de seguir. Por otro lado, el propio estudio ha mostrado la importancia de realizar este tipo de simulaciones con carácter previo a una implementación real. Se puede ahorrar una importante cantidad de dinero en material encontrando la configuración óptima del sistema antes de implementarlo.

Cambiando de tercio, también se ha conseguido implementar el sistema multi-robot sobre el simulador Gazebo, con ayuda de las herramientas de ROS y MATLAB. La plataforma de ROS todavía no dispone de muchas funcionalidades encaminadas hacia los sistemas multi-robot, pero sí que se observa una tendencia en la comunidad.

El sistema de visualización desarrollado ayuda a la comprensión de los movimientos que realizan los robots en Gazebo. Mejora mucho el aspecto visual a pesar de hacer la simulación bastante más lenta.

6.2. Líneas futuras de investigación

Las líneas futuras en las que se pueden embarcar nuevas investigaciones acerca del tema tratado en este trabajo son muchas.

En primer lugar, como ya se ha mencionado en la introducción, cabría la posibilidad de implementar el algoritmo desarrollado de manera descentralizada. Uno de los objetivos principales de los sistemas multi-robot es, precisamente, repartir el mando de los algoritmos entre todos los agentes. Por lo que una implementación distribuida sería mejor vista en este aspecto.

Esta implementación abriría la puerta a otra posibilidad, que es simular el algoritmo al completo con Gazebo y ROS, sin emplear MATLAB. Cada robot implementaría todos los nodos y algoritmos necesarios para una comunicación entre agentes efectiva, que en el cómputo global ejecutara las labores de seguimiento.

En esta línea, también se podría llevar a cabo una implementación sobre robots reales. Para ello habría que desarrollar un sistema de localización efectivo para estos entornos tan dinámicos, empleando por ejemplo cámaras externas o dispositivos de visión diferentes a los empleados en este trabajo.

En cuanto al algoritmo, se podría estudiar un método de seguimiento que fuera un híbrido entre las dos soluciones propuestas. Si, en un primer instante, los robots se desplegaran homogéneamente con el método de modificación de los límites hasta que estuvieran repartidos en la zona de interés, a continuación se podrían llevar a cabo las labores de seguimiento con el método de las funciones de importancia, que es más rápido.

Poniendo la vista en el estudio paramétrico, otra posibilidad de investigación sería encontrar la relación matemática que existe entre la velocidad límite de la formación que el algoritmo es capaz de seguir y las capacidades visuales de los robots.

Bibliografía

[1] J. Cortés, S. Martínez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, pp. 243–255, April 2004.

[2] "MATLAB - MathWorks." `https://www.mathworks.com/products/matlab.html`.

[3] "Gazebo." `http://gazebosim.org/`.

[4] "Documentation - ROS Wiki." `http://wiki.ros.org/`.

[5] M. Schwager, D. Rus, and J.-J. Slotine, "Unifying geometric, probabilistic, and potential field approaches to multi-robot deployment," *The International Journal of Robotics Research*, vol. 30, no. 3, pp. 371–383, 2011.

[6] R. G. Gallager, P. A. Humblet, and P. M. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Transactions on Programming Languages and systems (TOPLAS)*, vol. 5, no. 1, pp. 66–77, 1983.

[7] J. Cortés, S. Martínez, and F. Bullo, "Spatially-distributed coverage optimization and control with limited-range interactions," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 11, no. 4, p. 691–719, 2005.

[8] R. Aragüés, C. Sagüés, and Y. Mezouar, "Triggered minimum spanning tree for distributed coverage with connectivity maintenance," in *2014 European Control Conference (ECC)*, pp. 1881–1887, June 2014.

[9] "turtlebot_gazebo/Tutorials - ROS Wiki." `http://wiki.ros.org/turtlebot_gazebo/Tutorials`.