Universidad Zaragoza

1542

Trabajo Fin de Grado

Implementación de estrategias de atrapamiento utilizando equipos de robots

Implementation of entrapment strategies using teams of robots

Autor

Juan Rueda Pérez

Director

Eduardo Montijano Muñoz

ESCUELA DE INGENIERÍA Y ARQUITECTURA
2017

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Juan Rueda Pérez,

con nº de DNI 73027614S en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Implementación de estrategias de atrapamiento utilizando equipos de robots,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 22 de Septiembre de 2017

Fdo: Juan Rueda Pérez

Implementación de estrategias de atrapamiento utilizando equipos de robots
RESUMEN

La investigación en equipos de robots que cooperan para realizar tareas ha adquirido un importante auge en los últimos años. La utilización de sistema multi-robot permite realizar tareas de forma más eficiente, segura y robusta, como en el caso del atrapamiento o la escolta de objetivos. Sin embargo, al aumentar el número de robots que realizan estas tareas, su ejecución es cada vez más compleja debido a las restricciones de movimiento de los robots.

El principal objetivo de este trabajo es la implementación de un algoritmo de atrapamiento en un sistema multi-robot real. Dicho algoritmo debe tener en cuenta posibles errores en la información local y global de la posición del target, así como en la movilidad de los robots involucrados.

Para este proyecto se ha partido de dos artículos de investigación para el estudio del control de sistemas multi-robot. En este trabajo se ha analizado un algoritmo de atrapamiento y se ha modificado para posibilitar su implementación en robots reales. Además se ha implementado un control de movimiento para robots no holónomos. Con las modificaciones realizadas y el control de movimiento ha sido posible realizar simulaciones del algoritmo de atrapamiento en un entorno realista.

Para llevar a cabo estas simulaciones ha sido necesario además implementar un sensor para medir distancia, posibilitando la creación de mapas y la localización de los robots en ellos. Por último se ha adaptado el código para trabajar con dichos mapas y poder realizar los experimentos con robots reales en el laboratorio.

Índice general

1. INTRODUCCIÓN

1.1. Motivación

La utilización de equipos de robots que cooperan para la realización de tareas ha adquirido un importante auge en la investigación en los últimos años y se ha incrementado el abanico de sus posibles aplicaciones tales como sistemas de vigilancia o de búsqueda y rescate, entre otros. Los sistemas multi-robot son también llamados sistemas robóticos multiagente, debido a que sus categorías involucran implícitamente el concepto de robot agente. Así, para poder llevar a cabo una mínima interacción cada robot debe percibir su medio, poseer mecanismos de decisión de acciones y además un conocimiento de la evolución del medio con las acciones realizadas por él y por los integrantes del equipo.

Uno de los problemas más estudiados en este tipo de sistemas es el problema de atrapamiento o escolta de objetivos [1]. En estos casos, la finalidad es alcanzar una posición en la cual los robots queden distribuidos uniformemente en torno al objetivo, impidiendo que esté salga, o que otros puedan entrar en el recinto.

Uno de los principales problemas en el atrapamiento es determinar la posición del objetivo. La hipótesis típica de las soluciones propuestas en la literatura es suponer que todos los robots tienen información precisa del objetivo a capturar. Sin embargo, en situaciones reales la posición del objetivo aunque se encuentra en cierto modo delimitada, no se conoce con exactitud. Otro problema lo constituyen las limitaciones que existen en el movimiento de los robots, siendo más complejo a medida que se aumenta el número de robots que componen el sistema, ya que se dificulta la ejecución de sus tareas.

Por todo ello, resulta interesante buscar nuevas soluciones que permitan al equipo de robots realizar una distribución lo más homogenea posible, incluso en situaciones en las que la información del objetivo no se conoce con precisión o no es compartida por todos los miembros del equipo. En este Trabajo Fin de Grado se muestra una solución completa

a los problemas que se plantean en las situaciones de atrapamiento, tanto en un entorno virtual como en una situación real.

1.2. Objetivos

El principal objetivo de este trabajo ha sido la implementación de un algoritmo de atrapamiento en un sistema multi-robot real con información imprecisa del objetivo. En concreto los objetivos a alcanzar en este proyecto han sido:

- Proponer un algoritmo de atrapamiento capaz de realizar una distribución homogénea entorno a targets con información incierta.
- Implementar un algoritmo para gestionar las restricciones en el movimiento de robots no holónomos.
- Establecer las modificaciones necesarias en el código para realizar simulaciones en una plataforma real.
- Combinar el algoritmo de atrapamiento con un algoritmo de localización para robots en entornos reales.

1.3. Alcance

En este trabajo se ha partido de dos artículos de investigación para el estudio del control de sistemas multi-robot. Un artículo trata sobre el estudio de algoritmos de atrapamiento en sistemas multi-robot [3]. En este trabajo se ha complementado para añadirle restricciones en el movimiento y posibilitar la simulación tanto en robots reales como en Gazebo (plataforma de simulación que permite visualizar el comportamiento de un robot en un mundo virtual 3D).

El otro artículo trata sobre el control de robots no holónomos sin un marco de referencia global [2]. En este trabajo se ha implementado el código para que los robots se comporten como se describe en dicho artículo.

Además se ha realizado la implementación de un sensor para medir distancia (láser Hokuyo) en el entorno de simulación Gazebo.

Finalmente se ha adaptado todo el código para trabajar en mapas reales, generados utilizando técnicas de localización, y así poder realizar los experimentos de atrapamiento con robots reales.

1.4. Contenido de la memoria

La presente memoria explica los pasos seguidos para alcanzar los objetivos propuestos en este proyecto, así como los resultados obtenidos durante la elaboración del mismo.

En el primer capítulo se presenta una introducción al proyecto, donde se muestran las investigaciones previas en este campo así como los objetivos a alcanzar.

En el segundo capítulo se plantea la formulación del problema de atrapamiento incluyendo los problemas derivados de no poseer información global y local precisa, así como las limitaciones en el movimiento de los robots.

El tercer capítulo muestra las modificaciones efectuadas en el algoritmo de atrapamiento para solventar dichos problemas.

En el cuarto capítulo, partiendo de simulaciones en un entorno virtual, se incluye el proceso realizado para implementar el algoritmo en robots reales.

El quinto capítulo presenta las conclusiones obtenidas en este proyecto, mostrando las dificultades encontradas durante su realización y las posibles líneas de trabajo futuras.

2. FORMULACIÓN DEL PROBLEMA

En este capítulo se expone la formulación del problema de atrapamiento de un sistema multi-robot, partiendo del caso ideal, y añadiendo sucesivamente los problemas que se encuentran en su adaptación al mundo real.

