Máster en Investigación en Informática
Facultad de Informática
Universidad Complutense de Madrid

Proyecto Fin de Máster en Ingeniería Informática para la Industria

Planificación de Maniobras para Barcos Autónomos mediante Sistemas Bioinspirados

Autor:
José Mª Benítez Escario

Director:
Dr. Juan Jiménez Castellanos

Curso Académico: 2008-2009

El abajo firmante, matriculado en el Máster en Investigación en Informática de la Facultad de Informática, autoriza a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el presente Trabajo Fin de Máster: "Planificación de Maniobras para Barcos Autónomos mediante Sistemas Bioinspirados" , realizado durante el curso académico 2008-2009 bajo la dirección de Juan Jiménez Castellanos en el Departamento de Arquitectura de Computadores y Automática, y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo.

José Mª Benítez Escario

Abstract

This work presents some results achieved from a study on the use of an Ant Colony Algorithm Extension to plan feasible optimal or suboptimal trajectories for an autonomous ship manoeuvring. The scenario, for this work, comprises only open sea manoeuvres. The goal involves obtaining the least time consuming ship trajectory between to points, departing from the start point with arbitrary initial speed and attitude values and arriving to the end point with predefined speed and attitude values. The specific dynamic of the ship imposes typical restrictions to its manoeuvrability.

In recent years, several innovative optimisation techniques, based on heuristic search methods have been developed and proved in very different scenarios. Among them, the so called bioinspired algorithms, such the Ant Colony Optimisation or the Artificial Bee Colony Algorithm result particularly attractive by their capacity to solve complex optimisation problems in which, other classical techniques are unfeasible or difficult to implement. The aim of the present work is to prove the viability of one of these techniques to obtain the trajectory of an autonomous ship in the manoeuvring scenario described above. To accomplish this goal, a simplified dynamical model of a ship, considering only three degrees of freedom (surge, sway and yaw) was employed. The propulsion was modelled as a trimable waterjet system, which plays also the role of the rudder. Both, speed and course are controlled by a classical PID system, which stabilizes the course and speed, according to preset values of the PID constants.

Ant Colony Optimisation algorithms are based in the way in which ants are capable of finding the shortest path from a food source to their nest. Ants deposit a certain amount of pheromone while walking. When any ant searches a path to follow in its search for food, it prefers, in a probabilistic sense, the trails rich on pheromone. As far as shorter paths can be followed faster, the shorter the path the larger the number of ants that cover it by unit time. As a result, the shortest (optimum) path becomes more and richer on pheromone, and eventually it is the only one used.

Keywords— Ant Colony Optimization, Artificial Intelligence, Bio-inspired Algorithm, Motion Planning, Autonomous Ship Maneouvring

Resumen

Este trabajo presenta los resultados conseguidos mediante el uso de un algoritmo bio-inspirado, basado en el algoritmo Ant Colony Optimization, para la planificación y optimización de maniobras viables para un barco autónomo. El escenario de este trabajo incluye sólo maniobras en mar abierto. El objetivo consiste en obtener la trayectoria descrita por una barco entre dos puntos empleando el mínimo tiempo posible. Partiendo de un estado inicial con velocidad y orientación arbitraria, alcanzando el punto de destino con unos valores predeterminados para la velocidad y la orientación. La dinámica especifica del barco impone las correspondientes restricciones a su maniobrabilidad.

En los últimos años, se han desarrollado y probado en distintos escenarios técnicas de optimización heurísticas. Entre ellas los llamados algoritmos de inspiración biológica, como por ejemplo el Ant Colony Optimization o el Artificial Bee Colony Algorithm, resultan particularmente atractivos por su capacidad para resolver problemas de optimización complejos, en los cuales las técnicas clásicas no son aplicables o resultan difíciles de implementar.

El objetivo del presente trabajo es probar la viabilidad de una de estas técnicas para obtener la trayectoria de un barco autónomo en el escenario de maniobras previamente descrito. Para conseguir este objetivo, se utiliza un modelo simplificado de la dinámica de un barco, considerando únicamente tres grados de libertad (avante, guiñada y derivada). La propulsión se ha modelado utilizando un sistema de waterjets ajustable, que juega también el papel de timón. La velocidad y el rumbo se ajustan utilizando un controlador PID clásico que estabiliza el rumbo y la velocidad.

El algoritmo Ant Colony Optimization se basa en el método que utilizan las hormigas para encontrar los caminos más cortos en la recolección de alimento. A medida que avanzan las hormigas depositan una cierta cantidad de feromona, cuando una hormiga tiene que buscar el camino hacia la comida, prefiere, en un sentido probabilístico, aquellos caminos que presentan una mayor concentración de feromona. Como los caminos más cortos, son recorridos en menos tiempo, cuanto más corto es el camino mayor número de hormigas lo utilizan. El resultado es que el camino mínimo (óptimo) va aumentando la concentración de feromona y es el que finalmente se utiliza.

Palabras Clave— Ant Colony Optimization, Inteligencia Artificial, Algoritmos Bio-Inspirados, Planificación de Movimiento, Maniobra de Barcos Autónomos

Índice general

1. Introducción **5**

2. Estado del Arte **7**
2.1. Planificación 7
2.1.1. Algoritmos de Planificación 7
2.1.2. Planificación de movimiento 8
2.1.3. Problemas dinámicos y restricciones diferenciales 9
2.1.4. Planificación de movimiento basada en muestreo bajo restricciones diferenciales 10
2.2. El algoritmo 'Ant Colony Optimization' (ACO) 12
2.2.1. De hormigas reales a hormigas artificiales 12
2.2.2. Simple Ant Colony Optimization (S-ACO) 13
2.2.3. Detalles del S-ACO 15
2.3. Razonamiento Heurístico 16

3. Modelo dinámico en tres grados de libertad de un barco **17**
3.1. Ecuaciones del movimiento 17
3.2. El Control de rumbo y velocidad 19
3.3. Resultados del modelo dinámico 20
3.4. El problema de la no-holonomicidad, y las restricciones diferenciales 22
3.5. Descripción de un escenario de Maniobras 23

4. Método de resolución **26**
4.1. Algoritmo Discreto Para un espacio de estados continuo 26
4.1.1. Reducción del Espacio de estados de continuo a discreto infinito contable . . . 27
4.2. Construcción de la heurística 28
4.2.1. El autómata celular y la orientación del espacio de búsqueda 30
4.3. Ant Colony Extended (ACE): Extensión del algoritmo ACO 34
4.3.1. Introducción de la dinámica del problema 34
4.3.2. Hormigas exploradoras y hormigas recolectoras 36
4.3.3. Constitución y modificación de la tabla de feromonas 38
4.3.4. Arquitectura del sistema 42
4.3.5. Mínimos locales y selección colectiva 44
4.3.6. Algoritmos para el control de poblaciones 50

4.3.7. Auto-Organización de la búsqueda 52
4.3.8. Algoritmo Completo 53

5. Análisis de resultados **56**
5.1. Ejemplo de ejecución: Zonas independientes de búsqueda 56
5.2. Comparación con soluciones analíticas 61
5.3. Casos de estudio 64
5.3.1. Experimento aislado 64
5.3.2. Lote de experimentos 65
5.4. Discusión 82

Capítulo 1

Introducción

Este trabajo se enmarca en una línea de investigación conocida como métodos de optimización de inspiración biológica (Biologically-inspired Optimisation Methods), que se engloba dentro de lo que se conoce como Computación Evolutiva (Evolutionary Computation). El trabajo se centra en el desarrollo y aplicación de un sistema bio-inspirado a la planificación y optimización de las trayectorias de un barco autónomo cuando realiza maniobras.

La computación evolutiva consiste en el desarrollo de métodos para la resolución de problemas aplicando el esquema de trabajo propio de la evolución natural, ensayo y error. Este tipo de aproximación a la resolución de problemas tiene su ejemplo más clásico en los Algoritmos Genéticos [1] desarrollados por J. Holland en 1975. El esquema básico es partir de una población de soluciones, de las cuales se selecciona un subconjunto, a partir de ese subconjunto se aplican operadores de recombinación y mutación con el fin de generar una nueva población de soluciones.

Este tipo de métodos funcionan generando soluciones emergentes en contraposición a los métodos clásicos de búsqueda de soluciones, donde el cálculo de una solución se realiza explícitamente. Mediante los operadores de recombinación es posible generar nuevas soluciones a partir de las ya conocidas. Gracias a la selección de las mejores soluciones como base a esta recombinación se consigue que las nuevas soluciones mejoren a las anteriores, de tal modo que la población –el conjunto de soluciones– evoluciona. Hoy en día existen diferentes técnicas de optimización basadas en este esquema evolutivo, como puedan ser los Algoritmos Genéticos, Algoritmos Miméticos (Memetic Algorithms), Ant Colony Optimization,...

Esta aproximación se utiliza en diferentes campos de investigación como puedan ser vida artificial, simulación social y por supuesto optimización. La aplicación de esta técnica a la optimización es lo que se conoce como Biologically-inspired Optimisation Methods. El patrón de resolución ya no esta tan centrado en la evolución, sino que adapta alguna técnica de resolución presente en la naturaleza, el Artificial Bee Colony [2] por ejemplo, aplica la técnica de decisión adoptada por las abejas en un esquema algorítmico aplicado a la optimización numérica.

Este trabajo presenta los resultados conseguidos mediante el uso de un algoritmo bio-inspirado, basado en el algoritmo Ant Colony Optimization, para la planificación y optimización de maniobras viables para un barco autónomo. El escenario de este trabajo incluye sólo maniobras en mar abierto. El objetivo consiste en obtener la trayectoria descrita por una barco entre dos puntos empleando el mínimo tiempo posible. Partiendo de un estado inicial con velocidad y orientación arbitraria, alcanzando el

punto de destino con unos valores predeterminados para la velocidad y la orientación. La dinámica especifica del barco impone las correspondientes restricciones a su maniobrabilidad.

Para conseguir este objetivo, se utiliza un modelo simplificado de la dinámica de un barco, considerando únicamente tres grados de libertad (avante, guiñada y derivada). La propulsión se ha modelado utilizando un sistema de waterjets ajustable, que juega también el papel de timón. La velocidad y el rumbo se ajustan utilizando un controlador PID clásico que estabiliza el rumbo y la velocidad.

El algoritmo Ant Colony Optimization se basa en el método que utilizan las hormigas para encontrar los caminos más cortos en la recolección de alimento. A medida que avanzan las hormigas depositan una cierta cantidad de feromona, cuando una hormiga tiene que buscar el camino hacia la comida, prefiere, en un sentido probabilístico, aquellos caminos que presentan una mayor concentración de feromona. Como los caminos más cortos, son recorridos en menos tiempo, cuanto más corto es el camino mayor número de hormigas lo utilizan. El resultado es que el camino mínimo (óptimo) va aumentando la concentración de feromona y es el que finalmente se utiliza.

Partiendo de este algoritmo, construimos una extensión que permita aplicarlo a problemas en espacios de estados continuos. Esto incluye dotar a las hormigas de una dinámica que es la propia del barco. Por otro lado, a diferencia del algoritmo original, se emplean dos clases de hormigas. Por último se introduce un cambio de arquitectura y un control para las poblaciones de hormigas.

Los resultados obtenidos demuestran la viabilidad de esta aproximación, y abren la puerta a su aplicación para la planificación de otros tipos de sistemas de dinámica continua.

Capítulo 2

Estado del Arte

2.1. Planificación

2.1.1. Algoritmos de Planificación

Los desarrollos producidos en las últimas décadas en campos como la robótica, la inteligencia artificial o la teoría de control han despertado un interés creciente por los problemas de planificación. Siguiendo el excelente trabajo de revisión sobre algoritmos de planificación realizado por La Valle [3], podría decirse que en robótica la planificación estaba inicialmente relacionada con problemas tales como el movimiento de un piano dentro de una casa, de una habitación a otra, sin golpear contra ningún objeto. Actualmente, el tipo de problemas abordado ha crecido e incluye aspectos tales como la dinámica del problema, incertidumbres o la presencia de múltiples robots. En Inteligencia artificial, originalmente se entendía por planificación la búsqueda de una secuencia de operadores lógicos que transformaran un estado inicial en un estado final deseado. En la actualidad el problema se ha extendido e incluye ideas de teoría de la decisión, como cadenas de decisión de Markov, información imperfecta de estados, y teoría de juegos. La teoría de control tradicionalmente ha estado relacionada con problemas de estabilidad, realimentación y optimización, pero hay también un interés creciente en el diseño de algoritmos que permitan obtener trayectorias en lazo abierto para sistemas no lineales. A medida que se ha ido ampliando el área de interés en cada uno de los campos, se ha producido un intercambio de conocimientos y técnicas entre ellos, que permite establecer una base de conocimiento común.
Desde un punto de vista general, podemos descomponer un problema de planificación en los siguientes elementos:

- **Estados.** Un problema de planificación supone siempre la existencia de un espacio de estados, que configura todas las posibles situaciones en que pueda encontrarse el sistema objeto de estudio, por esta razón recibe también el nombre de espacio de configuración. Un estado puede venir representado por la posición en el espacio y la orientación de un móvil, las orientaciones de las articulaciones de un brazo robótico, o las posiciones de las fichas en un tablero de ajedrez. En general, según el sistema de que se trate o el grado de definición del mismo que sea preciso manejar, podemos encontrar espacios de estados discretos (finitos o infinitos) y espacios de estados continuos. En muchas aplicaciones el tamaño del espacio de estados es demasiado

grande para poder representarlo explícitamente. Pero en cualquier caso, supone un componente necesario para la formulación de un problema de planificación.

- **Tiempo.** Cualquier problema de planificación, supone una secuencia de decisiones que deben aplicarse en el tiempo. En un problema concreto, puede ser necesario incluirlo explícitamente, como sucede en los problemas de planificación de movimiento, o tan solo implícitamente, reflejando el hecho de que las acciones tomadas deben ser sucesivas. Como en el caso de los estados, el tiempo puede ser discreto o continuo, en este último caso, podemos considera la toma de decisiones como una función continua que evoluciona en el tiempo.

- **Acciones.** Un plan genera acciones que manipulan los estados. Desde un punto de vista de control es más frecuente hablar en términos de controladores y entradas al sistema. Cuando se formula un plan, es necesario conocer cómo las acciones modifican los estados. Habitualmente esto se expresa como una función de los estados para el caso de sistemas de tiempo discreto y mediante ecuaciones diferenciales para sistemas de tiempo continuo. En este último caso, es bastante frecuente en los problemas de planificación tratar de obtener una trayectoria en el espacio de estados integrando las ecuaciones diferenciales.

- **Estado inicial y estado objetivo.** Un problema de planificación supone la existencia de un estado inicial en que se encuentra el sistema y un estado o un conjunto de estados finales a los que se desea llevar el sistema.

- **Criterio.** Un problema de planificación lleva habitualmente asociado un criterio con el que se quiere hacer que el sistema evolucione desde el estado inicial hasta el estado objetivo. Generalmente, se distinguen dos tipos de criterios,

    - **Viabilidad.** Se busca un plan que permita llegar al estado objetivo, con independencia de la eficiencia de dicho plan. Es decir el único criterio es que el plan sea realizable.
    - **Optimalidad.** Se busca un plan viable que además de alcanzar el estado objetivo, sea óptimo desde el punto de vista de su desarrollo en algún sentido; minimizar el tiempo empleado, la energía consumida, etc.

2.1.2. Planificación de movimiento

Se entiende por planificación de movimiento en general la planificación en espacios de estado continuos. Por otro lado, y en el ámbito de la robótica y del movimiento de vehículos autónomos se entiende la planificación del movimiento de dichos objetos dentro del mundo 2D ó 3D, con posibles obstáculos, en el que se encuentran inmersos. No es difícil entender que la segunda acepción está relacionada con la primera, puesto que el espacio real en que se mueven los vehículos puede constituir parte de su espacio de configuraciones, o al menos será posible encontrar una ley de transformación más o menos complicada entre ambos espacios. La dimensión completa del espacio de estados coincide con el número de grados de libertad que tenga el vehículo o robot. Por supuesto, para este tipo de problemas no es posible encontrar una definición explícita de los estados del sistema puesto que se trata de un espacio de estados infinito e incontable. En este contexto, un plan puede describirse como una trayectoria continua en el espacio de configuraciones.

Una práctica habitual a la hora de tratar con problemas de planificación de movimiento es discretizar el problema. Se construye un modelo discreto a partir del sistema continuo y se planifica sobre el modelo discreto. Así por ejemplo para el caso de la búsqueda de una trayectoria del movimiento de un vehículo se descompone ésta en puntos de paso, cada uno de ellos se considera como un estado discreto, y se planifican las trayectorias entre estados discretos. Este método simplifica la búsqueda de soluciones viables, siempre y cuando los puntos de paso elegidos como estados discretos sean alcanzables, y exista algún modo de obtener las acciones, en general funciones de los estados, que definen la transición de un estado a otro.

Como se indicó en el apartado anterior, desde el punto de vista de la planificación es posible buscar soluciones óptimas o soluciones simplemente viables. En mucho problemas de planificación de movimiento, la búsqueda de soluciones viables es ya de por sí complicada. En cualquier caso, la discretización del problema que acabamos de describir, puede hacer que se simplifique dicha búsqueda en la medida en que obtener una trayectoria entre los nuevos estados discretos sea sencillo. Si éste es el caso, la dificultad se traslada a la discretización del espacio de búsqueda. Cuando se está buscando una solución óptima el problema se complica puesto que analíticamente supone el cálculo de un extremal para la función de coste que se pretende minimizar. En general, el camino a seguir suele ser entonces la búsqueda entre las posibles soluciones en el espacio de configuración. Esta búsqueda puede ser sistemática: se prueban todos los casos posibles hasta dar con el óptimo. Es lo que se conoce como planificación combinatorial de movimiento, y por razones obvias generalmente solo es aplicable en situaciones concretas donde el número de posibles combinaciones es limitado. La segunda vía es la que se conoce en términos generales como planificación de movimiento basada en muestreo. En este caso no se exploran todas las posibilidades sistemáticamente. Se prueban algunas de ellas, es decir se muestrea el espacio de soluciones, y se busca un método heurístico que permita converger hacia la solución óptima a partir de las muestras analizadas. Por último indicar que ambos métodos pueden también emplearse para obtener soluciones viables cuando no es sencillo o posible obtener directamente la trayectoria que une dos estados consecutivos.

Los tipos de planificación descritos hasta ahora, dan lugar a lo que se conoce como planificación en lazo abierto. Es decir, no hay ningún mecanismo que permita controlar los resultados del plan, una vez puesto en marcha y corregir posibles desviaciones. Un paso más en el estudio de los métodos de planificación lo constituye lo que se conoce como planificación de movimiento realimentada. En este caso, se estudia la evolución del sistema en el espacio de estados como fruto de la planificación, y en función de dicha evolución se modifica o reconstruye el plan elaborado. Este tipo de planificación es necesaria en presencia de incertidumbres.

2.1.3. Problemas dinámicos y restricciones diferenciales

Cuando el sistema sobre el que se pretende realizar una planificación es de tiempo continuo y viene descrito por un espacio de estados continuo, como es el caso del movimiento de un vehículo en dos o tres dimensiones, la transición de estados es también un proceso continuo y viene descrito por la 'velocidad' de transición de estados, que en general será una función del estado y de la acción aplicada en cada instante de tiempo,

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \tag{2.1}$$

Por tanto, la función de transición de estados suministra un vector velocidad en el espacio de estados en lugar de suministrar un nuevo estado. Los estados futuros se obtienen por integración de la velocidad. Resulta por tanto natural especificar el sistema en términos de velocidades. Es interesante notar que la ecuación 2.1, impone restricciones a los valores que puede tomar la velocidad. Este tipo de restricciones se conocen en general con el nombre de restricciones diferenciales y pueden entonces especificar restricciones físicas como por ejemplo el hecho de que un vehículo con ruedas no puede desplazarse lateralmente.