En el caso general, se asume un sistema multi-robot compuesto por N robots móviles, moviéndose en un espacio Euclideo de 2 dimensiones. Además se supone que cada robot tiene un identificador único conocido por los demás robots (p.e. dirección MAC, nombre,...). De esta forma denotamos la posición de cada robot como:

$$p_i(t) \in \mathbb{R}^2, i \in \{1, ..., N\}. \tag{2.1}$$

Además en el caso general se asume un punto de referencia común y disponible para todos los robots. Los robots considerados en los algorítmos de atrapamiento clásicos se comportan de forma holónoma, es decir, no tienen restricciones en el movimiento, quedando su velocidad definida como:

$$\begin{aligned} \dot{x}_i &= u_{xi} \\ \dot{y}_i &= v_{xi}. \end{aligned} \tag{2.2}$$

El objetivo del sistema multi-robot es atrapar el target denotado como p_0. En los algorítmos de atrapamiento clásicos se establece una órbita circular en torno al target, y todos los robots se desplazan en torno a dicha órbita. Además todos los robots conocen la posición exacta del target. Por lo tanto, conocidas las posiciones del target y del robot, denotamos el ángulo formado entre ellos como $\psi_i \in (0, 2\pi)$. Dichos parámetros se muestran en la Figura 2.1.

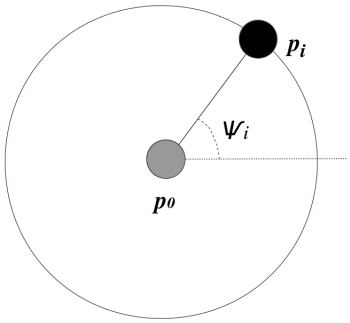


Figura 2.1: Atrapamiento clásico de un target.

Sin embargo, a la hora de localizar el target en los casos reales nos encontramos con situaciones en las que la información global es incierta, debido a que la información que tenemos del target es incompleta, y problemas de incertidumbre ya que la información sobre la posición local es imprecisa.

Debido al problema de percepción, la información global disponible sobre la posición del target es incierta por lo que ya no es posible establecer una órbita circular, sino que hay que utilizar una órbita elíptica. Esta región elíptica queda definida como:

$$\varepsilon = (\widehat{p}_0, a_0, \phi_0). \tag{2.3}$$

Donde $\widehat{\boldsymbol{p}}_{\mathbf{0}}$ ahora es el centro de la elipse, $\boldsymbol{a_0} \in \mathbb{R}^2$ contiene los semiejes mayor y menor de la elipse y $\boldsymbol{\phi}_0 \in (0, 2\pi)$ es la orientación de la elipse en el plano. Estos parámetros quedan reflejados en la Figura 2.2.

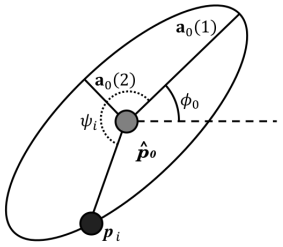


Figura 2.2: Atrapamiento de un target con información global incierta [3].

Además, como se ha indicado podemos encontrarnos con situaciones en las que la información local sobre la posición del target es incierta. Debido a este problema, la posición estimada por los robots de un target único no suele coincidir, lo que conduce a la existencia de N targets, correspondiéndose cada uno con el centro de una región elíptica $\varepsilon^i = (\widehat{p}_0^i, a_0^i, \phi_0^i)$. Estos targets están localizados en una región muy próxima a la del target

real. Este problema se representa en la Figura 2.3.

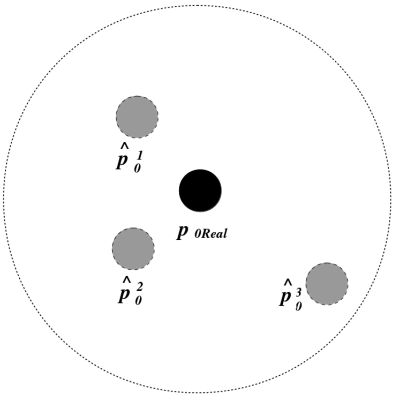


Figura 2.3: Incertidumbre en la posición del target.

Por otra parte, en los casos reales, existen restricciones que hacen que los robots se comporten de forma no holónoma, limitando su capacidad de movimiento, estando por lo tanto su velocidad determinada como:

$$\begin{aligned} \dot{x}_i &= V \times cos\theta \\ \dot{y}_i &= V \times sin\theta \\ \dot{\theta} &= \phi. \end{aligned} \tag{2.4}$$

3. ALGORITMO DE ATRAPAMIENTO

En este capítulo se va a tratar el acercamiento que se ha realizado al algoritmo de atrapamiento. Como en el capítulo anterior, se partirá desde el caso más simple, añadiendo sucesivamente problemas y restricciones generados por posibles errores de medición, ruido o movimiento de los agentes.

El objetivo de las simulaciones del algoritmo de atrapamiento es que los robots acaben distribuidos uniformemente en torno al target. Esta distribución se alcanza facilmente en los casos mas sencillos y se complica gradualmente a medida que se incluyen más problemas y restricciones.

Todas las situaciones que se describen en este capítulo han sido probadas veinte veces en diferentes escenarios para verificar su correcto funcionamiento. No obstante, para facilitar la compresión de los resultados, en este capítulo se muestran las figuras que corresponden a una misma configuración de los robots a excepción de los reflejados en la Figura 3.8.

3.1. Situación ideal

El caso más sencillo que se plantea del algoritmo de atrapamiento es aquel en el que todos los agentes se desplazan por una única circunferencia y tienen la misma estimación del target, no existiendo problemas de percepción ni de incertidumbre. Además no se consideran restricciones de movimiento en los agentes, es decir, se supone un comportamiento holónomo.

Al inicio del atrapamiento, cada agente determina su posición angular respecto del target, situándose en un punto de la circunferencia. Para poder realizar el atrapamiento del target se parametriza la trayectoria que el robot i sigue alrededor de la circunferencia mediante el ángulo $\psi_i \in (0, 2\pi)$:

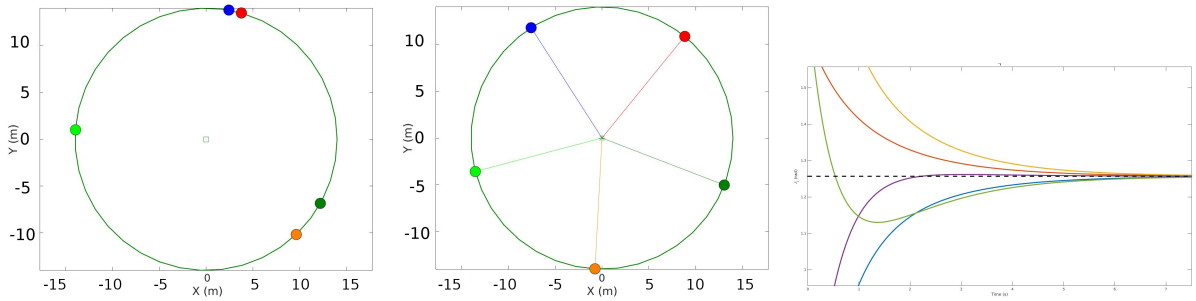
$$p_i(\psi_i) = [r \times cos(\psi_i), r \times sin(\psi_i)]^T + p_0. \tag{3.1}$$

De esta forma, cada robot calcula la posición angular relativa de todos los demás robots, pero sólo se comunica con el inmediatamente anterior y posterior, describiendo una topología de anillo. Así la dinámica de ψ_i resulta ser la suma de las diferencias angulares de los robots i-1 e i+1.

$$\dot{\psi_i} = (\psi_{i+1} - \psi_i)\ mod\ 2\pi + (\psi_{i-1} - \psi_i)\ mod\ 2\pi. \tag{3.2}$$

La distancia angular entre dos robots queda expresada como $d_i = (\psi_{i+1} - \psi_i)\ mod\ 2\pi$, haciendo converger el sistema cuando el $lim_{t\to\infty} d_i(t) = 2\pi/N, \forall i$.

Las gráficas que se derivan de esta implementación se muestran en la Figura 3.1:



(a) Posición inicial (b) Posición final (c) Distancia angular entre agentes

Figura 3.1: Algoritmo de atrapamiento en situación ideal.

Como puede observarse la distancia angular de los robots converge, alcanzando una posición de equilibrio perfecta y estando separados entre ellos por una distancia angular 1,256 radianes (72º). Además, en este caso, la distancia absoluta entre los robots es la misma.

3.2. Información global incierta

En esta sección se desarrolla la forma en que se ha tratado el problema de percepción del target para llegar a una solución. Debido a este problema, la información global de la que disponemos no es exacta, por lo que no es posible utilizar una órbita circular, sino que hay que utilizar órbitas elípticas. La dimensión de los ejes de esta elipse dependerá del

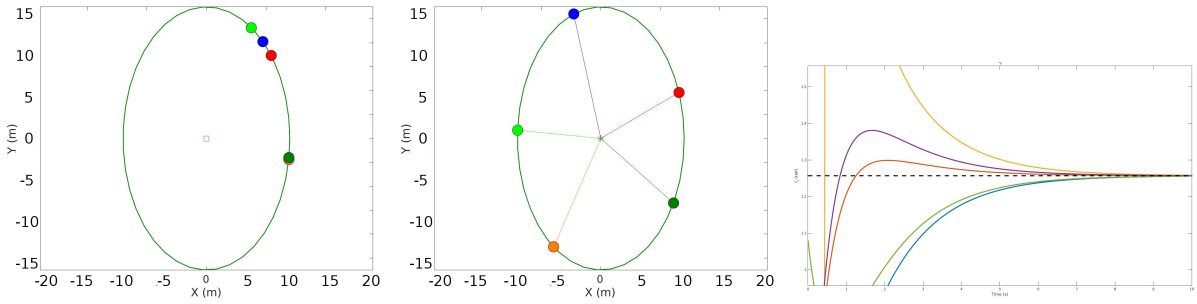
nivel de confianza de las estimaciones realizadas sobre la posición del target. De esta forma, la parametrización de la trayectoria que el robot i sigue alrededor de la elipse se define mediante el ángulo $\psi_i \in (0, 2\pi)$, siendo ψ_i en este caso:

$$p_i(\psi_i) = R(\phi_0)[a_0(1)cos(\psi_i), a_0(2)sin(\psi_i)]^T + \widehat{p}_0. \tag{3.3}$$

Donde $\boldsymbol{R}(\phi_0) \in \mathbb{R}^{2x2}$ es la matriz de rotación asociada al ángulo ϕ_0.

$$\boldsymbol{R}(\phi_0) = \begin{pmatrix} cos(\phi_0) & sin(\phi_0) \\ -sin(\phi_0) & cos(\phi_0) \end{pmatrix} \tag{3.4}$$

Las gráficas que se derivan de esta implementación se muestran en la Figura 3.2.



(a) Posición inicial (b) Posición final (c) Distancia angular entre agentes

Figura 3.2: Algoritmo de atrapamiento con información global incierta.

Como se puede apreciar, al igual que en el caso anterior, los robots encuentran una posición de equilibrio en la elipse, siendo la distancia angular entre ellos 1,256 radianes. No obstante, en este caso, la distancia absoluta entre ellos ya no es la misma.

3.3. Información local incierta

El siguiente problema a tratar es la incertidumbre en la información local de la posición del target. En un sistema de N agentes con incertidumbre, cada uno realiza una estimación de la posición del target, dando lugar a N elipses con diferentes radios, ángulos de rotación y centros.

Como cada robot esta situado en una elipse diferente, necesitamos definir las proyecciones que cada robot forma con las diferentes elipses, para así poder calcular la distancia

angular entre ellos. Así, $\mathcal{P}(\psi_j, \varepsilon^i)$ es la proyección del robot j sobre la elipse del robot i y $\delta_i = \mathcal{P}(\psi_{i+1}, \varepsilon^i) - \psi_i$ la distancia angular entre los robots (Figura 3.3).

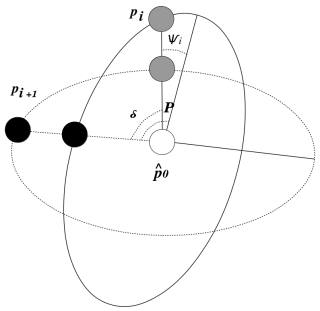


Figura 3.3: Elipses y sus proyecciones.

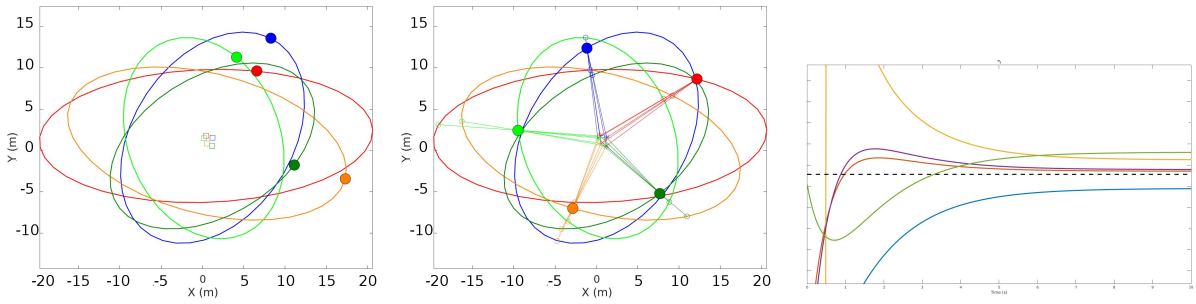
En este caso la dinámica de ψ_i queda expresada como:

$$\dot{\psi}_i = (\delta_{i+1} - \psi_i) \; mod \; 2\pi + (\delta_{i-1} - \psi_i) \; mod \; 2\pi. \tag{3.5}$$

En el siguiente cuadro, se presenta el esquema del bucle de control empleado para el atrapamiento.