En problemas de planificación de movimiento suele ser necesario incluir en la descripción del sistema su dinámica. Por ejemplo si consideramos un vehículo avanzando a su velocidad máxima hacia una pared, no es razonable empezar a frenar cuando está a un milímetro de ella, porque será imposible evitar la colisión. Es preciso entonces introducir las aceleraciones y las restricciones a que estas están sometidas. Para ello suele expandirse el espacio de estados de modo que incluya las velocidades, se llega así al concepto de espacio de fases, en que cualquier problema se representa imponiendo restricciones solo sobre velocidades. Se reduce el orden de las aceleraciones o, en su caso, el de las derivadas de orden superior hasta primer orden, a cambio de aumentar la dimensión del espacio de estados. De este modo, las restricciones sobre las aceleraciones pueden considerarse también como restricciones diferenciales.

2.1.4. Planificación de movimiento basada en muestreo bajo restricciones diferenciales

El problema de la planificación de movimiento bajo restricciones diferenciales puede enunciarse de acuerdo con La Valle [3] en los siguientes términos,

1. Un 'mundo' $\mathcal{W}$, un robot (o en general un vehículo no tripulado) $\mathcal{A}$, una región obstáculo $\mathcal{O}$, y un espacio de configuración $\mathcal{C}$.

2. Un intervalo de tiempo continuo $T = [0, \infty)$

3. Un espacio de estados $\mathcal{X}$ que puede coincidir con $\mathcal{C}$, $\mathcal{X} = \mathcal{C}$ o puede tratarse de un espacio de fases si incluye la dinámica del problema. Una función κ que asocia cada punto $q \in \mathcal{C}$ con un punto de $\mathbf{x} \in \mathcal{X}$, $\mathbf{q} = \kappa(\mathbf{x})$

4. Una región obstáculo $\mathcal{X}_{obs}$ en el espacio de estados. Si $\mathcal{X} = \mathcal{C}$ entonces $\mathcal{X}_{obs} = \mathcal{O}$. La notación $\mathcal{X}_f = \mathcal{X} \setminus \mathcal{X}_{obs}$ indican los estados accesibles, aquellos que están libres de obstáculos u otras restricciones globales.

5. Para cada estado $\mathbf{x} \in \mathcal{X}$, se define un espacio de acción acotado $U(\mathbf{x})$, de dimensión m igual al número de variables de acción o entradas del sistema. El espacio de acción incluye una acción terminal $\mathbf{u}_T$. Cuando se aplica la acción terminal se asume que $f(\mathbf{x}, \mathbf{u}_T) = 0$. Si el problema emplea algún tipo de función de coste, la acción final no acumula coste adicional.

6. Un sistema especificado mediante una ecuación de transición de estados $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$, definido para todo $\mathbf{x} \in \mathcal{X}$ y $\mathbf{u} \in U(\mathbf{x})$.

7. Un estado $\mathbf{x_I} \in \mathcal{X}_f$ que se designa como estado inicial.

8. Un conjunto $\mathcal{X}_G \subset \mathcal{X}_f$ que se designa como la región objetivo.

9. Un algoritmo de planificación completo debería calcular una trayectoria $\tilde{\mathbf{u}}$ en el espacio de acción para la cual la trayectoria en el espacio de estados definida por,

$$\tilde{\mathbf{x}} \equiv \mathbf{x}(t) = \mathbf{x}(0) + \int_0^t f(\mathbf{x}(t'), \mathbf{u}(t'))dt' \tag{2.2}$$

satisface que $\mathbf{x}(0) = \mathbf{x}_\mathbf{I}$, y que existe algún valor del tiempo t para el cual $\mathbf{u}(t) = \mathbf{u}_\mathbf{T}$ y $\mathbf{x} \in \mathcal{X}_G$

Es posible añadir restricciones adicionales, como por ejemplo que $\tilde{\mathbf{u}}$ sea continua o diferenciable. Si además se pretende buscar una solución óptima, es preciso añadir la correspondiente función de coste.

Una vez definido el problema, no es difícil comprender que en general, la obtención analítica de una trayectoria en el espacio de acciones, es decir un plan continuo, es en la mayoría de los casos una tarea inabordable. Los métodos de planificación basados en el muestreo permiten sin embargo encontrar soluciones realizables y aproximarse a una solución óptima. Son muchas las técnicas posibles de abordar el problema, en general es preciso muestrear en dos espacios continuos, el de estados , sometido a las restricciones diferenciales impuestas por las ecuaciones del sistemas, y el de acciones. En la literatura, es posible encontrar métodos muy diversos de muestreo de soluciones, de nuevo, el trabajo de La Valle [3] ofrece una excelente revisión de ellos. En lo que sigue nos centraremos en las ideas generales del método empleado en este trabajo.

En primer lugar, dado que solo son alcanzables por el sistema aquellos puntos del espacio de estados obtenidos aplicando trayectorias de acción, es preferible no muestrear directamente en el espacio $\mathcal{X}$. Solo tiene sentido trabajar con aquellos estados que son alcanzables mediante la integración de trayectorias de acción eq. 2.2. El conjunto de dichos estado recibe el nombre de conjunto alcanzable. Por tanto, en primera aproximación solo se muestreará dentro de estados alcanzables.

Una segunda condición del muestreo razonable es imponer un límite máximo de tiempo. En principio, el intervalo de tiempo definido se extiende hasta infinito, pero para un problema de planificación concreto es suficiente elegir un t_{max} adecuado al problema de modo que solo se muestrea el conjunto de estados alcanzables en el tiempo límite t_{max}.

Una tercera condición impuesta al muestreo consiste en discretizar la trayectoria seguida en el espacio de acciones U de este modo, la trayectoria $\tilde{\mathbf{u}}$ se muestrea a intervalos de tiempo Δt, no necesariamente iguales. En otras palabras, la acción permanece constante durante los periodos Δt. Con la acción discretizada, es posible realizar un árbol de 'alcanzabilidad'; partiendo de estado inicial, se aplica un primer valor de la acción durante Δt y se obtiene la trayectoria resultante en el espacio de estados. De este modo se llega a un nuevo estado, al que se aplica un nuevo valor de la acción, y así sucesivamente. En el presente trabajo, el tiempo de cálculo de la acción se ha discretizado indirectamente. La discretización directa se ha realizado dividiendo en celdas cuadradas el espacio 2D donde se mueve el barco, y permitiendo una nueva acción tan solo cuando se cambia de celda.

Es evidente que el hecho de muestrear los espacios de estados y de acción, puede no ser suficiente para abordar el problema directamente. Una búsqueda exhaustiva en los nuevos espacios muestreados es en la mayoría de los casos implanteable. Surge así el atractivo de los métodos heurísticos, para resolver este tipo de problemas de planificación.

2.2. El algoritmo 'Ant Colony Optimization' (ACO)

Ant Colony Optimization [4, 5] es un algoritmo de optimización propuesto por Marco Dorigo en 1992, basándose en el comportamiento que exhiben las colonias de hormigas en la búsqueda de alimento, describe un marco formal para la resolución de problemas más complejos.

2.2.1. De hormigas reales a hormigas artificiales

La percepción visual de muchas especies de hormigas es muy reducida. De hecho, investigaciones recientes acerca del comportamiento de las hormigas revelan que la mayor parte de la comunicación que se realiza entre individuos o entre individuos y su entorno, esta basada en el uso de sustancias químicas producidas por las hormigas. Dichas sustancias son llamadas feromonas.

Los "senderos de feromonas", particularmente importante para ciertos tipo de hormigas, están constituidos por un tipo especial de feromonas que se usan para marcar caminos en el suelo. Guiadas por los restos de feromonas depositadas en el suelo las hormigas recolectoras pueden seguir el camino hacia a la fuente de alimento descubierta por otras hormigas anteriormente. Este comportamiento colectivo de creación de senderos y seguimiento de senderos es lo que inspira el desarrollo del algoritmo Ant Colony Optimization (ACO).

El experimento del doble puente: este experimento (ver figura 2.1) fue desarrollado por Goss, Aron, Deneubourg y Pasteels [6], los cuales usaron un doble puente para conectar una colonia de hormigas de la especie argentina I. humilis a una fuente de comida. Realizaron los experimentos variando el ratio $r = \frac{l_1}{l_2}$ que es la longitud entre los dos ramales del puente. Siendo l_1 es la longitud del mas largo y l_2 la del mas corto.

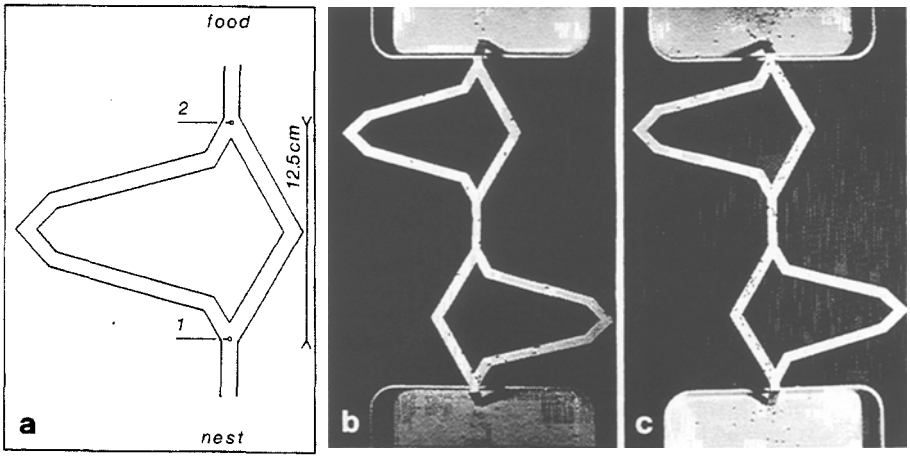


Fig. 1. A colony of *I. humilis* selecting the short branches on both modules of the bridge; a) one module of the bridge, b) and c): photos taken 4 and 8 min after placement of the bridge

Figura 2.1: Experimento del doble puente. Imágenes reproducidas del articulo "Self-organized shortcuts in the Argentine ant" [6]

De los experimentos que realizaron dos son de gran interés:

1. En el primer experimento, un ramal es mas largo que el otro y ambos están disponibles desde el principio para las hormigas. El resultado al cabo de un tiempo fue que el ramal corto fue escogido preferente.

2. En el segundo experimento solo se le presentó el ramal largo y pasado un tiempo se introdujo el corto. El resultado fue que el ramal largo fue el escogido frente al corto.

La explicación del primer experimento es la siguiente: el ramal mas corto será recorrido mas rápidamente, si una hormiga deposita feromona en cada viaje, el corto tendrá una mayor concentración. Por tanto, cuando una hormiga tiene que tomar la decisión acerca de que camino seguir, el corto presentará una mayor carga de feromona inclinando la decisión por este último y al cabo de un tiempo la gran mayoría de hormigas circulará por el ramal mas corto.

Por el contrario en el segundo experimento sólo hay un camino disponible, el cual tendrá toda la concentración de feromona. Cuando se introduce el nuevo ramal más corto, éste no tiene ninguna marca porque no ha sido recorrido ninguna vez, por lo que en el momento de decidir que camino tomar la concentración de feromona disponible en el largo frente a la concentración del corto inclinará la decisión en favor del primero.

La conclusión que se obtiene de ambos experimentos es la siguiente: la decisión tomada, en conjunto por toda la colonia, acerca de que camino es el mejor para alcanzar la comida lleva tiempo y es tomada una única vez, por supuesto hasta que las hormigas vuelven al hormiguero y las marcas depositadas en el suelo se evaporan. Por tanto, sólo deciden entre los elementos que tengan disponibles. En el segundo experimento el ramal corto se puso a disposición de la colonia una vez escogido el camino, por lo que no se tomó en cuenta. A modo de introducción, este problema que deja en evidencia el experimento del doble puente será uno de los mayores retos del ACO y es lo que se soluciona utilizando la denominada Evaporación de Feromona Artificial.

2.2.2. Simple Ant Colony Optimization (S-ACO)

A partir del experimento del doble puente, Dorigo crea un algoritmo recreando el comportamiento de las hormigas para encontrar el camino mínimo en un grafo. Comienza considerando un grafo estático G=(N,A) donde N es el conjunto de vértices del grafo y A es el conjunto de aristas no dirigidas que los conecta. Los dos puntos entre los que se quiere establecer el camino mínimo los llama fuente y destino o análogamente con las hormigas reales hormiguero y fuente de alimento.

El primer problema es la aparición de ciclos mientras las hormigas construyen la solución, un ciclo aparece cuando una hormiga alcanza un nodo del grafo visitado previamente. Como consecuencia del mecanismo de actualización de feromona, a medida que la hormiga avanza los ciclos tenderían a ser más y más atrayentes para las hormigas y estas quedarían atrapadas indefinidamente. Incluso si las hormigas pudiesen escapar de los ciclos, la distribución global de feromona ya no reflejaría la ruta mínima entre la fuente y el destino. Ya que este problema es debido a la actualización de feromona a medida que avanza la hormiga, parece razonable eliminar la actualización de feromona en el avance, pero si se suprime este mecanismo el sistema deja de funcionar, incluso para el caso mas simple del doble puente.

Esta dependencia sobre el marcado de feromona a medida que la hormiga avanza se explica atendiendo a las hormigas reales. Como se señaló anteriormente, la orientación básica de una hormiga son los rastros dejados en el suelo, cuando encuentra una fuente de comida debe volver a la colonia, pero si no ha marcado su rastro con feromona no es capaz de volver a la colonia, porque *no recuerda* el camino que ha seguido para alcanzar la fuente de comida. Es la misma situación que el cuento infantil de Hansel y Gretel, tienen que marcar el camino para salir del bosque a medida que avanzan por él, para que cuando quieran volver sean capaces de identificar el camino.

Dorigo soluciona esta situación extendiendo las capacidades básicas de las hormigas artificiales, las provee de una memoria primitiva capaz de almacenar la ruta que han seguido, así como el coste de la misma. Gracias a esta memoria las hormigas artificiales son capaces de implementar una serie de comportamientos que les permiten construir eficientemente una solución:

1. Construcción probabilista de la solución basada en rutas de feromonas, sin mecanismo de actualización de feromona a medida que la hormiga avanza.

2. Recorrido inverso determinista con eliminación de ciclos a la vez que se actualizan las rutas de feromonas.

3. Evaluación de la calidad de la solución generada, según dicha evaluación se determina la cantidad de feromona que deberá ser depositada.

Construcción probabilista de la solución. Las hormigas del S-ACO pueden ser concebidas con dos modos de funcionamiento: hacia delante y hacia atrás (a partir de ahora por similitud con los textos ingleses nos referiremos a ellas como forward y backward). Se encuentran en modo forward cuando se están moviendo desde el hormiguero hacia la fuente de comida y en modo backward cuando regresan de la fuente hacia el nido.

Cuando una hormiga en modo forward alcanza la fuente de comida, cambia su modo de comportamiento a modo backward y comienza su viaje de regreso hacia la colonia. En S-ACO las hormigas forward construyen una solución escogiendo de manera probabilista el nodo siguiente al que moverse entre los vecinos del nodo donde se encuentren actualmente. Dado un grafo G= (N,A), dos nodos $i, j \in N$ son vecinos si existe un arco(i,j) $\in$ A que los conecte, la elección probabilista esta ponderada mediante la feromona depositada previamente por otras hormigas en ese arco. Las hormigas en modo forward no depositan ninguna cantidad de feromona mientras se mueven.

Recorrido inverso determinista. El uso de una memoria explícita permita a una hormiga volver por el camino que ha le ha llevado hasta la fuente de comida. S-ACO mejora el rendimiento de las hormigas, eliminado los ciclos que se encuentren en el camino que les haya llevado a la solución. Mientras se mueven hacia atrás actualizan la feromona presente en los arcos que recorren.

Actualización de feromona basada en la calidad de la solución. En S-ACO la hormiga memoriza el camino que le ha llevado hasta la solución a la vez que el coste de los arcos que ha recorrido. De tal modo que pueden evaluar el coste de la solución obtenida y utilizar dicho coste para modular la cantidad de feromona que depositan en cada arco que forma el camino recorrido.

La cantidad de feromona a depositar puede ser fija ó variable según el coste del camino, así se consigue que se deposite más cantidad en los caminos más cortos, de tal manera se logra que la búsqueda de caminos converja más rápido hacia las mejores soluciones.

Evaporación de Feromona. En las hormigas reales la intensidad de la feromona presente en el medio decrece en función del tiempo. En el S-ACO dicha evaporación es simulada aplicando una regla de reducción de feromona.

La evaporación de feromona reduce la influencia del marcado en las primeras iteraciones, cuando las hormigas construyen las peores soluciones. Aunque este mecanismo no influye en las hormigas reales, la evaporación es muy lenta comparada con el movimiento de las hormigas, en los algoritmos basados en hormigas juega un papel muy importante.

2.2.3. Detalles del S-ACO

Búsqueda de caminos: cada hormiga construye, comenzando desde el nodo fuente una solución al problema aplicando una política de decisión local. En cada nodo, la información que este almacena acerca de la zona que le es más próxima (sus nodos vecinos), es percibida por la hormiga y usada de una manera estocástica para decidir que nodo debe visitar a continuación.

Al comienzo de cada proceso de búsqueda, una cantidad constante de feromona es asignada a todos los arcos (aristas). Cuando la hormiga k se encuentra en el nodo i, usa la marca de feromona τ_{ij} para calcular la probabilidad de escoger el nodo j de la siguiente manera :

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha}{\sum_{l \in N_i^k} \tau_{ij}^\alpha} & \text{si} \quad j \in N_i^k \\ 0 & \text{si} \quad j \notin N_i^k \end{cases}$$

donde N_i^k es la lista de nodos disponibles para la hormiga k
cuando se encuentra en el nodo i

En S-ACO la vecindad de un nodo contiene a todos los nodos que conectan directamente con dicho nodo, exceptuando aquel nodo del que procede la hormiga. De esta manera se evita que las hormigas vuelvan sobre el nodo inmediatamente visitado. Solo en el caso en que la vecindad de un nodo sea el conjunto vacío se permite a la hormiga volver sobre sus pasos.

Un hormiga salta repetidamente de nodo a nodo usando la política de decisión previamente descrita hasta que alcanza el nodo destino. Es claro que debido a las diferencias entre los caminos que siguen unas hormigas u otras, existirán diferencias en el tiempo que tardan en alcanzar el nodo objetivo, las que vayan por los caminos mas cortos finalizaran antes.

Trazado de rutas y actualización de la feromona: cuando una hormiga alcanza su destino cambia de comportamiento, de forward a backward, y comienza a reconstruir el camino que ha hecho en sentido inverso paso a paso. Antes de comenzar el regreso a la fuente, la hormiga elimina los ciclos que pueda haber en el camino que ha construido mientras buscaba el nodo objetivo. El problema de los ciclos es que pueden recibir cargas de feromonas varias veces mientras la hormiga realiza su viaje de retorno a la fuente, generando un fenómeno de auto-reforzamiento de los ciclos frente a los caminos no cíclicos.