> Obtener posición actual;
> Mientras posición final $\neq$ 1;
>> Obtener posición angular del agente anterior y el siguiente;
>> Comparar error angular con el anterior y el siguiente vecino;
>> Calcular punto de destino en la elipse;
>> Ejecutar acción;
>> Actualizar posición y proyecciónes en otras elipses;

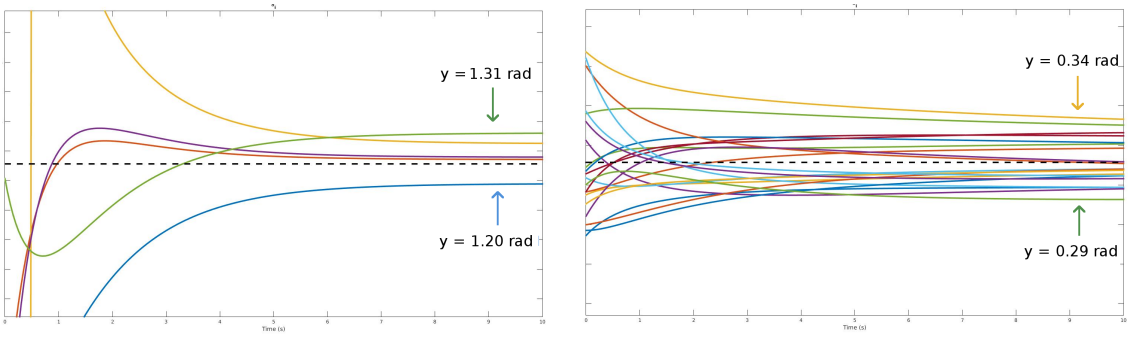
Las gráficas que derivan de la adición del problema de incertidumbre se muestran en la Figura (3.4).



(a) Posición inicial (b) Posición final y proyecciones (c) Distancia angular entre agentes

Figura 3.4: Algoritmo de atrapamiento con información global y local incierta.

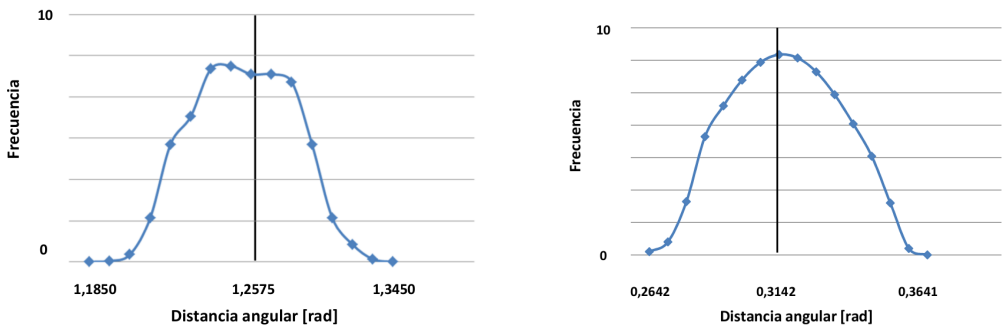
En este caso, existe un cierto error respecto de la posición ideal de equilibrio, debido a que la información que tienen los agentes del target no es totalmente exacta, por lo que la distancia angular $d_i \neq 2\pi/N$, pero el error en este caso es considerablemente pequeño. Se han realizado además simulaciones con un número mayor de robots ($N = 20$) para verificar que el error angular entre ellos disminuye al aumentar el número de robots.



(a) 5 robots (b) 20 robots

Figura 3.5: Distancia angular entre robots en experimentos con diferente número de robots.

La distancia angular mínima y máxima entre los robots de las simulaciones agregadas con 5 y 20 robots se muestra en la Figura 3.6.



(a) 5 robots. (b) 20 robots.

Figura 3.6: Distancia angular medida en simulaciones agregadas.

Como puede observarse la distancia angular tiene en ambos casos un comportamiento similar a una distribución normal. En el caso de las simulaciones con 5 robots, los valores extremos están proximos a 1,18 y 1,34 radianes y el valor central es de 1,2575, que corresponde a $2\pi/5$. Para las simulaciones con 20 robots el intervalo de valores de las distancias angulares es menor, con valores extremos proximos a 0,26 y 0,36 radianes, y el valor central es de 0,3641 que corresponde a $2\pi/20$.

3.4. Restricción de movimiento

Por último se han añadido restricciones en el movimiento de los robots, haciendo que éstos adquieran un comportamiento no holónomo. Debido a esta restricción los robots ya no sólo se desplazan por su elipse, sino que tienen que trazar trayectorias considerando puntos externos o internos a sus correspondientes elipses.

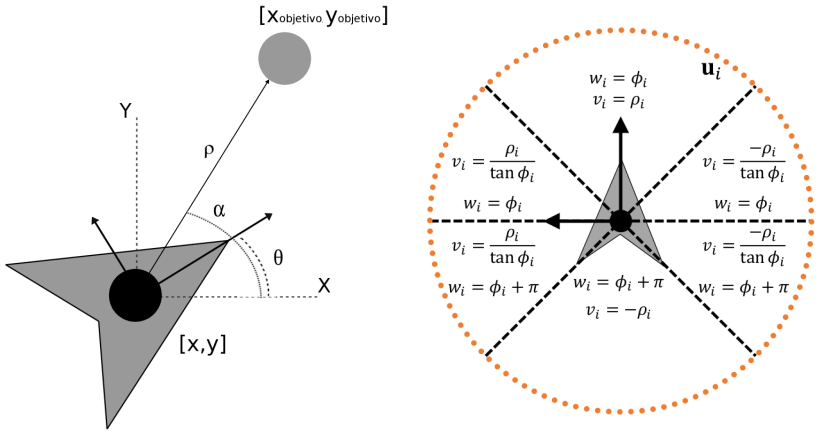
Como solución al problema del movimiento, se ha realizado la implementación de un control de formación distribuida para robots no hólonomos sin un marco de referencia global [2]. Esta forma de control introduce una nueva variable: la orientación relativa del robot θ. La velocidad linear y angular quedan en este caso definidas por la orientación del robot θ, su posición actual $[x, y]$ y la posición objetivo $[x_{objetivo}, y_{objetivo}]$.

A partir de estos parámetros es posible calcular la distancia entre el punto objetivo y el punto actual $\rho = \sqrt{(x_{objetivo} - x)^2 + (y_{objetivo} - y)^2}$, el ángulo que se establece entre dichos puntos $\alpha = \arctan 2(x_{objetivo} - x, y_{objetivo} - y)$, y la diferencia entre el ángulo del robot y el del objetivo $\phi = \widehat{\theta\alpha}$. Así, la velocidad linear (v) y la velocidad angular (w) de cada robot quedan establecidas según las siguientes reglas:

$$v = \begin{cases} \rho & \text{if } \frac{-\pi}{4} < \phi \leq \frac{\pi}{4} \\ \frac{\rho}{tan(\phi)} & \text{if } \frac{\pi}{4} < \phi \leq \frac{3\pi}{4} \\ \frac{-\rho}{tan(\phi)} & \text{if } \frac{-3\pi}{4} < \phi \leq \frac{-pi}{4} \\ -\rho & \text{otherwise} \end{cases}$$

$$w = \begin{cases} \phi & \text{if } \frac{-\pi}{2} < \phi \leq \frac{\pi}{2} \\ \phi + \pi & \text{otherwise} \end{cases}$$

En la Figura 3.7 quedan representados gráficamente todos estos parámetros.



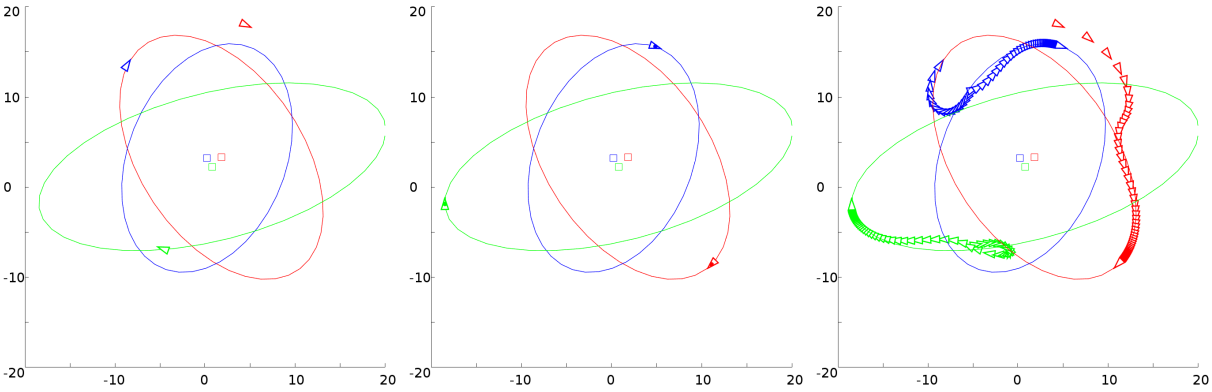
(a) Posición relativa y orientación (b) Velocidades no holónomas[2]

Figura 3.7: Parámetros del movimiento de robots no holónomos.

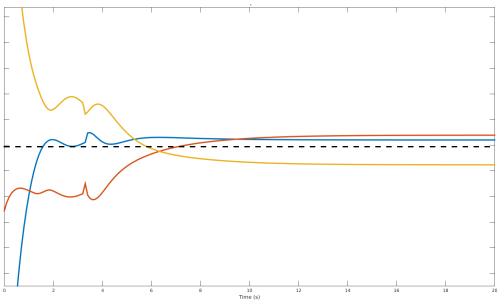
En el siguiente cuadro se presenta el esquema empleado para asignar la velocidad a los robots.

> Obtener orientación actual y posición del robot;
> Calcular el ángulo entre el robot y el objetivo = α ;
> Calcular la diferencia entre α y la orientación actual = ϕ ;
> Calcular la distancia entre el robot y el objetivo = ρ ;
> Comparar el valor obtenido de ϕ ;
> Asignar valores de v y w;

Las trayectorias que siguen los robots debido a estas restricciones en el movimiento se muestran en la Figura 3.8. En este caso se han utilizado 3 robots para que la representación de la trayectoria sea visible.



(a) Posición inicial (b) Posición final (c) Trayectoria realizada



(d) Distancia angular entre agentes

Figura 3.8: Algoritmo de atrapamiento con información global y local incierta, imponiendo restricción en el movimiento.

Al igual que en la sección anterior, los agentes encuentran un estado de equilibrio muy cercano a $d_i = 2\pi/N$ pero sigue existiendo un cierto error en la distancia angular. Además, se observa que la restricción del movimiento no supone una limitación para esta estrategia de atrapamiento, ya que los robots acaban alcanzando su posición objetivo. Estas simulaciones tambien se han realizado con un mayor número de robots para verificar el correcto funcionamiento del algoritmo con restricciones en el movimiento, obteniendo resultados satisfactorios.

3.5. Finalización del atrapamiento

Una vez que los robots se han distribuido uniformemente y han alcanzado la posición de equilibrio, se han planteado dos alternativas para finalizar el atrapamiento del target:

- En el caso de encontrarnos con un target estático, todos los robots modifican su posición angular (ϕ) para terminar orientados hacia el target, es decir, mirando hacia el centro de su elipse.

- La otra posibilidad es que cada robot sólo compare su posición angular con el robot inmediatamente siguiente. De esta forma la dinámica del sistema queda definida como $\dot{\psi}_i = (\psi_{i+1} - \psi_i)\ mod\ 2\pi$. Así se consigue que los robots roten en torno al target indefinidamente, trazando como trayectoria sus respectivas elipses. Esta alternativa resulta interesante cuando el target no es estático, ya que al rotar los robots en torno a él impiden que éste pueda salir, o que agentes externos accedan a él.

4. IMPLEMENTACIÓN

Este trabajo ha tenido como punto de inicio los experimentos previos realizados en este campo sobre el atrapamiento con un sistema multi-robot en el que se había considerado un grupo de N agentes holónomos[3].

El objetivo principal del proyecto que se presenta ha sido la implementación del algoritmo de atrapamiento en una plataforma real, utilizando para su realización Turtlebots (Figura 4.1).

Figura 4.1: Turtlebots utilizados en este trabajo.

Para ello una vez verificado el correcto funcionamiento del algoritmo, se ha procedido a simularlo en entornos realistas y finalmente se ha probado en robots reales en el laboratorio.