Mientras la hormiga regresa deposita una cantidad fija de feromona en los arcos que ha visitado. En particular, si una hormiga k en modo backward atraviesa el arco (i,j) cambia la cantidad de feromona que marca al arco de la siguiente manera:

$$\tau_{ij} \leftarrow \tau_{ij} + \triangle\tau^k$$

Con esta regla una hormiga que use el arco que conecta j e i incrementa la probabilidad de que el resto de las hormigas usen ese arco en el futuro.

La elección de la cantidad de feromona depositada se realiza en función de la calidad de la solución que la hormiga ha encontrado. En particular el caso más simple es que sea una cantidad constante.

Evaporación de las marcas de feromona: la evaporación de feromona puede ser vista como un mecanismo que evita la rápida convergencia de todas las hormigas hacia una ruta subóptima. De hecho la reducción de la cantidad de feromona que se encuentra en los caminos favorece la exploración de nuevas rutas durante el proceso de búsqueda global.

2.3. Razonamiento Heurístico

Polya [7] define el razonamiento heurístico como:

> **Heuristic reasoning** is reasoning not regarded as final and strict but as provisional and plausible only, whose purpose is to discover the solution of the present problem. We are often obliged to use heuristic reasoning. We shall attain complete certainty when we shall have obtained the complete solution, but before obtaining certainty we must often be satisfied with a more or less plausible guess. We may need the provisional before we attain the final.

Un razonamiento heurístico es un método sobre el cual el razonamiento se construye apoyándose en un aproximación previa, que si bien no es solución, tiene el propósito de servir de base para alcanzar la solución real. En un sistema algorítmico significa que tenemos dos partes, una primera que genera hipótesis de soluciones, la segunda parte construye a partir de estas hipótesis la solución real.

La forma de encontrar soluciones al problema sigue el esquema descrito por Polya, se dota a las hormigas de un componente heurístico a la hora de seleccionar el camino a seguir cuando no existen marcas de feromonas. De tal manera que si bien las hormigas no son algoritmos heurísticos, nuestra forma de resolver el problema si lo es. Las heurísticas sirven fundamentalmente para que la generación de soluciones reales se consiga en un tiempo razonable.

En el tipo de problemas planteado, incluso después de realizar una discretización como la sugerida en 2.1.4, el espacio de búsqueda sigue siendo muy extenso. Una búsqueda de tipo combinatorial se hace inviable, pero también lo son las estocásticas equiprobables. Por lo que se hace imprescindible un criterio para guiar el recorrido en el espacio de acciones, esta función es la que desempeña la heurística.

Capítulo 3

Modelo dinámico en tres grados de libertad de un barco

3.1. Ecuaciones del movimiento

Esta sección se centra en la descripción de los aspectos básicos utilizados para modelar los barcos. El sistema empleado, considera los movimientos del barco exclusivamente en el plano, por tanto, solo se consideran los movimientos en las direcciones avante (surge), deriva (sway) y guiñada (yaw). La figura 3.1 muestra una vista esquemática del sistema de coordenadas empleado en la descripción del barco y su dinámica.

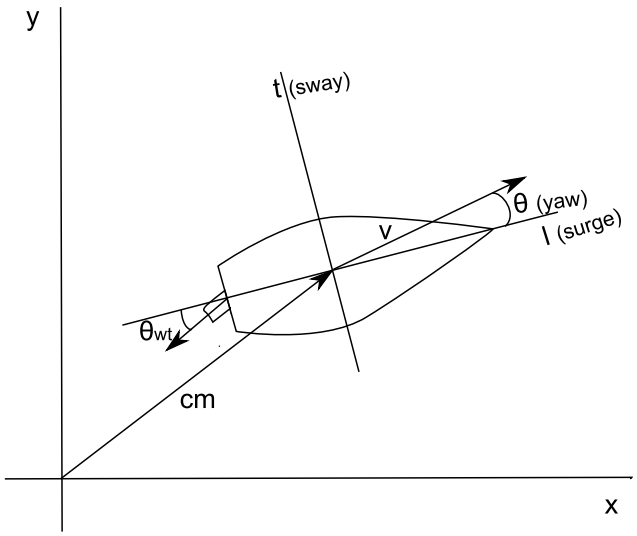


Figura 3.1: Esquema general de coordenadas empleadas

El sistema de propulsión empleado para el barco es de tipo *waterjet*: se considera que el barco tiene a popa un propulsor a chorro que puede además orientarse, con lo que dicho propulsor forma también parte del sistema de gobierno del barco, actuando como timón. El movimiento del barco se

descompone empleando los ejes fijos al barco l (longitudinal) y t (transversal), introducidos en la figura 3.1. Además, se considera la rotación del barco en torno al eje vertical que pasa por su centro de masas.

De acuerdo con dicha descripción, la aceleración en el eje l vendrá dada por:

$$a_l = \frac{F_m \cdot \cos\theta_{wj} - \mu_{ll} \cdot v_l - \mu_{lc} \cdot v_l \cdot |v_l|}{m_b + m_{al}} \tag{3.1}$$

Los tres términos de la ecuación anterior se corresponden con la componente l del empuje de los *waterjets*, la componente lineal (μ_{ll}) de la resistencia del barco al avance y la componente cuadrática (μ_{lc}) de la resistencia del barco al avance. m_b representa la masa del barco y m_{al} representa la masa añadida debida al desplazamiento de agua provocado por el avance del barco. En primera aproximación se considera proporcional a v_l; $m_{al} = r_{mal} \cdot v_l$. El ángulo θ_{wj} representa el ángulo formado por los *waterjets* con la dirección longitudinal l del barco. Análogamente, para la componente transversal de la aceleración se obtiene:

$$a_t = \frac{F_m \cdot \operatorname{sen}\theta_{wj} - \mu_{tl} \cdot v_t - \mu_{tc} \cdot v_t \cdot |v_t|}{m_b + m_{at}} \tag{3.2}$$

Por último, la rotación del barco alrededor de su eje vertical central, se estudia, calculando el momento generado por la fuerza F_m respecto al centro del barco:

$$M_d = \frac{l_s}{2} \cdot F_m \cdot \operatorname{sen}\theta_{wj} \tag{3.3}$$

La aceleración angular viene dada entonces por:

$$\alpha_b = \frac{M_d - \mu_{al} \cdot l_s \cdot \omega_b - \mu_{ac} \cdot l_s \cdot \omega_b \cdot |\omega_b|}{I_b + I_{ba}} \tag{3.4}$$

Donde μ_{al} representa la resistencia lineal al giro del barco, μ_{ac} la resistencia al giro cuadrática, ω_b la velocidad angular, I_b el momento de inercia e I_{ba} el momento de inercia añadido.

El movimiento del barco viene determinado por los valores de Fm, que representa la fuerza impulsora de los waterjets y de θ_{wj} que representa su orientación. Es preciso para completar el modelo fijar una valores máximos, F_{max} y θ_{wtmax}, de dichos parámetros. En total, es preciso determinar un conjunto de 12 parámetros para completar la descripción del modelo. En el presente trabajo, se han elegido unos valores arbitrarios, buscando exclusivamente obtener un comportamiento razonable de la dinámica del barco. Para ajustar el modelo a un barco real sería preciso, estimar sus valores u obtenerlos experimentalmente. La tabla 3.1 recoge los valores de los parámetros empleados en el presente trabajo.

En general, es posible encontrar en la literatura [8] modelos de barcos más sofisticados, que describen el movimiento del barco en seis grados de libertad, sin embargo, cuando se trata de describir maniobras es práctica habitual restringir la descripción a los tres grados de libertad empleados en el presente modelo.

Cuadro 3.1: Comparación en tiempo(s) con la solución analítica para diferentes distancias(m)

	Parámetro	símbolo	valor
1	Eslora	l_s	$4m$
2	Masa	m_b	$1000kg$
3	Fuerza máxima Waterjets	F_{max}	$2500N$
4	Ángulo máximo de giro del waterjet	θ_{wtmax}	$45grados$
5	Componente lineal de la resistencia al avance	μ_{ll}	$7N.s/m$
8	Componente cuadrática de la resistencia al avance	μ_{lc}	$10N.s^2/m^2$
9	Componente lineal de la resistencia al desplazamiento lateral	μ_{tl}	$1000N.s/m$
10	Componente cuadrática de la resistencia al desplazamiento lateral	μ_{tc}	$10000N.s^2/m^2$
11	Coeficiente lineal de las masas añadidas en la dirección l	r_{mal}	$0{,}5kg.s/m$
12	Coeficiente lineal de la masas añadidas en la dirección t	r_{mat}	$10Kg.s/m$
13	Momento de inercia	I_b	$kg.m$
14	Coeficiente lineal del momento de inercia añadido	r_{Iba}	1 kg.m.s/rad
15	Coeficiente lineal de la resistencia al giro	$\mu_a l$	$10N.m.s/rad$
16	Coeficiente cuadrático de la resistencia al giro	$\mu_a c$	$100N.m.s^2/rad^2$

3.2. El Control de rumbo y velocidad

Tanto la modificación del rumbo del barco como la de su velocidad, se realizan actuando sobre la fuerza y la orientación de los *waterjets*. Ambas variables están acopladas entre sí. Si se giran los *waterjets*, automáticamente disminuye la fuerza realizada en la dirección de avance del barco, con lo que éste tiende a perder velocidad. Por otro lado, el aumento o disminución de la fuerza impulsora aumenta o disminuye la velocidad de giro del barco y por tanto altera también su radio de giro.

En el presente trabajo se pretende trabajar con un barco autónomo, es decir, el rumbo y la velocidad del barco son controlados de modo automático sin que sea precisa la intervención humana. Para el control de rumbo y velocidad, se han cerrado unos lazos de control sobre los valores de la fuerza impulsora y ángulo de los *waterjets*, El movimiento de los *waterjets* se ha modelado como un sistema de primer orden con una constante de tiempo $\tau = 0{,}1s$. Se ha elegido un controlador tipo PD para el rumbo y un controlador tipo PI para la velocidad. La acción de control se ha desacoplado de modo que el error de rumbo solo actúa sobre la orientación de los waterjets, y el error de velocidad sobre la fuerza impulsora. Las ecuaciones de los dos controladores serían las siguientes:

$$F_m = kv_p \cdot (rv_l - v_l) + kv_i \int (rv_d - v_l)dt \tag{3.5}$$

$$\theta_{wjc} = kr_p \cdot (r_{rumbo} - \theta) + kr_d \frac{d(r_{rumbo} - \theta)}{dt} \tag{3.6}$$

Donde kv_p representa la constante proporcional del controlador de velocidad, kv_i su constante integral y rv_l la consigna de velocidad para el barco. Análogamente para el control de rumbo, kr_p es

la constante proporcional del controlador, kr_d la constante derivativa y r_{rumbo} la consigna de rumbo del barco. Las constantes de los controladores, se han asignado empíricamente al modelo para que este de una respuesta razonable, teniendo en cuenta que tanto la actuación en velocidad como en rumbo están limitadas. A efectos del algoritmo desarrollado para la planificación de trayectorias, el modelo de barco y su control descrito aquí se reciben como un sistema dado, sobre el que actuar. Es decir, el modelo de barco y su control se ha creado para suministrar un escenario al ACO, pero su optimización no es un objetivo del presente trabajo.

3.3. Resultados del modelo dinámico

Una vez obtenidas las ecuaciones de la dinámica del barco, se integraron empleando la función ode45 incluida en el entorno de Matlab. Dicha función emplea un par encajado de Runge-Kutta de órdenes cuarto y quinto, desarrollado por Dormand y Prince [9], para resolver sistemas de ecuaciones diferenciales. Para su empleo, la función ode45 exige la descripción de problema en términos de variables de estado, que para el modelo vendrían determinadas por los valores de su posición, orientación y los valores de las respectivas velocidades. La función ode45 admite el empleo de masas dependientes de las variables de estado, lo que permite trabajar con las masas añadidas del modelo descrito en la sección 3.1. Además permite la inclusión de funciones de eventos para finalizar la integración de las ecuaciones.

Para probar las características del modelo se realizó una batería completa de ensayos, combinado valores de consigna para el rumbo desde 0 a 180 grados, tomados de grado en grado y velocidades desde 1 m/s hasta 15 m/s con incrementos de 1 m/s. El barco parte siempre del reposo y está inicialmente orientado formando 0 grados con el eje x, que se toma como eje de referencia para medir los rumbos. El tiempo de simulación es de 50 s. En todos los casos, se obtuvieron tiempos razonables para la estabilización tanto del rumbo como de la velocidad del barco, aunque en los valores de consigna de velocidad extremos, se alcanza la saturación del actuador.

La figura 3.2 muestra los resultados obtenidos para rumbo 0 grados y los quince valores de la velocidad consignados. En este caso, el rumbo inicial coincide con el de consigna con lo que no hay acción de control sobre el rumbo. El gráfico de velocidad de avance muestra como ésta alcanza su valor de consigna entre los 20 y 30 s desde el inicio del movimiento.

La figura 3.3 Muestra un cambio de rumbo de 90 grados. De nuevo el barco parte del reposo y se han ensayado los mismos quince valores de consigna para la velocidad que en el ejemplo anterior. En el gráfico superior izquierdo de la figura puede observarse como el barco ha conseguido estabilizar su rumbo en todos los casos a partir de los 7 segundos de iniciar la maniobra. El gráfico superior derecho muestra las curvas descritas por el barco, para los distintos valores de velocidad de consigna. Es interesante observar como el radio de giro va creciendo paulatinamente a la vez que se incrementa la velocidad. El gráfico inferior izquierdo muestra la orientación de los *waterjets*. Se observa la saturación de éstos y el brusco cambio de orientación necesario para equilibrar el rumbo. El gráfico inferior derecho muestra la evolución de la velocidad del barco, los tiempos de estabilización son parecidos a los de caso anterior.

La figura 3.4 muestra un último ejemplo para un cambio de rumbo de 180 grados. De nuevo es interesante ver como el rumbo alcanza su valor de consigna en uno 8 segundos, así como la dispersión de las trayectorias en función de la velocidad de consigna.

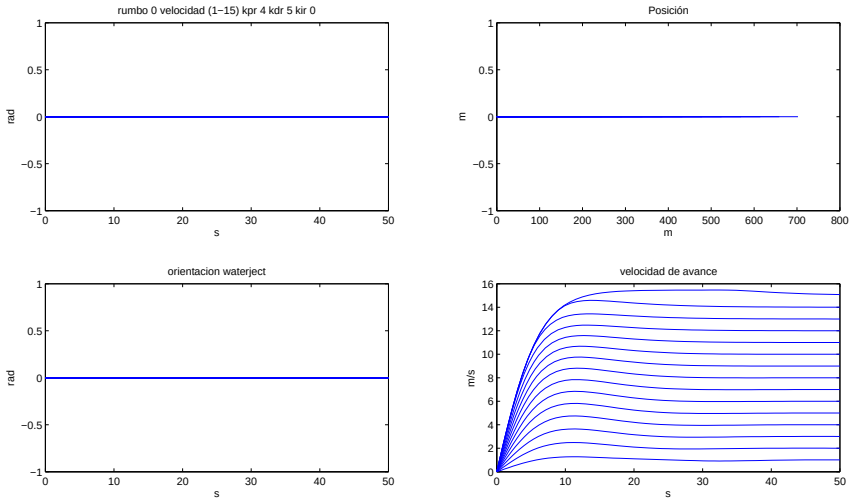


Figura 3.2: Resultados obtenidos por el modelo con consigna de rumbo 0 grados para velocidades entre 0 y 15 m/s

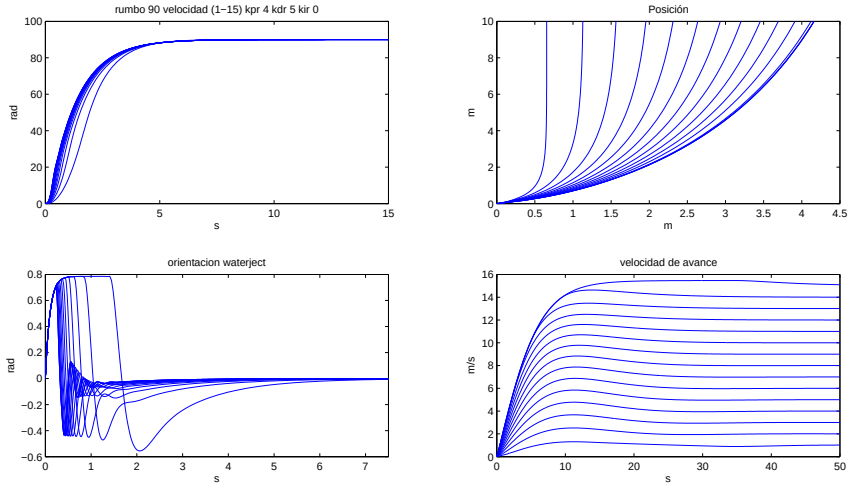


Figura 3.3: Resultados obtenidos por el modelo con consigna de rumbo 90 grados para velocidades entre 0 y 15 m/s

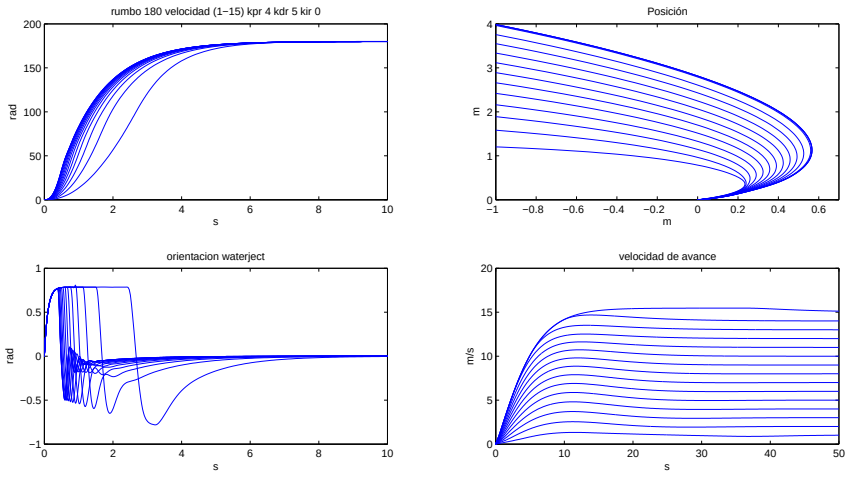


Figura 3.4: Resultados obtenidos por el modelo con consigna de rumbo 180 grados para velocidades entre 0 y 15 m/s

3.4. El problema de la no-holonomicidad, y las restricciones diferenciales

El objetivo del modelo desarrollado, es emplearlo para el estudio de la planificación de trayectorias. Es interesante en este punto analizar las características dinámicas del barco, ya que estas se van a incluir directamente en el planificador. Para realizar este análisis es conveniente describir el problema en términos de variables de estado. Tomaremos como variables de estado las posiciones x e y del barco respecto a un sistema de referencia en reposo así como su orientación θ (rumbo) respecto a dicho sistema de referencia. A partir de las ecuaciones del movimiento estas pueden obtenerse como,

$$\dot{x} = v_l cos(\theta) - v_t sen(\theta) \tag{3.7}$$

$$\dot{y} = v_t cos(\theta) + v_l sen(\theta) \tag{3.8}$$

$$\dot{\theta} = \omega_b \tag{3.9}$$

Además se incluyen como variables de estado las velocidades lineales v_l y v_t y la velocidad angular del barco,

$$\dot{v_l} = \frac{F_m \cdot \cos\theta_{wj} - \mu_{ll} \cdot v_l - \mu_{lc} \cdot v_l \cdot |v_l|}{m_b + m_{al}} \tag{3.10}$$

$$\dot{v_t} = \frac{F_m \cdot \operatorname{sen}\theta_{wj} - \mu_{tl} \cdot v_t - \mu_{tc} \cdot v_t \cdot |v_t|}{m_b + m_{at}} \tag{3.11}$$

$$\dot{\omega_b} = \frac{M_d - \mu_{al} \cdot l_s \cdot \omega_b - \mu_{ac} \cdot l_s \cdot \omega_b \cdot |\omega_b|}{I_b + I_{ba}} \tag{3.12}$$

y por último la orientación de los *waterjets*

$$\dot{\theta}_{wj} = \frac{\theta_{wjc} - \theta_{wj}}{\tau} \tag{3.13}$$

Tenemos por tanto un conjunto de siete variables de estado que conforman el espacio de configuraciones del barco. Su estado podría describirse como un vector de siete componentes en el espacio de fases$E(x, y, \theta, v_l, v_t, w_b, \theta_{wj})$.

El movimiento del barco puede describirse como una trayectoria sobre dicho espacio de configuraciones. Las ecuaciones de estado que acabamos de describir imponen dos tipos de restricciones al movimiento del barco.

La primera se debe a la no-holonomicidad del sistema: dado un estado del barco en el espacio de configuraciones, en general no es posible pasar a otro estado arbitrario siguiendo una línea recta. La no-holonomicidad refleja el hecho de que el barco no puede desplazarse directamente en cualquier dirección y que no puede girar sobre sí mismo, porque dichos desplazamiento son incompatibles con sus ecuaciones de estado.

La segunda restricción se debe al carácter dinámico de las ecuaciones de estado, el barco no puede cambiar su estado de movimiento instantáneamente o de manera arbitraria. Cualquier giro supone trazar una trayectoria cuya curvatura está definida por su estado en cada instante de tiempo. Por último, es preciso tener en cuenta las restricciones causadas por el sistema de actuadores. Los *waterjets* pueden producir una fuerza impulsora limitada, y no se ha previsto en ellos la posibilidad de frenar el barco. Además el ángulo de giro de los mismos y la velocidad en posicionarse, están también limitados. Este segundo tipo de restricciones se encuadra dentro de las llamadas restricciones diferenciales.

Desde el punto de vista de la planificación de maniobras, una maniobra es realizable si es posible encontrar una función $[F_m, \theta_{wtc}] = f(E)$ para los valores de consigna de los actuadores, que permita obtener una curva integral en el espacio de estados que lleve desde un estado inicial E_i hasta un estado final E_f. En un segundo paso, una maniobra sería optima si además de ser realizable, minimiza una determinada función de coste: El tiempo total, la distancia recorrida, el consumo energético, etc.

Como ya se ha explicado anteriormente, es frecuente en los algoritmos de planificación proponer una trayectoria tentativa, con unos puntos de paso. Para comprobar después si el sistema es capaz de seguirla –aunque sea solo de modo aproximado– con la condición de que el estado final sea alcanzable. La idea en el presente trabajo es que el propio planificador tome en cuenta toda las restricciones, y encuentre la función de control $f(E)$ que hace posible la maniobra.

3.5. Descripción de un escenario de Maniobras

El escenario de maniobras propuesto para el presente trabajo, incluye solo maniobras en mar abierto, es decir, no se ha previsto la existencia de obstáculos al movimiento del barco.

Si consideramos las maniobras descritas en el contexto del espacio de estados del sistema, podemos describirla como el paso de un estado inicial $E_i = [x(t_i), y(t_i), v_l(t_i), v_t(t_i), w_b(t_i), \theta(t_i)]$ A un estado final $E_f = [x(t_f), y(t_f), v_l(t_f), v_t(t_f), w_b(t_f), \theta(t_f)]$. De este modo, la maniobra quedaría completamente determinada dando las posiciones, orientaciones y velocidades iniciales y finales del barco. La transición entre el estado inicial y el final debe de hacerse respetando las restricciones diferenciales impuestas

por la dinámica del barco, es decir debe ser una trayectoria posible de acuerdo con las ecuaciones que describen la evolución del estado del barco. En principio el número de trayectorias en el espacio de estados del barco que conectan dos puntos sería infinito, aunque no es siempre trivial encontrar los valores de consigna en rumbo y velocidad que nos aseguren encontrar al menos una de dichas trayectorias. Dado un escenario podrían establecerse dos objetivos, el primero sería encontrar trayectorias posibles que permitan completar la maniobra, el segundo sería encontrar de entre ellas una que sea óptima en algún sentido. En el presente trabajo se aborda la búsqueda de trayectorias que minimicen el tiempo empleado en recorrerlas, es decir el objetivo es minimizar la función $\Delta t = t_f - t_i$.

Desde el punto de vista tanto de la búsqueda de las trayectorias posibles, como de la búsqueda de la trayectoria óptima, el problema pierde sentido si la distancia que separa la posición inicial de la final del barco es suficientemente grande. En este caso bastaría descomponer el movimiento completo del barco, en una maniobra de partida –una trayectoria en línea recta hacia la posición objetivo– y una maniobra final de aproximación hacia el objetivo. El concepto de maniobra cobra realmente sentido cuando se desarrolla en distancias cortas, porque la maniobra exige cambios de rumbo de cierta complejidad. En esto casos, será en los que se centre este trabajo. Algunos ejemplos de maniobras se describen a continuación, empleando para ello los valores de los estados inicial y final de cada maniobra.

- Virado en redondo sobre la dirección de avante: Se trata de hacer un viraje de 180 grados sobre la propia trayectoria del barco. Un ejemplo posible de esta maniobra sería,

$$E_i = [0m, 0m, 0m/s, 0m/s, 0rad/s, 90^o] \rightharpoonup E_f = [0m, 20m, 4m/s, 0m/s, 270^o] \tag{3.14}$$

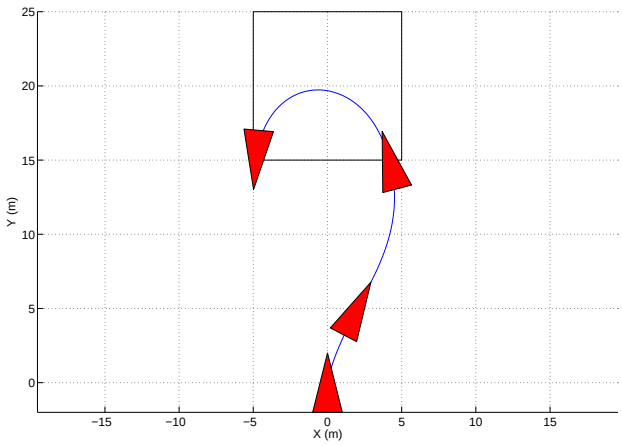


Figura 3.5: Ejemplo de maniobras: virado en redondo sobre la dirección de avante.

- Desplazamiento lateral: el barco debe alcanzar una posición a babor o estribor de su posición inicial, con el mismo rumbo con que inició la maniobra. Por ejemplo,

$$E_i = [0m, 0m, 0m/s, 0m/s, 0rad/s, 90^o] \rightharpoonup E_f = [20m, 0m, 15m/s, 0m/s, 90^o] \tag{3.15}$$

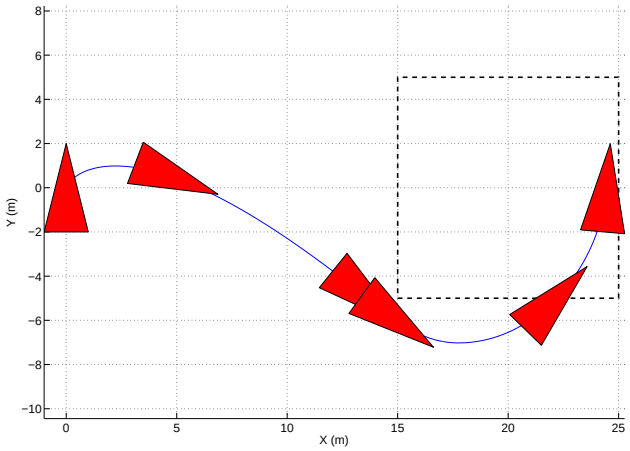


Figura 3.6: Ejemplo de maniobras: desplazamiento lateral.

- Desplazamiento en la dirección de popa: se trataría de situarse a popa de la posición inicial, conservando al final de la maniobra el rumbo inicial.

$$E_i = [0m, 0m, 0m/s, 0m/s, 0rad/s, 90^o] \rightharpoonup E_f = [0m, -20m, 15m/s, 0m/s, 90^o] \tag{3.16}$$

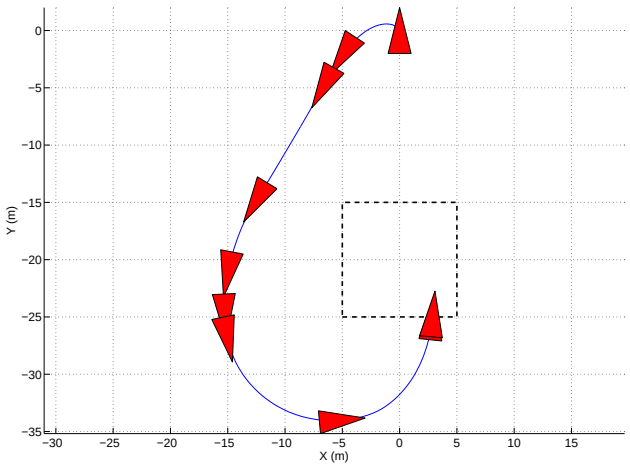


Figura 3.7: Ejemplo de maniobras: desplazamiento en la dirección de popa.

Las maniobras descritas tienen el interés de exigir grandes cambios de rumbo en distancias relativamente cortas, a la vez que ponen de manifiesto el carácter no holonómico de barco y el efecto de sus restricciones diferenciales. Es evidente que la lista de escenarios posibles puede alargarse indefinidamente. Los descritos son sólo algunos ejemplos cuya resolución se describe en los siguientes capítulos.

Capítulo 4

Método de resolución

4.1. Algoritmo Discreto Para un espacio de estados continuo

La generación de una trayectoria depende de los parámetros de control fijados, velocidad y rumbo. Por tanto, una trayectoria concreta estará asociada a una secuencia de decisiones para el valor del rumba y la velocidad de avance. Sobre el resto de variables de estado no se dispone de ningún control directo.

El cálculo de trayectorias se obtiene resolviendo numéricamente el modelo diferencial. Este cálculo se basa en aproximar mediante un paso de integración –fijo o variable– el valor real –continuo– del modelo.

De acuerdo con lo señalado en la sección 2.1.4 se discretiza la trayectoria, seguida en el espacio de acciones (U), de modo que los valores de las entradas se mantienen constantes en intervalos de tiempo Δt. El planificador emplea el modelo descrito en la sección 3.4 de la siguiente manera:

1. Se asignan unos valores determinados a las entradas –acciones–, se resuelve numéricamente el modelo y se obtiene un estado del sistema.
2. Se asignan nuevos valores a las entradas, según el estado del sistema calculado previamente.
3. Se resuelve nuevamente el modelo según las nuevas consignas y tomando como estado de partida el estado del sistema obtenido previamente.

La optimización consiste en calcular la secuencia de consignas discretas asociadas a la trayectoria cuyo recorrido exhiba el tiempo mínimo.

Todo el cálculo de integración se realiza durante el periodo de tiempo (Δt) , dicho periodo es lo que llamamos periodo de vigencia o simplemente vigencia de las decisiones. Es decir, el periodo de validez de los valores escogidos para las entradas o acciones, cuando este periodo expira se tomará –si es necesario– una nueva decisión.

Establecer un criterio de vigencia es establecer una discretización. Debido a lo mencionado anteriormente acerca de la integración, al permanecer fijas las consignas durante el periodo de integración se restringe el espacio de acciones. Como consecuencia existen estados del sistema que no son alcanzables, por ejemplo, aquellos que necesitarían de un cambio de consignas durante el periodo de integración. Al restringir el espacio de acciones, se restringe el conjunto de trayectorias posibles.

En nuestro caso nos planteamos dos posibles criterios de vigencia, uno temporal y otro espacial.

Vigencia temporal: consiste en fijar un tiempo de integración constante para la resolución del modelo. De este modo se van tomando sucesivas decisiones a intervalos constantes de tiempo. Este criterio no es muy adecuado, fundamentalmente por la poca flexibilidad que permite. Al estar manejado velocidades y fijando un tiempo de integración constante, la longitud de la secuencia de direcciones será dependiente de la velocidad. A velocidades altas la secuencia de acciones será corta, reduciendo la controlabilidad del barco. A velocidades bajas la secuencia será larga, dificultando el cálculo de trayectorias óptimas.

Vigencia espacial: consiste en discretizar el espacio mediante una malla y asociar la validez de cada decisión con una región del espacio. De este modo se van tomando sucesivas decisiones a medida que se van atravesando distintas regiones del espacio. El periodo de integración (Δt) será variable. En función de las consignas y el estado del sistema, será necesario más o menos tiempo para atravesar una región. Este criterio presenta una mayor flexibilidad que el criterio temporal. Porque la discretización depende únicamente del tamaño de la región escogida, por el contrario en el criterio temporal depende de la discretización temporal y la velocidad.

El criterio de discretización utilizado es el espacial, que consiste en dividir el espacio según una malla con un tamaño de celda de 10x10m. El tamaño escogido esta en proporción con la longitud del barco(4m). A su vez existe otra restricción, las consignas de dirección y velocidad están discretizadas. Las primeras varían de 0º a 359º con 1º de libertad, mientras que las segundas de 1 a 14 m/s, con 1 m/s de libertad.

4.1.1. Reducción del Espacio de estados de continuo a discreto infinito contable

Por lo mencionado anteriormente acerca del criterio de vigencia, se necesita discretizar el espacio. Ahora bien, el barco puede estar en distintas posiciones espaciales, las cuales están representadas mediante variables de estado que toman valores sobre el conjunto de los número reales. La discretización no se lleva a cabo sobre las posiciones del barco, éstas estarán restringidas únicamente por la dinámica. Se trata por tanto de proyectar la posición real del barco sobre un espacio discreto, la posición en este espacio discreto marcará el periodo de validez de la consignas de rumbo y velocidad.

Más formalmente:

1. El barco se posiciona sobre un espacio continuo, el espacio real (Espacio de estados infinito no contable).

2. Se proyecta la posición, en el espacio real, del barco sobre un espacio discreto (Espacio de estados infinito contable).

3. Unas determinadas consignas serán validas en tanto que el barco se mantenga en la misma posición sobre el espacio discreto.

El espacio discreto es infinito contable, porque aunque tenga infinitos elementos sus elementos se pueden asociar uno a uno sobre el conjunto de los números naturales.

Para realizar esta discretización se asigna una malla al espacio real, cada malla corresponde con un punto del espacio discreto. Esta discretización se construye según las siguientes ecuaciones, donde x, y son las posición real y X, Y la posición discreta.

$$X = 10 * sign(x) * |round(\frac{x}{10})| \tag{4.1}$$

$$Y = 10 * sign(y) * |round(\frac{y}{10})| \tag{4.2}$$

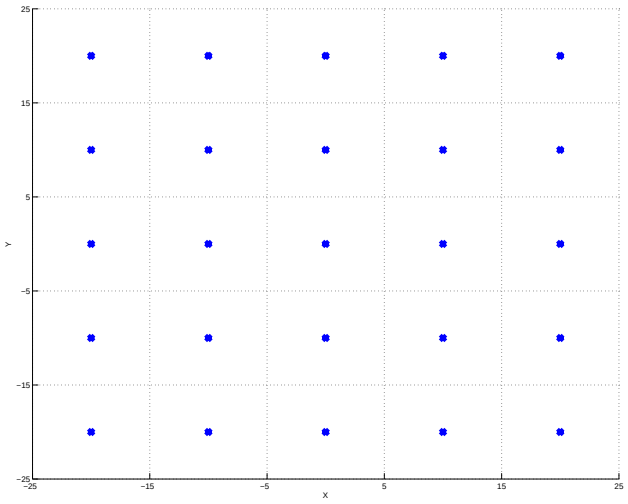


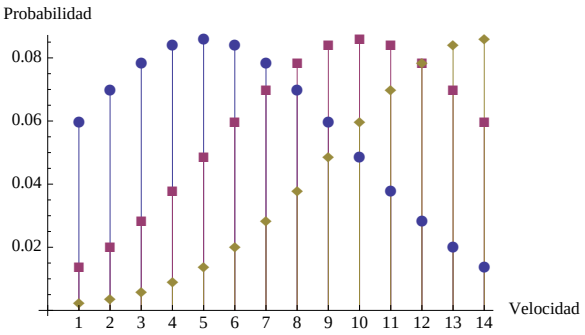
Figura 4.1: Discretización del espacio, empleando un mallado de celdas de 10x10m

4.2. Construcción de la heurística

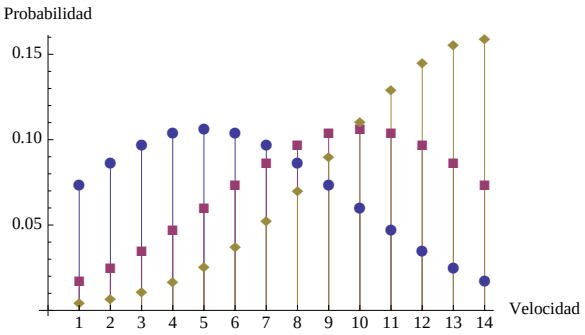
Puesto que la optimización se realiza sobre dos parámetros, velocidad y rumbo, necesitamos dos funciones heurísticas.

Heurística para la velocidad: se construye a partir de una distribución de probabilidad normal, de ésta distribución extraemos los valores de probabilidad para las consignas de velocidad disponibles, estos valores normalizados reflejarán la probabilidad de que una consigna determinada sea escogida. En el caso de que exista restricción de velocidad la distribución normal estará centrara en el valor que marca dicha restricción, si la velocidad no esta restringida la normal se centrará en el valor máximo de las velocidades disponibles. En ambos casos la desviación típica de dicha distribución se fija a N/3 donde N es el número de valores disponibles para la velocidad. El resultado de este proceso esta ilustrado en la figura 4.2

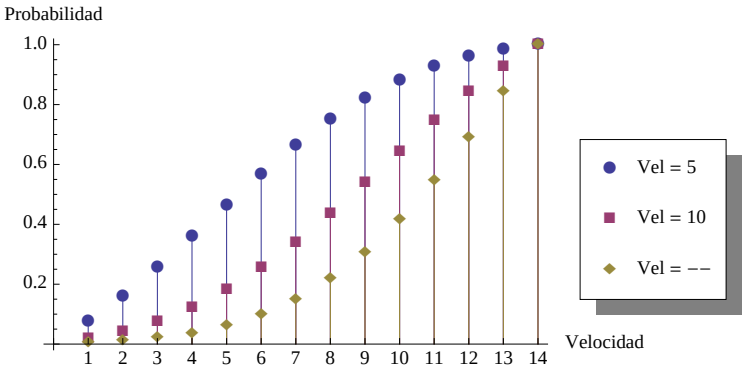
Heurística para el rumbo: en este caso construir una heurística es más difícil porque debemos considerar las restricciones dinámicas y la no holonomicidad del barco. El planteamiento que se



(a) Valores extraídos de la distribución Normal



(b) Valores normalizados



(c) Valores acumulados

Figura 4.2: Distribuciones de probabilidad para restricciones de velocidad {5, 10, sin-restricción}

ha seguido es relajar las restricciones. Es decir, considerar únicamente las restricciones holonómicas: no es posible girar sobre sí mismo y la fuerza de impulso actúa en un sentido único. La heurística a desarrollar deberá cumplir estas condiciones, es posible realizar la siguiente aproximación:

1. Resolver de manera no automática el caso más simple. Donde el caso más simple es encontrase en una posición adyacente al objetivo en el espacio discreto.
2. A partir de la solución del caso simple construir una heurística para el resto del espacio. Es decir, propagar la solución sobre cualquier posición del espacio discreto.