4.1. Gazebo

Gazebo es un simulador dinámico en 3D, con la capacidad de trabajar de forma precisa y eficiente con poblaciones de robots en ambientes complejos tanto interiores como exteriores. Además, ofrece la posibilidad de simular las físicas del entorno y de los robots con un alto nivel de fidelidad, y posee una gran variedad de sensores.

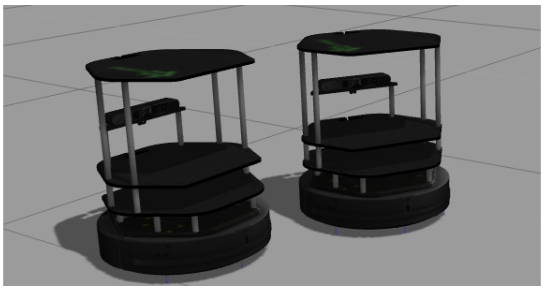
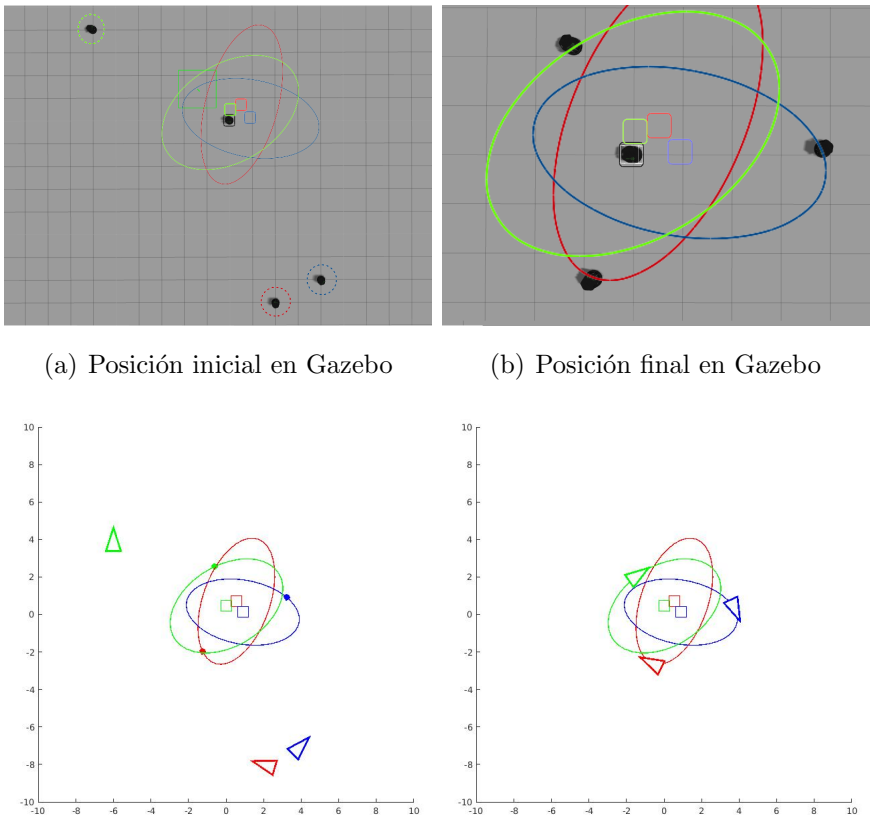


Figura 4.2: Turtlebots en Gazebo.

Para realizar las simulaciones también es necesario dotar a los robots de una interfaz, para lo cual se utiliza ROS (Robotic Operation System). ROS es un *framework* para el desarrollo de software para robots. Dada la facil conectividad que puede establecerse entre ROS y Matlab, se ha podido probar el algoritmo de atrapamiento en turtlebots simulados realizando las siguientes modificaciones en el código:

- Obtención de la posición y de la rotación de los robots.
- Obtención de la posición angular en el espacio de simulación de los robots.
- Envío de la velocidad linear y angular a los robots.

Como se puede apreciar en la Figura 4.3, los resultados de la simulación coinciden con los obtenidos en las simulaciones en Matlab. Así, los targets de cada robot, representados en cuadrados de color, se encuentran muy próximos al target real (cuadrado negro), pero no coinciden exactamente con éste debido al problema de incertidumbre.



(a) Posición inicial en Gazebo (b) Posición final en Gazebo

(c) Posición inicial en Matlab (d) Posición final en Matlab

Figura 4.3: Simulación del algoritmo de atrapamiento en Gazebo.

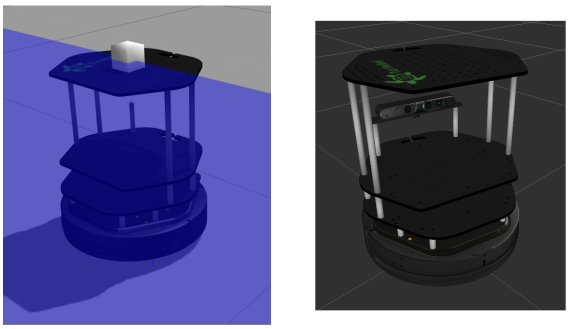
En esta Figura se muestra una simulación con 3 robots para aproximarlo a los experimentos reales, ya que solo se disponía de ese número de robots en el laboratorio.

En Gazebo somos capaces de obtener con exactitud la posición actual y la rotación de los Turtlebots sin tener que realizar ningún tipo de procedimiento o medida previa. Sin embargo, para trasladar este experimento a Turtlebots reales se ha necesitado cambiar el procedimiento por el que obteníamos esos dos parámetros.

La propuesta para resolver dicho problema ha sido incluir el uso de un algoritmo de localización. Para ello, primero se necesita un dispositivo capaz de medir distancias. Debido a que los Turtlebots disponibles en el laboratorio están equipados con láseres Hokuyo, se ha decidido implementar este láser en las simulaciones.

Esta tarea ha sido especialmente complicada debido a que hubo que modificar numerosos ficheros que rigen el comportamiento y las físicas de un Turtlebot en simulación.

Además, ha habido que modificar los ficheros que rigen la conectividad, sustituyendo el sensor de medida Kinect de Xbox360 que utiliza por defecto, por el láser Hokuyo.
Los resultados obtenidos en dichas modificaciones se muestran en la Figura 4.4, en la que se incluye un Turtlebot al que se le ha añadido un láser Hokuyo y eliminado la cámara Kinect (a) y un Turtlebot estándar (b).



(a) Turtlebot con láser Hokuyo

(b) Turtlebot standar

Figura 4.4: Modificaciones en los Turtlebots.

Cada Turtlebot puede incorporar un láser, permitiendo realizar múltiples medidas simultáneamente y visualizar la información de las lecturas en tiempo real, de forma que se pueden observar las distancias que los láseres están midiendo en cada momento (Figura 4.5).

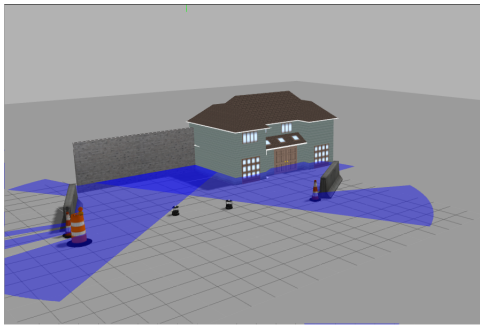


Figura 4.5: Turtlebots con láseres Hokuyo realizando mediciones.

Por último, los Turtebots estándar vienen incorporados con un elevado número de *topics* (variables sobre las que podemos escribir o leer información del robot). Estos tópicos se redujeron a los estrictamente necesarios para facilitar el manejo del robot.

4.2. Robots reales

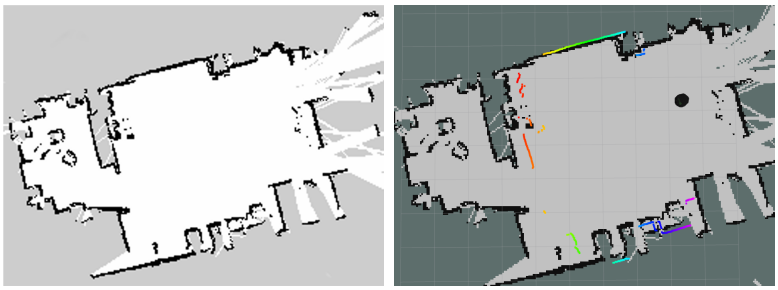
La última etapa de este trabajo ha sido realizar la implementación del algoritmo en Turtlebots reales. Previamente fue necesario adaptar el código, para lo que hubo que modificar la forma en que se recibía y enviaba la información. Además ha habido que modificar la forma de conectarse con el robot, que pasó de realizarse en un ordenador de forma local, a tener que utilizar una conexión a través de varias direcciones IP (una para cada Turtlebot).

Como se ha mencionado anteriormente era necesario obtener la posición y la orientación de los Turtlebots de una forma diferente a la que hacíamos en Gazebo.

Para realizar dicha tarea se ha utilizado el algoritmo de localización AMCL (Adaptative Monte Carlo Localization) y como aparato de medida un láser Hokuyo en cada Turtlebot. El procedimiento para poder trabajar con el AMCL en robots móviles incluye las siguientes etapas:

- Recorrer con un robot conectado a un aparato de medida (láser Hokuyo en este caso) todo el espacio de trabajo, de forma que tratando esta información, se puede generar un mapa del entorno.
- Dotar a cada robot de una posición y una orientación iniciales aproximadas dentro del mapa generado.
- Desde dicha posición, a traves de las medidas realizadas por el láser Hokuyo, el AMCL estima la posición y orientación real.

En la Figura 4.6 se puede observar el mapa creado para realizar los experimentos en el laboratorio de robótica del I3A (Instituto de Investigación en Ingeniería de Aragón), así como la información que un Turtlebot recién iniciado está leyendo del láser Hokuyo en tiempo real.



(a) Mapa del laboratorio (b) Medidas obtenidas en la inicialización del AMCL

Figura 4.6: Mapa y mediciones del laboratorio.

Como puede apreciarse, las medidas realizadas por el láser no coinciden exactamente con la posición de las paredes y objetos que componen el mapa. Esto es debido a que la posición y orientación que el Turtlebot tiene al iniciarse son unos valores introducidos por el usario de forma estimada. A pesar de este error, el AMCL localiza con una alta precisión al Turtlebot dentro del mapa.

Una vez que los Turtlebots tienen localizada su posición y orientación reales, se puede proceder a la realización de los experimentos del algoritmo de atrapamiento. Para ello se plantearon dos experimentos. En el primero se partió de una posición de los Turtlebot cercana al equilibrio (Figura 4.7), y en el segundo de una posición inicial desfavorable, encontrándose los Turtlebots alineados entre si y muy próximos entre ellos (Figura 4.8). En ambos casos y a pesar de las diferentes condiciones iniciales, los Turtlebots alcanzaron una posición de equilibrio formando un triángulo, es decir, separados angularmente entre ellos por una distancia de $2\pi/3$.

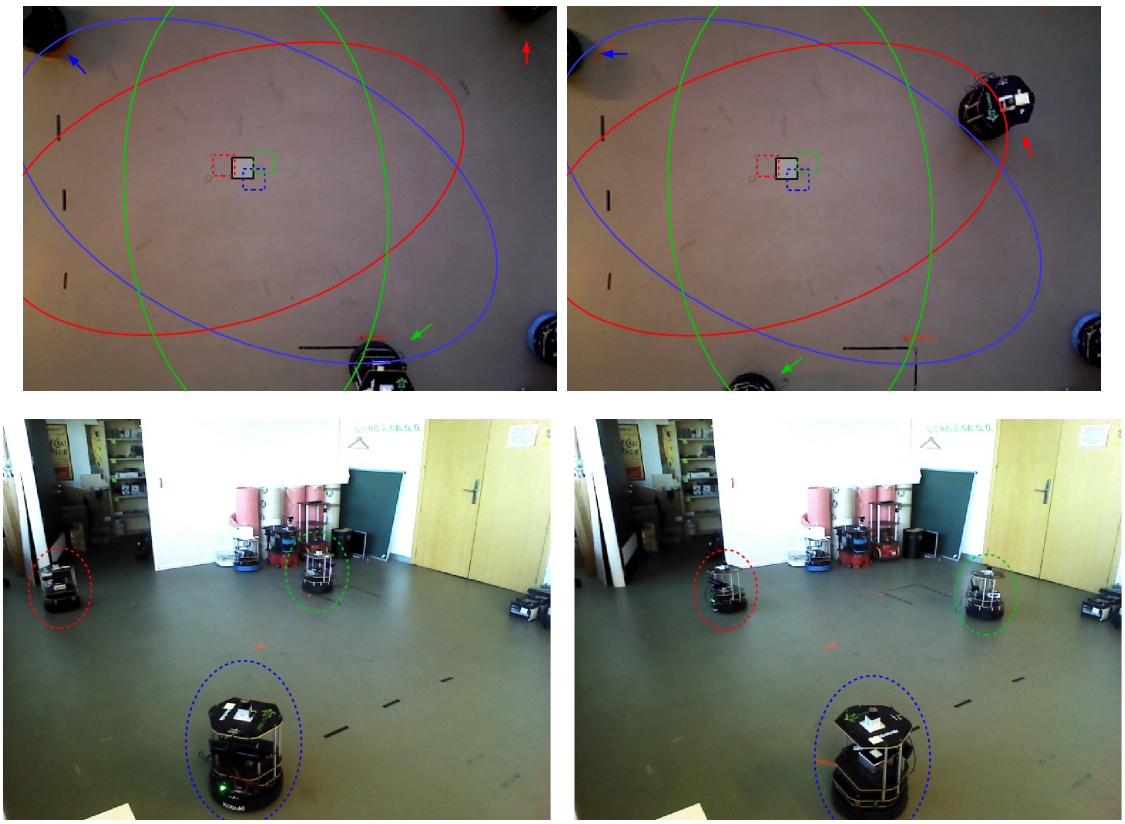


Figura 4.7: Primer experimento, posición inicial (izquierda), posición final (derecha).