Para realizar esta tarea utilizamos un autómata celular, ya que es un sistema que exhibe propiedades de propagación.

4.2.1. El autómata celular y la orientación del espacio de búsqueda

Un autómata celular [10, 11] es una matriz de autómatas, donde la entrada de cada autómata es el estado de los autómatas adyacentes. La característica fundamental es que a partir de una configuración inicial, el sistema –la matriz de autómatas– irá evolucionando y si es posible alcanzará una configuración estable. Un ejemplo típico en autómatas celulares es el juego de la vida de Conway [12]. La evolución se logra aplicando las reglas de transición de estados que define los autómatas individuales.

Este tipo de sistema auto-organizados se ha usado en el estudio de modelos para la propagación de enfermedades. Se define unas reglas tales que cada autómata intenta propagar su estado a los adyacentes, es decir, el estado de cada autómata es el resultado de la combinación de los estados de sus adyacentes. En nuestro caso nos interesa que los autómatas propaguen un rumbo, para conseguir esto modelizamos el rumbo como un vector de módulo 1, cuyo ángulo sobre el eje de las x indica el rumbo. De este modo la regla de transición se consigue utilizando las propiedades de la suma vectorial.

Hacemos coincidir posiciones del espacio discreto (Ver apartado 4.1.1) con un autómata concreto. Construimos una matriz de autómatas cuadrada, en el centro situamos al elemento que corresponde con el objetivo y definimos el tamaño de la matriz en función de la distancia entre el objetivo y el punto de partida. Las siguientes ecuaciones muestran como calcular la dimensión de la matriz (L) y el elemento (row,col) que corresponde con la posición real del objetivo (x,y) tomando como punto de partida la posición (0,0).

$$L = 2 * ceil(\frac{max(|x|,|y|)}{10}) + 1 \tag{4.3}$$

$$row = fix(\frac{L+1}{2}) \tag{4.4}$$

$$col = fix(\frac{L+1}{2}) \tag{4.5}$$

Existen dos tipos de autómatas[1], los fijos (FA) y los variables (VA). Los autómatas fijos son aquellos

[1] Si nos atenemos a la definición de autómata celular, todos los autómatas tienen que ser del mismo tipo. En nuestro

que no varían de estado, mientras que los variables son los que evolucionan.

- El estado de un autómata será el ángulo del vector unitario, restringido a valores enteros entre 0º y 359º.

- Los autómatas evolucionan de acuerdo con la siguiente función de transición, donde s representa el estado del autómata.

$$\delta_{FA}(s) = s \tag{4.6}$$

$$\delta_{VA}(s) = \sum^{adyacente(s_i,s)} [cos(s_i), sin(s_i)] \tag{4.7}$$

- Modo de Operación:

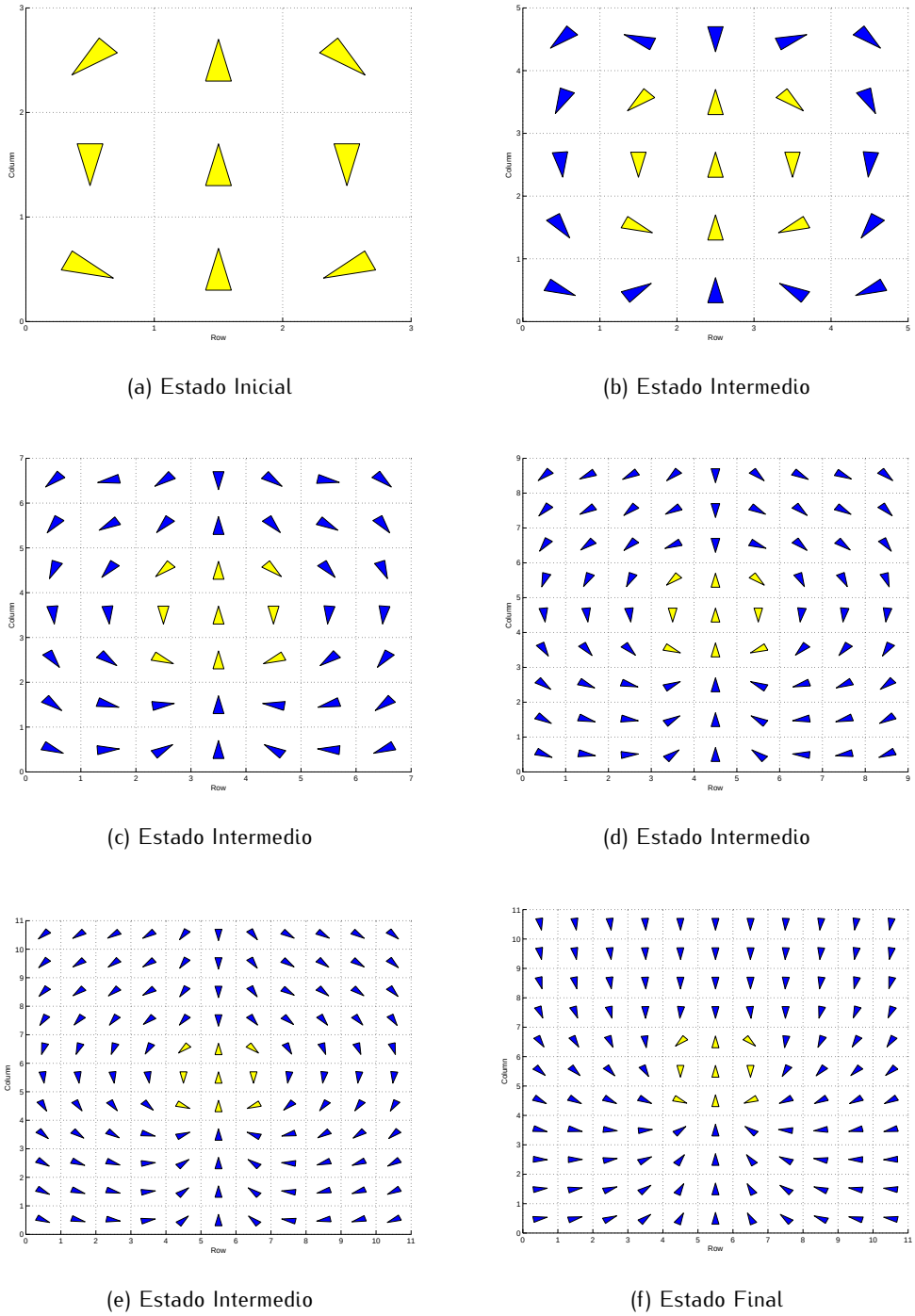
1. El objetivo fijado se define como un autómata, cuyo estado coincide con el rumbo que deberá tener el barco al finalizar la maniobra.

2. Se fijan sus vecinos con unos estados definidos previamente de tal manera que generen un bucle de circulación hacia el objetivo (ver figura 4.3a). Esto es lo que se señalo anteriormente cuando se hacía mención de propagar la solución más simple: estar en una casilla adyacente al objetivo.

3. Se hace evolucionar al autómata celular aplicando *simultáneamente* la función de transición a cada autómata. Esto propaga la solución, representada por los autómatas adyacentes al autómata objetivo, por el resto del espacio contenido en la matriz. El resultado es la generación de un campo vectorial cuyas líneas de flujo llevan a la celda objetivo. Este proceso se puede observar en la figura 4.3

4. La evolución se detiene cuando ha alcanzado un estado estable o se han alcanzado un número límite de iteraciones.

caso esto se sigue cumpliendo, porque hablar de dos tipos de autómatas es lo mismo que considerar que existen un estado más, el fijo, y definir una función de transición por casos.



(a) Estado Inicial

(b) Estado Intermedio

(c) Estado Intermedio

(d) Estado Intermedio

(e) Estado Intermedio

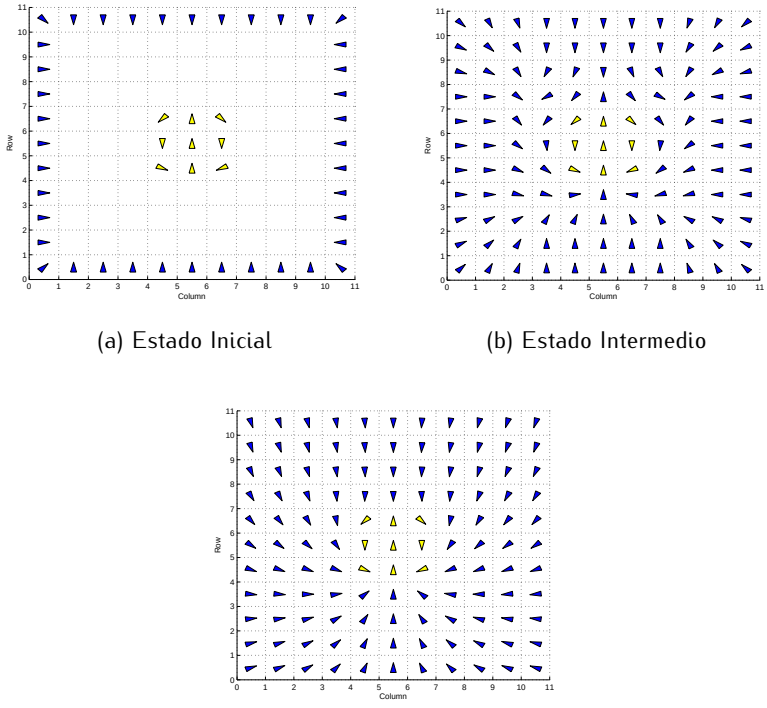
(f) Estado Final

Figura 4.3: Evolución del autómata celular.

- Para intentar acelerar la convergencia y evitar que las líneas de flujo se propaguen hacia el

exterior, se inicializan los autómatas correspondientes a los bordes de la matriz con un ángulo que apunte al interior de la matriz. Este proceso se puede observar en la figura 4.4.

- La solución, en las regiones del espacio no representadas directamente en la matriz, se obtiene asignando el valor de la celda de la matriz, más cercana a la región del espacio donde se desea obtener la solución.



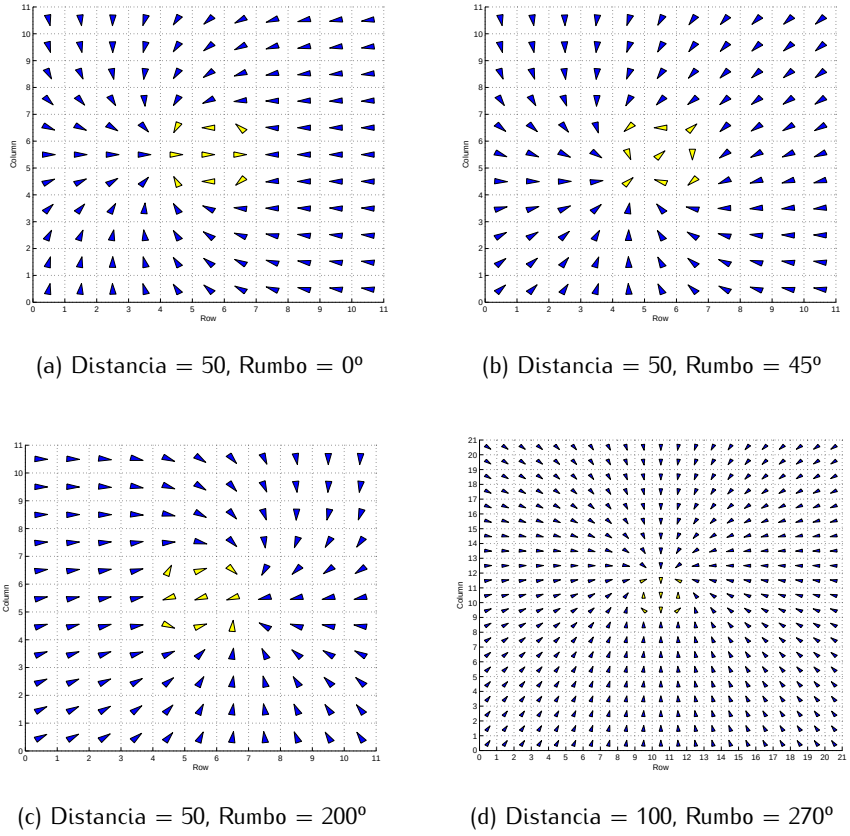
(a) Estado Inicial

(b) Estado Intermedio

(c) Estado final

Figura 4.4: Evolución del autómata celular con inicialización en los bordes.

En la figura 4.5 se pueden observar distintos autómatas celulares según el rumbo objetivo fijado.



(a) Distancia = 50, Rumbo = 0º

(b) Distancia = 50, Rumbo = 45º

(c) Distancia = 50, Rumbo = 200º

(d) Distancia = 100, Rumbo = 270º

Figura 4.5: Autómata celular para distintos rumbos objetivos.

4.3. Ant Colony Extended (ACE): Extensión del algoritmo ACO

4.3.1. Introducción de la dinámica del problema

Por lo dicho en secciones anteriores (2.1.2), *las restricciones dinámicas deben estar activas, deben ser tenidas en cuenta. Así se puede asegurar que la trayectoria será realizable y que realmente se esta obteniendo –o aproximando– el óptimo.*. Aunque la aproximación más empleada suele planificar sin incluir explícitamente las restricciones dinámicas en el planificador, y comprobar después la viabilidad de las soluciones, una alternativa más atractiva es incluir en el planificador las restricciones dinámicas del problema y obtener así un plan viable, incluso óptimo, en una sola etapa.

En nuestra resolución la dinámica se incorpora al movimiento de las hormigas. Es decir, las hormigas se mueven, generan trayectorias, de acuerdo a la dinámica introducida. Esta es una de las características incorporadas al algoritmo extendido (ACE). Este planteamiento presenta otra ventaja añadida que es la independencia del método de resolución frente a la dinámica, si tenemos que resolver trayectorias para distintos tipos de barcos, lo único que habría que hacer es introducir su dinámica concreta.

Para incorporar la dinámica al movimiento de las hormigas se calcula el movimiento de las hormigas mediante la resolución numérica del modelo del barco. Suponiendo una configuración S de las variables de estado definidas en el modelo, el procedimiento es el siguiente:

1. Se incorpora el modelo a la hormiga. Es decir, la hormiga almacena las variables de estado del modelo.

2. La hormiga asigna un valor de consigna a la velocidad y al rumbo.

3. Se resuelve el modelo según el estado almacenado en la hormiga y con los valores de consigna previamente calculados. La resolución del modelo consiste en la integración numérica de las ecuaciones diferenciales durante el periodo de tiempo que conlleve un cambio en la posición espacial sobre el espacio discreto (ver sección 4.1.1).

4. Se actualiza el estado de la hormiga en función del nuevo estado obtenido de la resolución del modelo.

Grafo vs Evaluación Dinámica

En la versión clásica del ACO propuesta Dorigo en [4] la optimización se realiza sobre un conjunto de estados finitos. Es decir, dado un conjunto finito de estados, de configuraciones del sistema, se busca la configuración óptima, aquella que minimiza la función de coste.

Se recurre a un grafo para expresar el problema. Los nodos del grafo representan diferentes estados del sistema. Las aristas salientes de un nodo las acciones disponibles. De tal modo que a partir de un estado determinado –un nodo– existen una serie de acciones –aristas– cuyo resultado son otros estados del sistema –otros nodos–. Las restricciones del sistema están implícitas en el grafo, las aristas representan las restricciones del sistema, ya que definen la transición entre los estados del sistema según las diferentes acciones realizadas. En nuestro caso no es posible expresar las restricciones dinámicas en forma de grafo, ya que existen infinitos estados. Incluso después de la discretización las posiciones del espacio discreto donde puede encontrarse el barco son infinitas, por lo que no es posible construir un grafo que las represente, pues dicho grafo necesitaría tener infinitos nodos, tantos como estados del sistema.

En lugar del grafo empleado por el ACO, el ACE utiliza lo que denominamos evaluación dinámica. Que consiste en representar los resultados de las acciones mediante una función de transición de estados. En nuestro caso se resuelve el modelo para un determinado estado y valores de velocidad y rumbo. De este modo los diferentes estados del sistema alcanzables a partir de otro estado y mediante las diferentes acciones, no están definidos explícitamente, ni son calculados previamente. Son generados de manera dinámica según las necesidades del sistema.

En el fondo estamos utilizando la misma técnica de computación que se conoce en programación funcional como *evaluación perezosa*[2] [13]. Esta técnica consiste en retrasar la evaluación de una expresión, el computo, hasta que sea necesario, de tal modo que permite el manejo de estructuras de datos infinitas. Por ejemplo, si queremos obtener el tercer elemento de la lista de números naturales. Una forma de computación es generar la lista de los números naturales y después extraer el elemento que ocupe la tercera posición, claro que al ser la lista de números naturales infinita, la generación de

[2]También se hace referencia a ella como “delayed evaluation”

la lista no tiene fin, por tanto la computación no puede realizarse. Pero también podemos ir generando la lista poco a poco, es decir, generamos un elemento y dejamos el resto de la generación de la lista en espera. En cada paso comprobamos si el elemento generado es el tercero, si lo es hemos acabado, si no lo es generamos uno más y volvemos a comprobar.

Cuando al ACO se le exige un grafo, se exige que previamente al proceso de optimización se calcule el resultado de aplicar a cada estado del sistema el conjunto de acciones disponibles. El ACE por el contrario trabaja con formas funcionales, genera los estados de manera dinámica según el estado actual y la acción aplicada. Utilizar esta aproximación, almacenar el conjunto de estados en forma funcional y generarlos según vayan siendo necesarios, presenta las ventajas propias de la evaluación perezosa:

1. Reduce la carga computacional ya que no se realizan cómputos innecesarios.

2. Además permite el manejo de espacios de estados infinitos, porque estos no se generan en su totalidad.

4.3.2. Hormigas exploradoras y hormigas recolectoras

> On one day we mapped the locations of patrollers, and then the locations of foragers. Then the next day we repeated the process. Comparing the distributions of forager and patroller directions, we learned that foragers use the trails explored earlier the same morning by patrollers. In this sense, patrollers choose the day's trails[3].

El trabajo de Deborah Gordon [14] sobre las hormigas del desierto de Arizona presenta una diferencia clara respecto al trabajo [6] que inspira al ACO. En el ACO la exploración se realiza por un único tipo de hormiga, recolectora. Por el contrario, en las hormigas de Gordon el proceso involucra dos tipos de hormigas: recolectoras (foragers) y exploradoras (patrollers). En sus estudios Gordon señala que este segundo tipo de hormiga abandona la colonia previamente a las recolectoras y presentan un menor número de individuos que la clase recolectora. Las exploradoras parten en distintas direcciones –en principio por las zonas de recolección habituales– y cuando encuentran alimento estimulan la recolección en la zona donde se encuentran.

Este comportamiento es interesante porque indica que la recolección y generación de rutas óptimas, no es como en el experimento del doble puente y como ocurre en el ACO. La generación de rutas de recolección aparecen únicamente en la zona que previamente han marcado las hormigas exploradoras. Gordon explica este tipo de comportamiento por la necesidad de optimizar el proceso de recolección, el desierto presenta unas condiciones muy duras para las hormigas, se ven sometidas a procesos de deshidratación severos. Por tanto una estrategia similar a la del ACO, es decir, las recolectoras parten hacia todas las zonas de recolección conocidas sería catastrófico. Porque habría muchas que se deshidratarían sin encontrar comida, con lo que el beneficio obtenido sería menor que el gasto realizado. Es necesario asegurarse de que la recolección se lleva a cabo en una zona donde exista comida, determinar esa zona es la tarea de las hormigas exploradoras.

Inspirándonos en las investigaciones biológicas de Gordon diseñamos el algoritmo extendido (ACE) para que utilice los dos tipos de hormigas, exploradoras y recolectoras.

[3] [14] D.Gordon "Ants at Work" pag. 49

Hormigas Recolectoras: son las hormigas clásicas del ACO, se mueven por el espacio de estados asignando valores a los parámetros de optimización. La elección de dichos parámetros se realiza en base a las probabilidades marcadas en la tabla de feromona.

Hormigas Exploradoras: este tipo de hormiga se encarga de explorar el espacio de búsqueda. Se mueven por el espacio de estados asignando valores a los parámetros de optimización. La elección de dichos parámetros se realiza en combinación con la heurística definida previamente (las recomendaciones del automata) y la tabla de feromona. Es decir, ya no hay un seguimiento ciego de la tabla de feromona, sino que es posible incorporar nuevas elecciones.

Volviendo a la discusión anterior (apartado 4.3.1 Grafo vs Evaluación Dinámica), la evaluación dinámica introduce una diferencia operacional entre el ACO y el ACE. Mientras que en el primero la tabla se inicializa previamente a la ejecución, en el ACE la tabla comienza vacía y se rellena de manera dinámica.

El modo de funcionamiento es muy sencillo, las posiciones del espacio que hayan sido utilizadas para las construcción de una trayectoria que alcance el objetivo, estarán presentes en la tabla de feromona. Las hormigas recolectoras utilizaran las decisiones de consigna marcadas y las exploradoras podrán hacerlo o guiarse por los resultados del autómata. De este modo pueden generar nuevas decisiones que no estén presentes en la tabla, ampliado el espacio de búsqueda. Este mecanismo protege al ACE de converger hacía mínimos locales, cumpliendo la misma función que la evaporación de feromona artificial en el ACO.

Este esquema recuerda a los algoritmos genéticos donde existe generación de soluciones por mutación y cruce. El operador de cruce genera soluciones por combinación de soluciones previas, y el operador de mutación introduce modificaciones al azar sobre una solución. En términos funcionales las hormigas recolectoras actuarían como los operadores de cruce de un genético y las hormigas exploradoras como el operador de mutación[4].

Formalización

A continuación se describe la dinámica de decisión para ambos tipos de hormigas. Siendo s el estado actual del sistema, s^+ el estado siguiente y $\sigma \in [0.,1]$ una constante.

La elección de valores para las variables de decisión se realiza atendiendo a las siguientes ecuaciones de probabilidad.

- *Explorar*: Probabilidades para la elección de rumbo y velocidad.

$$P_{velocidad}(s_i) = Normal(\mu, \sigma, P(X = s_i)) \tag{4.8}$$

$$P_{rumbo}(\alpha_i) = max(0, 1 - \frac{mod(\theta - \alpha_i, 2\pi)}{\pi/4}) \tag{4.9}$$

[4]Investigar como de equivalente resulta el esquema del ACE y los genéticos es una de las líneas de investigación paralelas que tenemos abiertas. El atractivo de esta equivalencia reside en estudiar si la teoría subyacente a los algoritmos genéticos es aplicable al ACE. En los genéticos dado un problema, hay que definir la representación cromosómica, el operador de cruce, el operador de mutación y la función de selección. En el ACE únicamente hay que definir las entradas de la tabla de feromona, ya que el movimiento de las hormigas viene definido por el propio problema. Los procesos de selección, recombinación y mutación son intrínsecos a la dinámica del sistema, no hay que definirlos.

Algorithm 1 Esquema de decisión para hormigas exploradoras

if $random \geq \sigma$ **then**
  $rumbo(s^+) \leftarrow explorar(s)$
  $velocidad(s^+) \leftarrow explorar(s)$
else
  $rumbo(s^+) \leftarrow feromona(s)$
  $velocidad(s^+) \leftarrow feromona(s)$
end if

Algorithm 2 Esquema de decisión para hormigas recolectoras

if $isEmpty(feromona(s))$ **then**
  $rumbo(s^+) \leftarrow explorar(s)$
  $velocidad(s^+) \leftarrow explorar(s)$
else
  $rumbo(s^+) \leftarrow feromona(s)$
  $velocidad(s^+) \leftarrow feromona(s)$
end if

donde,

- μ = velocidad máxima disponible o velocidad objetivo.
- σ = desviación estándar predefinida.
- s_i = i-velocidad disponible.
- α_i = i-rumbo disponible.
- θ = sugerencia de rumbo del automata.

- *Feromona*: Probabilidades para la elección de rumbo y velocidad.

$$P_{velocidad}(s_i) = \Gamma_{s_i} \tag{4.10}$$
$$P_{rumbo}(\alpha_i) = \Gamma_{\alpha_i} \tag{4.11}$$

donde

- s_i = i-velocidad disponible.
- α_i = i-rumbo disponible.
- Γ_s = tabla de probabilidades (tabla de feromona) para la velocidad.
- Γ_α =tabla de probabilidades (tabla de feromona) para el rumbo.

4.3.3. Constitución y modificación de la tabla de feromonas

La tabla de feromona almacena la distribución de probabilidad que van construyendo las hormigas. Para almacenar esta distribución utilizamos como estructura de datos una tabla hash.

- Tabla de feromona para la velocidad
  - Clave = Posición en el espacio discreto (ver sección 4.1.1).
  - Valor = Vector de probabilidades de [1..N] donde cada posición del vector corresponde con una velocidad disponible, siendo N la máxima velocidad disponible.
- Tabla de feromona para el rumbo
  - Clave = Posición en el espacio discreto (ver sección 4.1.1).
  - Valor = Vector de probabilidades de [1..360] donde cada posición del vector corresponde con un rumbo determinado.

Cuando una hormiga (hormiga$_{gl}$) alcanza el objetivo, actualiza las tablas según las siguientes ecuaciones.

- Actualización de la tabla de probabilidades

$$\Gamma^{+}_{s_n} = \Gamma_{s_n} + f(t,\mu,\delta)(1-\Gamma_{s_n}) \tag{4.12}$$
$$\Gamma^{+}_{\alpha_n} = \Gamma_{\alpha_n} + f(t,\mu,\delta)(1-\Gamma_{\alpha_n}) \tag{4.13}$$
$$\forall i \neq n \qquad \Gamma^{+}_{s_i} = \Gamma_{s_i} - f(t,\mu,\delta)\Gamma_{s_i} \tag{4.14}$$
$$\forall i \neq n \qquad \Gamma^{+}_{\alpha_i} = \Gamma_{\alpha_i} - f(t,\mu,\delta)\Gamma_{\alpha_i} \tag{4.15}$$

  donde

  - s_n = valor de velocidad utilizado por la hormiga$_{gl}$
  - α_n =valor de rumbo utilizado por la hormiga$_{gl}$
  - s_i = i-velocidad disponible.
  - α_i = i-rumbo disponible.
  - Γ_s = tabla de probabilidades (tabla de feromona) para la velocidad.
  - Γ_α =tabla de probabilidades (tabla de feromona) para el rumbo.
  - t = tiempo consumido en realizar la trayectoria.
  - μ = media de los tiempos de todas las trayectorias que han llevado al objetivo.
  - δ = tiempo de la mejor trayectoria encontrada.
  - $f(t,\mu,\delta)$ = función de evaluación de trayectorias.
- Evaluación de trayectorias.

$$f(t,\mu,\delta) = max(0, \frac{t-\mu}{\delta-\mu}) \tag{4.16}$$

  donde

  - t = tiempo consumido en realizar la trayectoria.

- μ = media de los tiempos de todas las trayectorias que han llevado al objetivo.
- δ = tiempo de la mejor trayectoria encontrada.

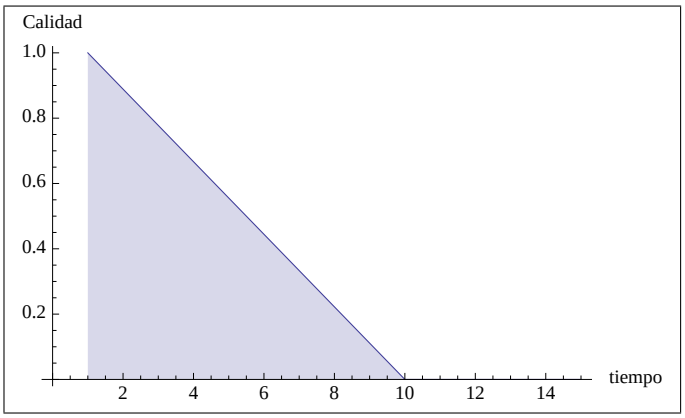


Figura 4.6: Función de evaluación para $[t, \mu, \delta] \rightarrow [\{1.,15\}, 10, 1]$

Existen, en tiempo, dos casos extremos: la trayectoria a evaluar esta por debajo de la media o la trayectoria a evaluar es la mejor encontrada. En estos casos el comportamiento se puede simplificar de la siguiente manera:

Trayectoria es inferior a la media

$$\begin{aligned} & f(t,\mu,\delta) \rightarrow 0 \\ & \Gamma_n^+ = \Gamma_n \\ \forall i \neq n \qquad & \Gamma_i^+ = \Gamma_i \end{aligned}$$

Las probabilidades no sufren cambios, se desecha la trayectoria evaluada.

Trayectoria es la mejor encontrada

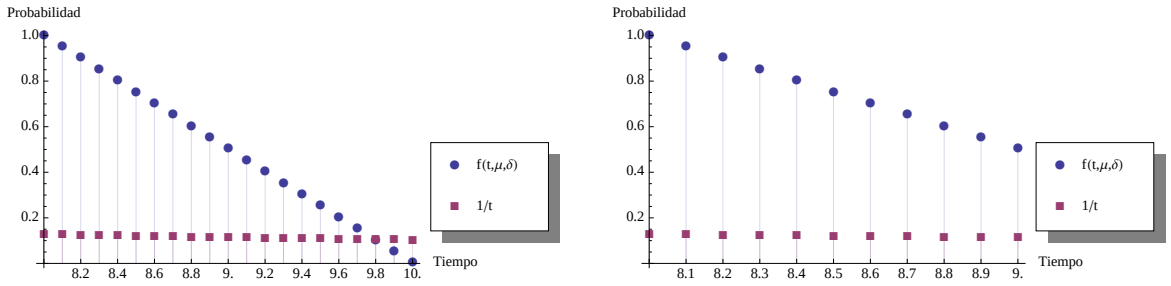
$$\begin{aligned} & f(t,\mu,\delta) \rightarrow 1 \\ & \Gamma_n^+ = \Gamma_n + (1 - \Gamma_n) = 1 \\ \forall i \neq n \qquad & \Gamma_i^+ = \Gamma_i - \Gamma_{s_i} = 0 \end{aligned}$$

Se desechan el resto posibilidades, dejando como única alternativa la trayectoria evaluada.

Evaluación dinámica

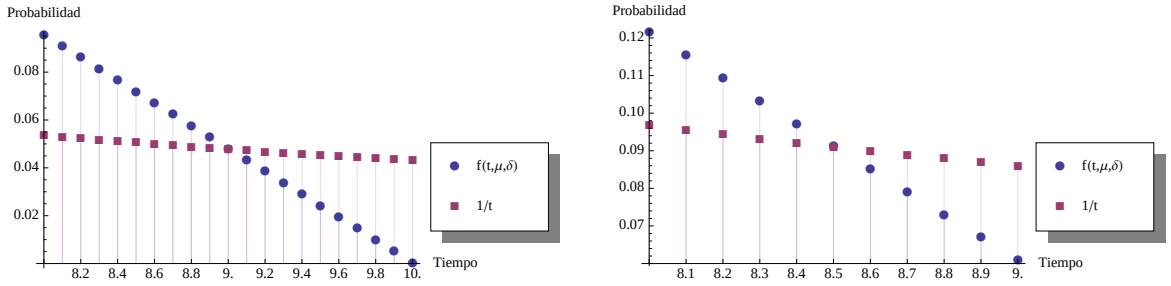
Es habitual utilizar un criterio estático para la evaluación de una solución, es decir, la calidad asignada a una solución no varia. Por el contrario nosotros utilizamos un criterio que no es dependiente únicamente de la solución en sí, sino que también depende del resto de soluciones encontradas. De este modo la calidad asignada a una solución es variable.

Evaluar las soluciones según la posición que ocupen en relación a la mejor solución encontrada y la media del resto de soluciones, ofrece un mayor refinamiento de la comparación. Esto es debido a que la comparación se realiza únicamente teniendo en cuenta las diferencias y reduciendo las soluciones a comparar (se eliminan las soluciones por debajo de la media). La figura 4.7 muestra la diferencia entre comparar distintos tiempos de solución, utilizando como criterio de calidad directamente la inversa del tiempo empleado en recorrer la trayectoria ($\frac{1}{t}$) y la función de evaluación descrita en 4.16. Para la situaciones donde $[t,\mu,\delta] \rightarrow [\{8\ldots10\},10,8]$ y $[t,\mu,\delta] \rightarrow [\{8\ldots9\},10,8]$



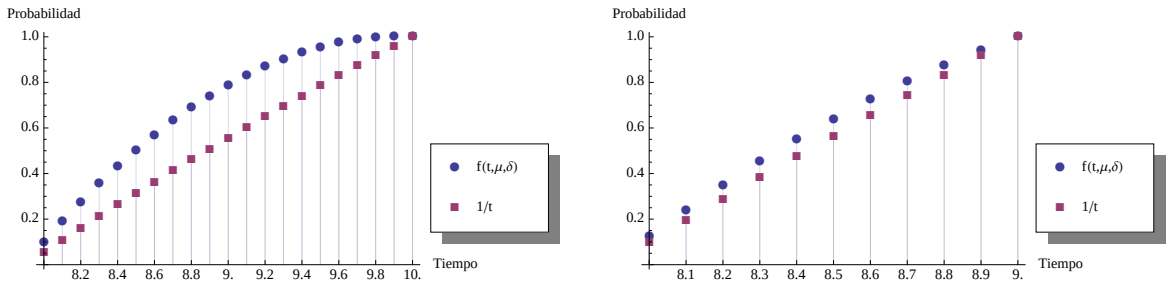
(a) Valores de salida $(t,\mu,\delta) \rightarrow (\{8\ldots10\},10,8)$

(b) Valores de salida $(t,\mu,\delta) \rightarrow (\{8\ldots9\},10,8)$



(c) Valores normalizados $(t,\mu,\delta) \rightarrow (\{8\ldots10\},10,8)$

(d) Valores normalizados $(t,\mu,\delta) \rightarrow (\{8\ldots9\},10,8)$



(e) Valores acumulados $(t,\mu,\delta) \rightarrow (\{8\ldots10\},10,8)$

(f) Valores acumulados $(t,\mu,\delta) \rightarrow (\{8\ldots9\},10,8)$

Figura 4.7: Calidad de las trayectorias según la función de evaluación utilizada.

4.3.4. Arquitectura del sistema

En los Algoritmos Genéticos se hace referencia como paralelización implícita (implicit parallelism) al hecho de que el algoritmo maneje y procese múltiple soluciones simultáneamente. En general en métodos algorítmicos evolutivos y los denominados Swarm Intelligence (inteligencia de enjambre) se caracterizan por realizar un procesamiento múltiples de distintas soluciones, trabajan con "poblaciones" de soluciones.

En los algoritmos genéticos y en distintas aplicaciones del ACO, aunque no en todas, este procesamiento múltiple se realiza de manera síncrona. Es decir, el procesamiento se lleva a cabo de una manera seriada, hasta que el sistema no ha finalizado una etapa, no comienza la siguiente. La arquitectura básica del ACO puede observarse en la figura 4.8, su funcionamiento es el siguiente:

1. *Construcción de la solución*, que conlleva esperar a que todas las búsquedas llevadas a cabo por las hormigas hayan finalizado.

2. *Actualización de la solución*, conlleva la actualización de la tabla de feromona y el procesamiento de la mejor solución encontrada.

3. *Asignación de hormigas*, es decir, se vuelven a posicionar las hormigas en el espacio de búsqueda.

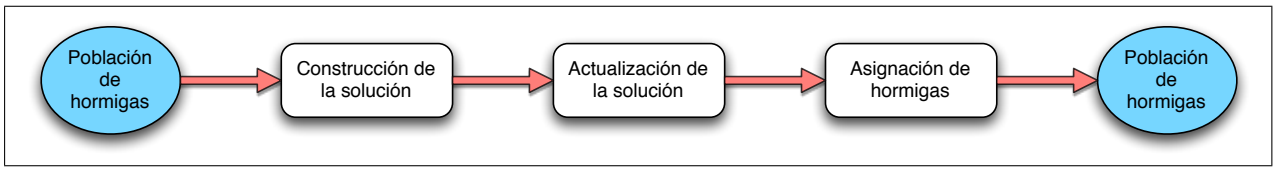


Figura 4.8: Arquitectura básica del ACO

El ACE presenta otra arquitectura representada en la figura 4.9. Nos basamos en la arquitectura de subsunción propuesta por R. Brooks [15, 16, 17] para el desarrollo de sistemas complejos. Dicha arquitectura se basa en un diseño modular en capas interconectadas, donde todas las capas actúan en paralelo. Su funcionamiento es el siguiente:

1. *Construcción de la solución*, se realiza una paso en la construcción de la solución. Es decir, se toma una decisión de rumbo y velocidad.

2. *Actualización de la solución*, aquellas hormigas que hayan terminado de construir su solución, actualizan la tabla de feromona. También se comprueba la mejor solución encontrada.

3. *Asignación de hormigas*, se posicionan nuevas hormigas en función de las que hayan terminado.

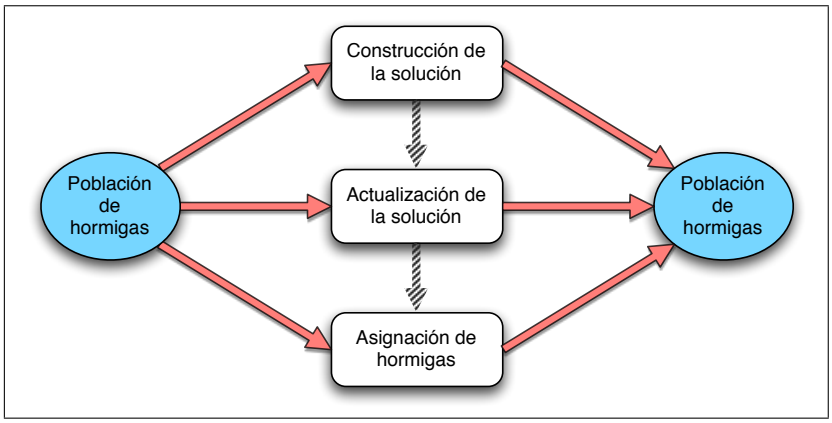


Figura 4.9: Arquitectura del ACE

Si bien no es la aplicación estricta de la arquitectura de subsunción, ya que esta requiere que cada capa produzca una salida independiente del resto. Esto no se cumple exactamente en nuestro sistema, ya que la salida de una capa es la entrada de la siguiente, pero se siguen conservando otros principios de este tipo de arquitectura:

- Todas las capas actúan modificando el entorno –la población de hormigas–.
- Todas las capas actúan en paralelo, es decir, en cada iteración del sistema todas las capas producen una salida.