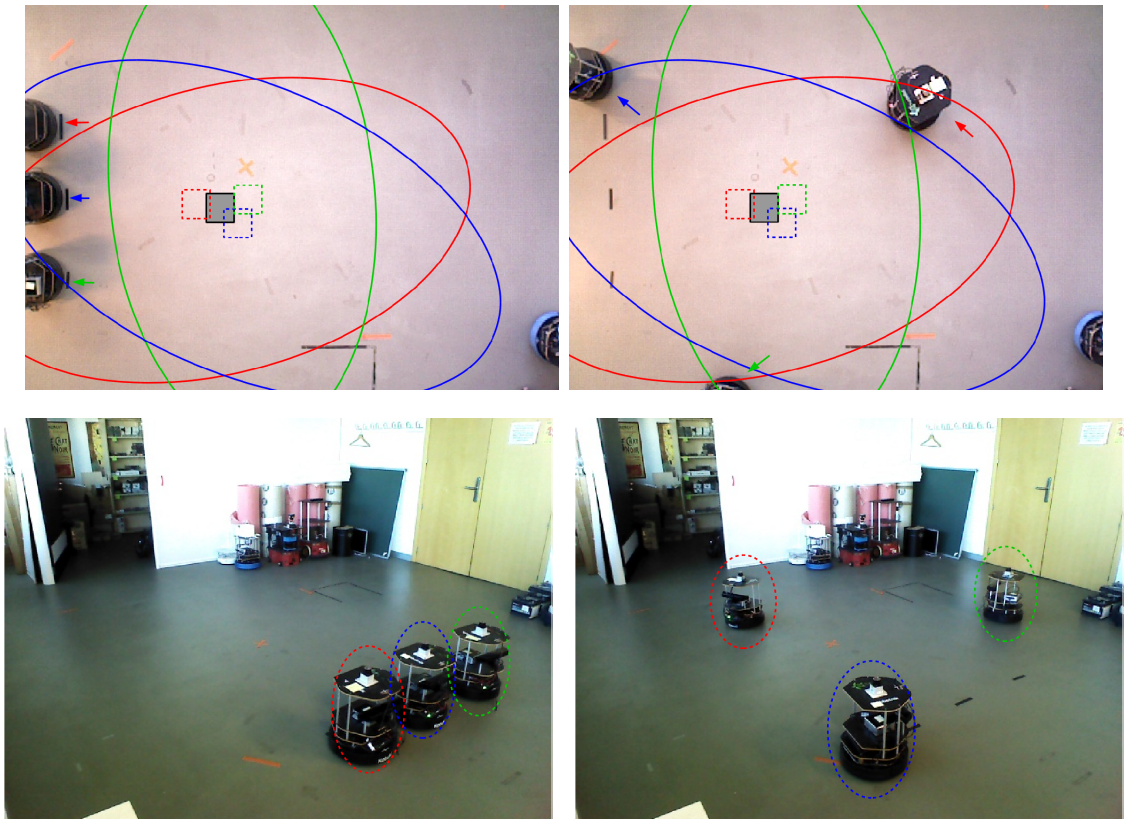


Figura 4.8: Segundo experimento, posición inicial (izquierda), posición final (derecha).

5. CONCLUSIONES

5.1. Valoración del trabajo realizado

En este proyecto se ha tratado el problema de implementar un algoritmo de atrapamiento en un sistema multi-robot, teniendo en cuenta las restricciones de movimiento de los robots, pasando de simulaciones virtuales a una situación con robots reales.

En primer lugar se ha estudiado, implementado y analizado el algoritmo de atrapamiento en Matlab. Se ha observado que a pesar de no tener información global y local precisa, generados por la falta de información de la posición del target, y de las restricciones en el movimiento propias de los robots no holónomos, el algoritmo de atrapamiento converge en una solución.

En segundo lugar se ha procedido a la implementación del algoritmo modificado en una plataforma de simulación más realista como es el caso de Gazebo. Las modificaciones realizadas en el algorítmo, fundamentalmente la forma de obtener y enviar la información, han facilitado el poder pasar de esta plataforma virtual a los robots reales sin tener que realizar grandes modificaciones a la hora de tratar la información.

Finalmente se han realizado experimentos en el laboratorio utilizando tres robots reales. En ellos se ha podido verificar el correcto funcionamiento del algoritmo de atrapamiento, teniendo en cuenta problemas de percepción e incertidumbre, y restricciones en el movimiento de los robots.

5.2. Líneas Futuras

Existen varios aspectos cuyo estudio permitiría avanzar y mejorar los resultados de este proyecto.

Se podrían considerar situaciones con targets móviles. En este trabajo se ha implementado la situación en la que una vez alcanzada la posición de equilibrio, los robots rotan en torno al target, impidiendo que este salga del atrapamiento. Sin embargo, sería de gran interes el contemplar una situación en la que los robots se desplazaran de forma simultanea con el target móvil.

En cuanto a la posición de cada robot en experimentos reales, en este trabajo se ha partido de la posición generada por el AMCL. A este respecto, sería de gran utilidad poder estimar de forma continua el error en las posiciones generadas para así conocer con mayor exactitud la posición real de cada robot.

Bibliografía

[1] Gianluca Antonelli, Filippo Arrichiello, y Stefano Chiaverini. The entrapment escorting mission. En *IEEE Robotics Automation Magazine*, págs. 22–29. 2008.

[2] Eduardo Montijano, Eric Cristofalo, Mac Schwager, y Carlos Sagues. Distributed formation control of non-holonomic robots without a global reference frame. En *Robotics and Automation (ICRA), IEEE International Conference*, págs. 5403–5408. 2016.

[3] Eduardo Montijano, Attilio Priolo, Andrea Gasparri, y Carlos Sagues. Distributed entrapment for multi-robot systems with uncertainties. En *IEEE 52nd Conference on Decision and Control*, págs. 1–7. 2013.

Índice de figuras