La flexibilidad que permite este tipo de aproximación propuesta por Brooks se puede observar si se comparan ambas arquitecturas, la del ACO y la de nuestro sistema:

- Dado un momento de cómputo cualquiera, una iteración concreta:
  - El ACO estará en un modo de funcionamiento: construcción, actualización o asignación. Por tanto el ACO, como sistema, nunca exhibirá ningún comportamiento que no sea producto directo de alguno de sus módulos computacionales.
  - Nuestra sistema solo tiene un modo de funcionamiento. El comportamiento que exhibe a nivel global será producto de la combinación de *todos* los módulos, no directamente de uno solo.

Por ejemplo, en el ACO clásico el número de hormigas es fijo, en nuestro sistema es variable. Es posible introducir en el ACO clásico un número variable de hormigas, pero este número será fijo durante todas las etapas hasta que se alcance el módulo de asignación de hormigas. Por el contrario en nuestro sistema el número de hormigas puede variar en cada iteración. Esto ofrece la ventaja de que se puede auto-organizar, es decir, el sistema puede ajustar el número de hormigas en cada instante según su estado.

Pérdida del sincronismo y tabla de feromona

En nuestro sistema al romper el funcionamiento por etapas, tiene la consecuencia de que la tabla de feromona no esta sincronizada. Es decir, no permanece inalterada durante la construcción

de soluciones. Cuando hormiga alcance el objetivo modifica la tabla de feromonas, las hormigas que se encuentran construyendo su solución verán alterada su tabla. Las primeras consignas que tomaron estaban vinculadas a una tabla de feromona distinta de la actual. En el ACO clásico mientras las hormigas construyen la solución, la tabla sobre la que se realiza el cálculo permanece fija, solo se modifica cuando la etapa de construcción ha finalizado.

Además en el caso de que la trayectoria a evaluar sea la mejor encontrada hasta el momento, se deja como única alternativa posible la trayectoria evaluada (ver sección 4.3.3).

En un primer momento se puede pensar que estos dos efectos solo tendrían consecuencias negativas:

- La función de evaluación, utilizada para medir la calidad de las trayectorias, acelera excesivamente la convergencia. Por lo tanto se explora menos el espacio de búsqueda, aumentando el riesgo de los mínimos locales.

- El no sincronizar la tabla de feromona elimina la convergencia del algoritmo. Porque para que esto ocurra es necesario que todas las hormigas trabajen sobre la misma tabla, construyendo una única distribución de probabilidad.

Si bien es cierto que cada una de estas circunstancias por separado son perjudiciales, cuando las juntamos ocurre exactamente lo contrario. De la combinación resulta una modificación del método de selección colectiva llevado a cabo por los algoritmos de hormigas.

4.3.5. Mínimos locales y selección colectiva

Es común a todos lo métodos de optimización el converger hacia mínimos o máximos locales. El ACO no es una excepción. El peligro radica en la convergencia propia de los métodos de optimización. Es decir, es una condición necesaria al diseñar un método de optimización que este converja, esto significa que bien el proceso de búsqueda de soluciones finalice ó que alcance un estado estable donde la salida no varíe con el tiempo.

El problema reside en que si la convergencia se da hacia un mínimo o máximo local, la solución global no va a ser alcanzable. En principio no existe garantía ninguna de como evitar las zonas de convergencia local, lo único que se puede es diseñar mecanismos para evitar sus efectos. Por ejemplo, los algoritmos genéticos incorporan los operadores de mutación que ayudan a evitar zonas de convergencia, el ACO incorpora la evaporación artificial de feromona para minimizar el riesgo de convergencia hacia un mínimo o máximo local.

En nuestro sistema ya hemos incorporado un mecanismo para disminuir los efectos de la convergencia hacia mínimos. Se trata de las hormigas exploradoras, estas al seguir de manera parcial la feromona, pueden descubrir nuevas zonas de búsqueda evitando que el sistema caiga en un mínimo local.

En la naturaleza también existen procesos de optimización, como se explico anteriormente (ver apartado 2.2.1) los algoritmos de hormigas están basados en los comportamientos de hormigas reales, concretamente en el mecanismo de decisión colectiva. La selección colectiva tiene el objetivo de explotar la fuente de alimento con mejor calidad, esto es una optimización. Lo que se persigue es optimizar el gasto requerido por la recolección frente al beneficio obtenido, es decir se explota la fuente de alimento que mejor ratio de $beneficio/gasto$ presente. En el libro de Camazine at al. [18]

se puede encontrar una comparación entre la selección colectiva llevada a cabo por las hormigas y por las abejas, debido a que son mecanismos muy similares.

Si bien la dinámica de las hormigas es fácilmente trasladable a un esquema algorítmico, la dinámica de una colmena de abejas es difícilmente reproducible en dicho esquema. Esto es debido a que la construcción de las soluciones por parte de las hormigas se realiza poco a poco, cosa que no ocurre con las abejas. Por el contrario el esquema de selección de las abejas, desde el punto de vista de la optimización, ofrece la ventaja de ser un método con una gran flexibilidad para cambiar la elección una vez tomada, lo que ayuda a evitar la convergencia en soluciones subóptimas. Nuestro objetivo es modificar la selección colectiva propia de las hormigas para asemejarla a la de las abejas, y de este modo conseguir mantener la dinámica de construcción de soluciones por parte de las hormigas pero utilizando un criterio de selección colectiva más propio de las abejas. La ventaja de esta mezcla es dotar al sistema de una segunda barrera para evitar la convergencia hacia mínimos locales.

Selección colectiva de las fuentes de néctar por parte de las abejas

Lo interesante del proceso de selección llevado a cabo por las abejas, es su capacidad de cambiar la fuente de explotación del néctar, si en su entorno aparece una fuente mejor. Es decir, cambiar de solución incluso si el proceso de selección ha convergido. El método de selección esta explicado por Seeley, Camazine y Sneyd en [19, 18]. El esquema fundamental es el que se representa en la figura 4.10.

> An important first step toward understanding how the allocation process work is to distinguish between *employed* and *unemployed* foragers; between foragers currently engaged in exploiting a patch of flowers and those that are not. The members of these two groups have markedly different behaviors. Employed foragers generally bring home food and may provide information about the location of a food source, while unemployed foragers receive information about food sources and then search for one of the advertised work sites[5].

Las abejas recolectoras son las encargadas de traer el néctar a la colonia y de informar acerca de la fuente de néctar que esta explotando. Cuando una recolectora ha descargado el néctar, puede tomar tres decisiones:

- Abandonar la fuente de explotación. Pasa a estar desocupada y a ocupar una posición de observación en la zona de baile.
- Continuar explotando la fuente de néctar de manera individual, esto es sin informar al resto de la colonia sobre ella.
- Continuar explotando la fuente de néctar pero informando a la colonia sobre ella. Para esto realiza un baile, que es observado por las abejas desocupadas, donde les comunica la localización de la fuente.

La decisión que una abeja toma de manera individual depende en gran medida de la calidad de la fuente. Si la calidad no compensa el esfuerzo, la abeja la abandona y se incorpora a la zona de

[5] "Self-Organization in Biological Systems" [18] pag.193

observación. Si la calidad de la fuente es suficiente para ser explotada pero no es lo suficientemente buena como para informar de ella, la abeja la explota en solitario. Por último si la calidad de la fuente es suficiente para ser explotada y para informar de ella, la abeja lo comunicará al resto de la colonia mediante un baile. Dependiendo de la calidad de la fuente, la duración del baile será mayor o menor.

Las abejas desempleadas ocupan posiciones de observación en la zona de baile, esperando que alguna recolectora les informe acerca de una fuente que merezca ser explotada. Las abejas desocupadas no se fijan en todas las abejas que estén bailando, sino que atienden a una sola, el criterio de elección es por azar. Es decir, normalmente la primera abeja bailarina que ven es escogida.

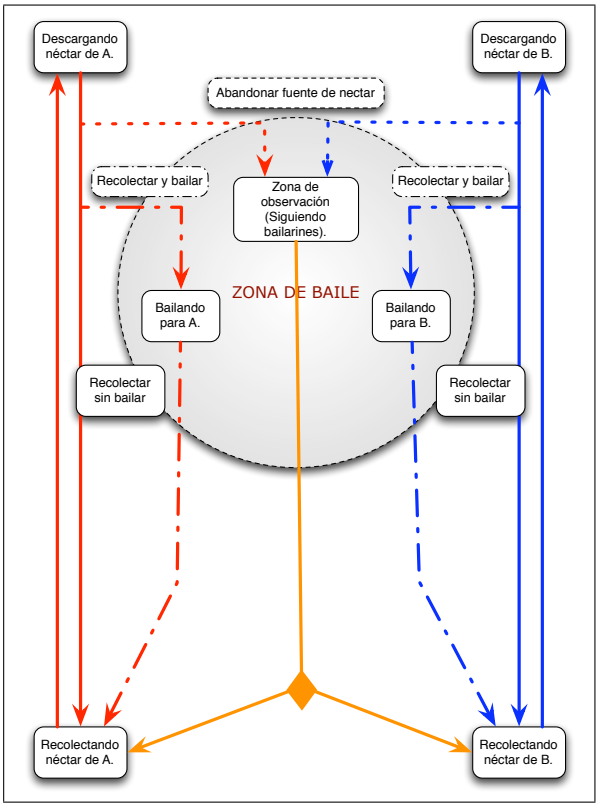


Figura 4.10: Diagrama de selección de fuentes de néctar por la abeja melifera.

Selección colectiva de las fuentes de alimento por parte de las hormigas

También existen experimentos de selección colectiva por parte de las hormigas ([18] pas. 219), en estos experimentos (figura 4.11) se les presentan a las hormigas dos fuentes de comida con distinta concentración de azúcar. Cuando la diferencia de concentración era suficientemente grande, la colonia explotaba la que mayor concentración presentaba.

El mecanismo que funciona es el mismo que en el ACO, la cantidad de feromona depositada en el camino es proporcional a la calidad de la solución. Por tanto el camino que lleva a la mejor fuente presentará una concentración de feromona mayor y será escogido en mayor proporción que el otro.

Finalmente el proceso converge y las hormigas recorren mayoritariamente un único camino: el que lleva a la fuente con mejor concentración.

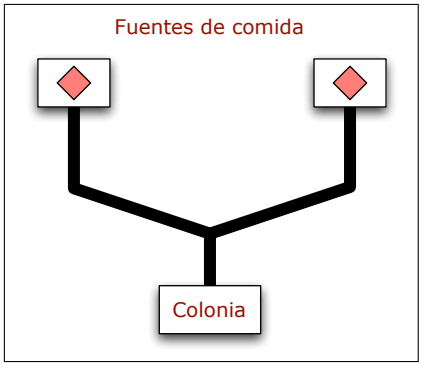


Figura 4.11: Experimento de selección de fuentes de comida por las hormigas.

Aunque en principio los dos mecanismos de selección colectiva, el de las abejas y el de las hormigas, son muy parecidos. Ambos están guiados por la calidad de la comida, existe una diferencia considerable. Si a las hormigas una vez que han escogido una fuente de comida, se cambian las concentraciones; de tal modo que la fuente escogida sea la de peor calidad, las hormigas continuaran explotándola. Por el contrario si a las abejas se les cambia la calidad en las fuentes de néctar, reaccionan cambiando la fuente escogida. Es decir, siempre explotan la de mejor calidad.

Esto es interesante desde el punto de vista de la optimización porque el mecanismo de las abejas muestra una gran flexibilidad a la hora de cambiar la solución adoptada. Es decir, es capaz de escapar de mínimos locales si se presenta una alternativa mejor. La diferencia clave la explican los autores del libro "Self-Organization in Biological Systems":

> *Bees do not compare dancers.* This seemingly small difference is responsible for the rather different properties of honey-bee recruitment system versus the recruitment system of many ants. As has been shown, bees are always able to select the most rewarding nectar source even if many weaker food sources have been discover previously[6].

Desde el punto de vista de la computación evolutiva y la optimización el no comparar soluciones tiene una consecuencia interesante:

- Explotación varías soluciones simultáneamente: en los Algoritmos Genéticos y en el ACO existe competición de soluciones, en los Algoritmos Genéticos la función de selección, en el ACO la construcción de la distribución de probabilidad. En ambos casos se selecciona de todas las posibles soluciones solamente una.

  Esto garantiza la convergencia del método pero tiene el inconveniente de que la exploración se reduce a una única zona, la cual puede ser una zona de mínimos locales. Por el contrario en las abejas no se da esta convergencia hacia una solución única, sino que son capaces de manejar varias zonas de búsqueda simultáneamente de manera independiente.

[6] "Self-Organization in Biological Systems" [18] pag.246

En nuestro sistema incorporamos este mecanismo de las abejas. La capacidad de manejar *independientemente* distintas zonas de búsqueda sin que exista elección entre ellas. El mecanismo para incorporar esta característica a los algoritmos de hormigas es producto de varios factores:

1. Perdida de sincronismo.
2. Función de evaluación que provee de una convergencia acelerada.
3. Criterio de evaluación dinámico para medir la calidad de las soluciones.

Generamos zonas de búsqueda independientes de la siguiente manera:

- Sea $H(n)$ el subconjunto de hormigas generado en la iteración n.
- Sea $F(n)$ el estado de la tabla de feromona en la iteración n.
- Sea $Z(n)$ la zona de búsqueda marcada por el estado de la tabla de feromona en la iteración n.
- Sea $\bigtriangledown$ la operación de actualización de la tabla de feromona.
- Iteración n : $H(n) \xrightarrow{F(n)} Z(n)$
- Iteración n^{+1}: $H(n^{+1}) \xrightarrow{F(n^{+1})} Z(n^{+1})$
  - $F(n^{+1}) = \bigtriangledown[F(n), H(n^{-\alpha})]$ donde $H(n^{-\alpha}) \xrightarrow{F(n^{-\alpha})} Z(n^{-\alpha})$
  - Si $Z(n) \cap Z(n^{-\alpha}) \approx \phi$
  - Dependiendo de la calidad y el número de soluciones encontradas en $Z(n^{-\alpha})$ se tendrá que:
    $$Z(n^{+1}) \approx Z(n) \vee Z(n^{-\alpha})$$

Debido a la dinámica del ACO, las zonas de búsqueda compiten entre sí. Se expulsa de la tabla, reduciendo su probabilidad aproximadamente a cero, aquellas zonas donde existen menor número de soluciones o estas son de calidad inferior.

- Iteración m^{+1} : $F(m)$ tal que $F(m) \cap F(n) \approx \phi$
- $F(m^{+1}) = \bigtriangledown[F(m), H(n^{-\alpha})]$
  donde $H(n^{-\alpha})$ es la población que comenzó su búsqueda en el instante n: $H(n) \xrightarrow{F(n)} Z(n)$
- Dependiendo de la calidad y el número de soluciones encontradas por $H(n^{-\alpha})$ en $Z(n)$ se tendrá que:
  $$Z(m^{+1}) \approx Z(m) \vee Z(n)$$

Una zona de búsqueda expulsada de la tabla de feromona por la dinámica competitiva, no se pierde. Sino que permanece almacenada en la población de hormigas que se encuentran explorando dicha zona, existiendo la posibilidad de que se reincorpore a la tabla.

En las abejas cuanto mejor sea una solución, más tiempo durará el baile, de este modo la abeja reclutará más recolectoras para su zona. En nuestro caso ocurre igual. Las zonas de exploración con el tiempo serán expulsadas de la tabla de feromona ya que las sucesivas poblaciones de hormigas irán incorporando sus respectivas zonas. Cuanto más calidad tengan las soluciones de una determinada zona, mayor tiempo sobrevivirá en la tabla, luego más hormigas explorarán esa zona.

Como todas las hormigas no actualizan la tabla de feromona al mismo tiempo, se consigue que la comparación entre soluciones no se haga de manera global, sino que cada solución se compara únicamente con las que se hallan generado en la misma iteración. En cada instante la tabla de feromona refleja la probabilidad de buscar en una región del espacio determinada. Cada población de hormigas que actualice la tabla cambiará la zona de exploración marcada en la tabla por la suya propia. De este modo se consigue que todas las zonas de exploración no se encuentran simultáneamente en la tabla, las zonas permanecen independientemente unas de otras en el sistema, a través de las poblaciones que las instancian.

Por último *la calidad que se le asigna a una solución no es fija, sino que dependen del resto de soluciones encontradas. De este modo se solventa el problema que presenta el criterio de las abejas: la no convergencia. El método consiste en modificar el criterio de selección a nivel de individuo*.

Simplemente se fuerza a que los individuos abandonen las zonas de búsqueda si no las encuentran atractivas. En las abejas su método de selección no converge porque el criterio de decisión de cada abeja permanece estático. En nuestro el criterio de evaluación depende de la media de la calidad de las soluciones encontradas, por lo que a medida que se van descubriendo nuevas soluciones el limite fijado por la media ira aumentado, lo que impide que las zonas de búsqueda menos atractivas vean su probabilidad aumentada, lo que conlleva que la zona sea abandonada.

Nos hemos decidido a utilizar la media como método de evaluación dinámica porque ofrece una componente de inercia a la convergencia del algoritmo. Ya que a medida que se vayan descubriendo soluciones, el ritmo de variación de la media se ira reduciendo, cada solución individual tendrá menos peso al ser el conjunto de soluciones totales mayor. La ventaja fundamental es que en las primeras iteraciones no ofrece casi resistencia a la convergencia del algoritmo, permitiendo una primera discriminación de las zonas de búsqueda. Pero a medida que la búsqueda esta más madura, es más difícil variar la media. Si esto lo unimos a que es el criterio de discriminación –individual– mediante el cual una región de búsqueda se continua explorando o por el contrario se desecha, el resultado es que el ritmo de discriminación entre regiones se va frenando a medida que avanza la exploración. Lo cual permite un mejor conocimiento de las regiones de búsqueda, por tanto se puede evitar mejor caer en zonas de mínimos locales.

Por supuesto el sistema no esta a salvo de convergencias que acaben en mínimos locales, pero ofrece ventajas respecto a otros algoritmos de hormigas. El sistema no presenta la exigencia de los algoritmos de hormigas donde se fuerza a elegir entre distintas soluciones aunque estas sean muy similares, nuestro sistema es capaz de manejarlas simultáneamente sin verse forzado a escoger una única zona de búsqueda. Cuando la diferencia en la calidad de las soluciones encontradas, en distintas zonas aumenta, entonces el sistema tiende a converger. De este modo se puede evitar mínimos locales debidos a una pronta convergencia hacia zonas donde no esta muy clara la diferencia, por así decirlo, nuestro sistema "intenta asegurase", que en el caso de zonas aparentemente equivalentes, "escoge" siempre la mejor.

4.3.6. Algoritmos para el control de poblaciones

Otro aspecto del sistema algorítmico propuesto es que el número de hormigas no es fijo, sino que varía según el estado del propio algoritmo. La única restricción que se establece es un limite máximo de individuos, que pueden estar ejecutándose simultáneamente. Por tanto es necesario introducir un mecanismo de control de poblaciones para manejar este aspecto.

El Alimento disponible y la distribución de poblaciones de búsqueda

Es conocida la capacidad de los sistemas biológicos para auto-organizarse [18], adquieren la estructura más adecuada para realizar una tarea dependiendo del entorno. La misma idea es la que aquí se persigue, que el propio sistema se auto-organice. En este caso significa distribuir los individuos, los tipos de hormiga según el entorno. Este tipo de control basado en el entorno fue propuesto originalmente por R. Brooks [15, 16, 17], en nuestro caso el entorno es el propio estado del sistema.

Gordon advierte, en su trabajo [14], que las hormigas únicamente recolectan cuando tienen necesidad de alimento y no existen otras tareas más urgentes que atender. En nuestro sistema podemos identificar dos tareas:

1. Explorar el espacio de búsqueda para incorporar nuevas soluciones.
2. Buscar entre las soluciones disponibles la óptima.

La primera se lleva a cabo fundamentalmente por las hormigas exploradoras, mientras que la segunda por las hormigas recolectoras. El planteamiento para determinar que tarea debe ser predominante en el sistema es similar a las hormigas reales: se priorizan las tareas y se actúa según las necesidades.

Creación de necesidades:

1. Se crea una variable que representa el alimento de la colonia. La necesidad de comida vendrá marcada por la siguiente ecuación :

$$Necesidad = 1 - \frac{Alimento \quad Disponible}{Total \quad Hormigas}$$

   La necesidad aumenta con el número total de hormigas y disminuye con el alimento disponible.

2. El sistema crece, en número de hormigas, hasta alcanzar su tamaño limite.

Procedimiento que controla la dinámica de la población:

- Sea *asignar* el número de hormigas que han vuelto a la colonia tras alcanzar su objetivo.

Algorithm 3 Dinámica general de la colonia: Asignación de Tareas

```
if random < necesidad then
  return ← true
end if
```

Algorithm 4 Dinámica general de la colonia: Crecimiento

```
if random > necesidad ∧ tamaño < limite then
    return ← true
end if
```

Algorithm 5 Dinámica general de la colonia

```
while true do
    if Expansion(necesidades, limite) then
        asignar ← asignar + 1
    end if
    if isempty(asignar) then
        break
    else
        comida ← comida − 1
        if Asignacion(necesidades) then
            recolectoras ← [recolectoras  nuevaHormiga]
        else
            exploradoras ← [exploradoras  nuevaHormiga]
        end if
    end if
    necesidades ← comida / hormigas
end while
```

Tomando como punto de partida el valor de la necesidad en un instante concreto. Si al azar generamos un valor mayor que la necesidad creamos una nueva hormiga. Dicha hormiga deberá ser asignada a las exploradoras o las recolectoras, según el valor de la necesidad se asignará a un grupo u otro. La generación de un valor al azar marca el grupo correspondiente, por debajo de la necesidad la asignamos a las recolectoras y por encima a las exploradoras.

En caso de que no se genere una nueva hormiga, se evalúa las hormigas que han alcanzado el objetivo para reasignarlas. Aquellas hormigas que no han alcanzado el objetivo, simplemente se desechan. Si no existen hormigas para su reasignación, se termina el procedimiento de control de poblaciones.

Algorithm 6 Dinámica general de la colonia: Recolección de alimento

```
if isRecolectora(hormiga) ∧ alcanzadoObjetivo(hormiga) then
    comida ← C + K * f(t_hormiga, μ, δ)
end if
```

La comida disponible aumenta de manera proporcional a la calidad de las trayectorias solución recorridas por las hormigas recolectoras.

Reglas simples, comportamientos complejos El comportamiento global del sistema podía describirse como:

- El sistema intentará expandirse hasta alcanzar el límite de hormigas disponibles siempre que tenga las necesidades cubiertas. Es decir, se priorizan necesidades: lo más importante es sobrevivir, con la supervivencia asegurada la población intenta crecer.
- Se equilibra el ritmo de exploración frente a la recombinación de soluciones:
  - Si la recombinación de soluciones no ha convergido, muchas recolectoras incorporaran poca comida a la colonia, lo que aumenta la necesidad de alimento. Se prioriza la generación de hormigas recolectoras frente a la exploradoras para conseguir un flujo constante de alimento.
  - Si la recombinación ha convergido, todos los esfuerzos se centran en la exploración ya que el flujo de alimentos hacia la colonia es constante.
- Se minimiza el riesgo de la pronta convergencia hacia las primeras soluciones. Cuanto mejor es la solución, más cantidad de comida introducirá en la colonia la recolectora lo que disminuye las necesidades. Por tanto más hormigas se asignaran a la exploración para intentar encontrar una solución alternativa a la descubierta.

4.3.7. Auto-Organización de la búsqueda

El método de optimización "Branch and Bound" (Ramificación y Poda) [20, 21] organiza el espacio de estados, lo restringe para que la búsqueda se lleve siempre a cabo en un espacio mínimo. La idea básica es la siguiente:

- Sea P el problema que se quiere resolver.
- Sea RP una versión relajada del problema que se quiere resolver.
- Sea S el espacio definido por las restricciones de un problema.
- Sea f la función de coste a optimizar.

Entonces:

1. $S(P) \subseteq S(RP)$
2. $RP_i \rightarrow f(solucion(RP_i)) < f(solucion(RP_j)) \quad \forall RP_j \in C = \{RP_1, RP_2, ..., RP_n\}$

$\Rightarrow solucion(P) \subseteq S(RP_i)$

El método consiste en construir versiones relajadas del problema. Donde una versión relajada es el mismo problema pero con menos restricciones activas. La solución del problema real, con todas las restricciones activas, se encontrará dentro del espacio de soluciones de la versión relajada que mejor solución presente.

Un planteamiento similar puede aplicarse en el problema de optimización que estamos resolviendo.

- Sea S_n la región del espacio, cuyos bordes son alcanzables en un tiempo t_n y $Objetivo \in S_n$
- Sea T una trayectoria cualquiera.

Entonces:

1. $t_i < t_j \Rightarrow S_i \subseteq S_j$
2. $C = \{T | T \in S_i\}$

$\Rightarrow T_{óptima} \in C$

Dada una trayectoria cualquiera que sea solución –que satisfaga todas las restricciones–, con un tiempo de recorrido t. Si existe una trayectoria mejor en tiempo, podemos asegurar que esta se encuentra dentro de la región del espacio cuyos bordes son alcanzables en un tiempo t. Por tanto la búsqueda queda restringida a dicha zona espacial, no hay necesidad de buscar en el espacio total.

Introducimos la siguiente regla en el sistema:

Algorithm 7 Limitación de la búsqueda

```
if Tiempo(hormiga) > μ then
  eliminar(hormiga)
end if
```

El limite de exploración, tanto para las recolectoras como para las exploradoras, se fija en la media de tiempos de todas las trayectorias-solución generadas. De este modo se consigue reducir la búsqueda a una zona más pequeña del espacio, acelerando la convergencia del algoritmo y asegurando una mejor cobertura del espacio mediante las búsquedas. Se consigue que el sistema se auto-organice, estableciendo las regiones de búsqueda, existen regiones que nunca van a ser exploradas: aquellas que caigan fuera de los limites establecidos.

Se opta por la media y no por el tiempo de la mejor solución por lo siguiente:

1. Fijar el limite al tiempo de la mejor solución invalida el carácter evolutivo del sistema. Establecer dicho criterio significa que el sistema solo puede manejar una solución: la mejor. Aunque existan múltiples búsquedas en paralelo, se invalida la recombinación y la selección colectiva mencionadas anteriormente.
2. De lo anterior se sigue que aumenta el riesgo de convergencia hacia mínimos locales, ya que se impide al sistema a manejar múltiples zonas de búsqueda. Se elimina la competición entre zonas de búsqueda, ya que la búsqueda solo tiene lugar en la zona marcada por la mejor solución.

4.3.8. Algoritmo Completo

La figura 4.12 muestra el diagrama de funcionamiento del algoritmo completo. Se comienza por una inicialización, que consiste en la definición de las condiciones iniciales y finales de la maniobra que se pretende planificar. Además se suministran los parámetros de control de algoritmo: número máximo de hormigas, número de iteraciones (tanto para el autómata como para el ACE) y la probabilidad de que una hormiga exploradora utilice el autómata o la feromona.

A continuación se inicializa el autómata según las condiciones del objetivo y se hace evolucionar. La evolución termina cuando el autómata se ha estabilizado o se ha alcanzado el limite de iteraciones. Este proceso devuelve una matriz de orientaciones.

Una vez que se ha construido la matriz de orientaciones comienza el funcionamiento del ACE. Como se puede observar en la figura los tres módulos se comunican a través de la población de hormigas, recibiendo como entrada la población y actuando sobre la misma.

El primer módulo es el de construcción de la solución, recibe como entrada las hormigas que todavía no han alcanzado el objetivo y su función consiste en realizar un paso en la construcción de la solución mediante la resolución de la dinámica del barco. Dependiendo del tipo de hormiga, recolectora o exploradora, se asignan los valores de consigna según el contenido de la feromona o la matriz de orientaciones calculada previamente por el autómata, en el caso de que no exista información en la tabla de feromona se utiliza la solución propuesta por el autómata celular. Una vez que se han calculado los valores de consigna se procede a la construcción de la solución, se resuelve el modelo del barco según el estado almacenado en la hormiga y los valores de consigna. Por último se actualiza el estado de la hormiga y se devuelve de nuevo a la población.

El segundo módulo consta de dos fases, en la primera se procede a actualizar la media de tiempos para las soluciones encontradas y el tiempo de la mejor solución. Una vez terminada la primera fase comienza la segunda, que se encarga de actualizar la feromona empleando las hormigas que han alcanzado el objetivo y la comida disponible en el caso de que se trate de hormigas recolectoras. Una vez que ha finalizado la actualización, se eliminan las hormigas que hayan alcanzado el objetivo o superado el tiempo de búsqueda límite.

El tercer módulo recibe como entrada el numero de individuos de la población, si está por debajo del limite fijado comienza el proceso de creación de hormigas. Se van creando sucesivas hormigas en función de la comida disponible y el tamaño de la población. Obviamente las nuevas hormigas se incorporan a la población, en el estado inicial definido.

El proceso del ACE finaliza cuando se alcanza el número máximo de iteraciones fijado.

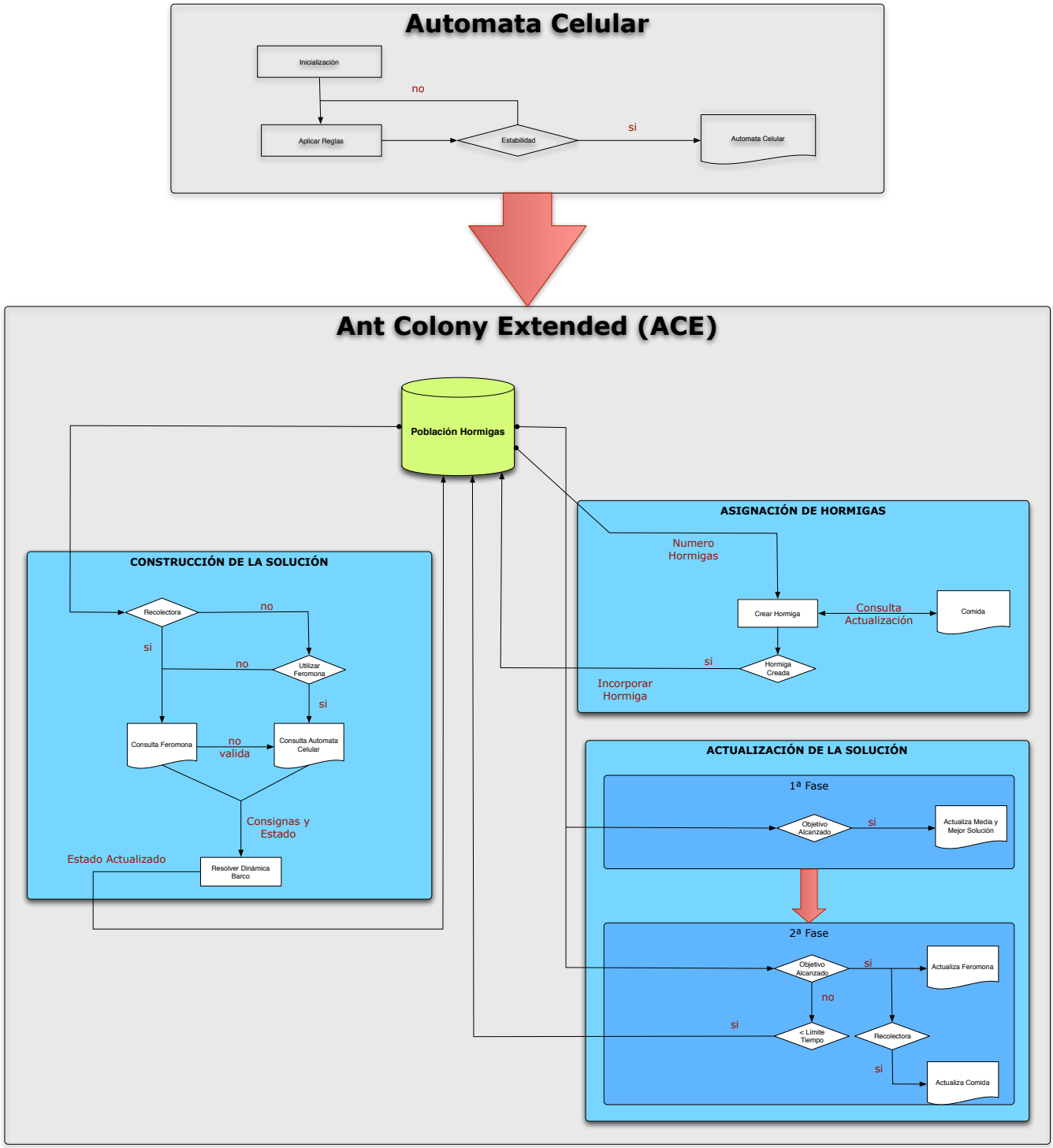


Figura 4.12: Diagrama de flujo del algoritmo completo.

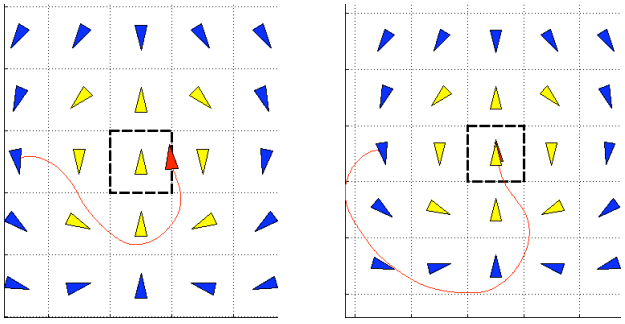
Capítulo 5

Análisis de resultados

5.1. Ejemplo de ejecución: Zonas independientes de búsqueda

Se ha escogido este caso en particular para ejemplificar el método de resolución porque existe un mínimo local, la figura 5.1 muestra cual es una posible solución frente al mínimo local. El sistema escapa del mínimo local gracias a que es capaz de mantener la exploración en varias zonas de búsqueda.

Cada cuadrícula representa una región del espacio de 10x10. Sobre cada casilla se sobreimpresiona el estado autómata correspondiente. La cuadrícula objetivo esta delineada en negro, se considera alcanzado el objetivo cuando se satisfacen las siguientes condiciones: la posición discreta del barco es la correspondiente a la región delineada, la desviación respecto a la orientación pedida es igual o menor a 4º y la velocidad de llegada presenta una desviación de 1m/s. En el caso de que la velocidad no este restringida, obviamente la última condición no se comprueba.



(a) Posible Solución (b) Mínimo Local

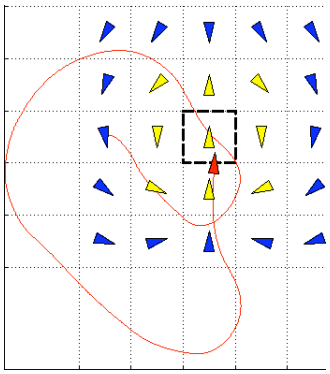
Figura 5.1: Mínimo local vs Solución Optima

En el presente ejemplo, el estado de partida se caracteriza porque el barco está situado en el origen, con una orientación de 90º respecto al eje de las X, y en reposo. Se pretende realizar una maniobra para alcanzar un estado objetivo, en el que en la posición final corresponda con el punto de coordenadas X=20m e Y=0m, una orientación de 90º y una velocidad de llegada no restringida.

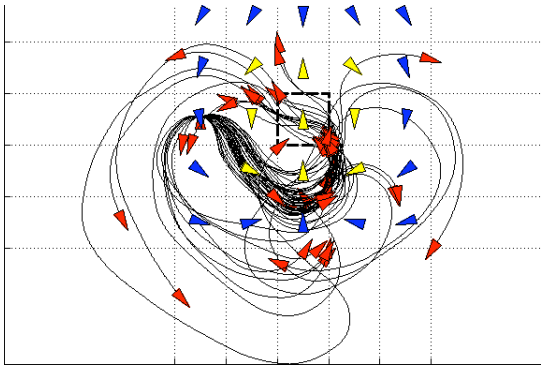
A continuación se muestra un ejemplo de evolución del ACE para el objetivo definido([$X = 20m, Y = 0m, \theta = 90^{o}, Velocidad = --$]). Los colores negro y rojo, en las figuras correspondientes al estado, representan –respectivamente– a hormigas exploradoras y recolectoras.

En las primeras iteraciones tiene lugar las búsquedas dirigidas por el autómata. Donde en cada iteración se genera una hormiga exploradora nueva. Cuando aparece la primera solución, comienza el proceso.

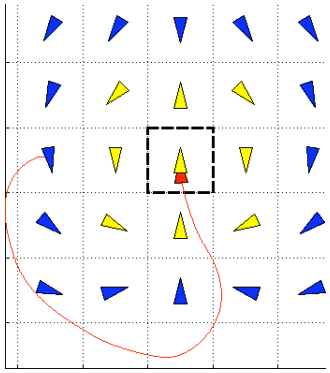
La primera solución (figura 5.2a) aparece en la iteración 32, en cada iteración las hormigas se mueven a lo largo de una cuadrícula. En la iteración 38, como se puede observar en la figura 5.2b, el ACE crea múltiples hormigas exploradoras que buscan mayoritariamente en la zona marcada por la solución encontrada –giran a derechas–, pero gracias a que el seguimiento de la feromona es parcial existe un reducido grupo de hormigas que parte hacia otra zona de búsqueda –giran a izquierdas–. Gracias a este comportamiento en la iteración 55, figura 5.2c, la solución encontrada es otra, bastante mejor que la anterior.



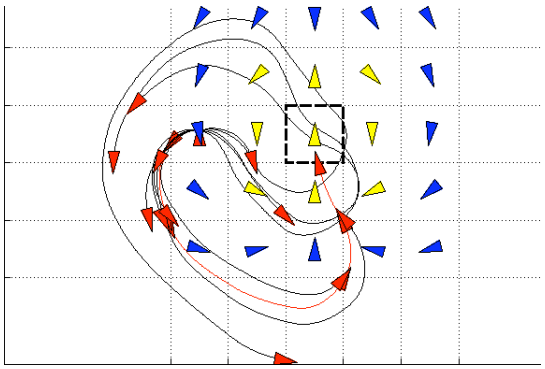
(a) Iteración 38: Mejor Solución



(b) Iteración 38: Estado del Proceso



(c) Iteración 55: Mejor Solución



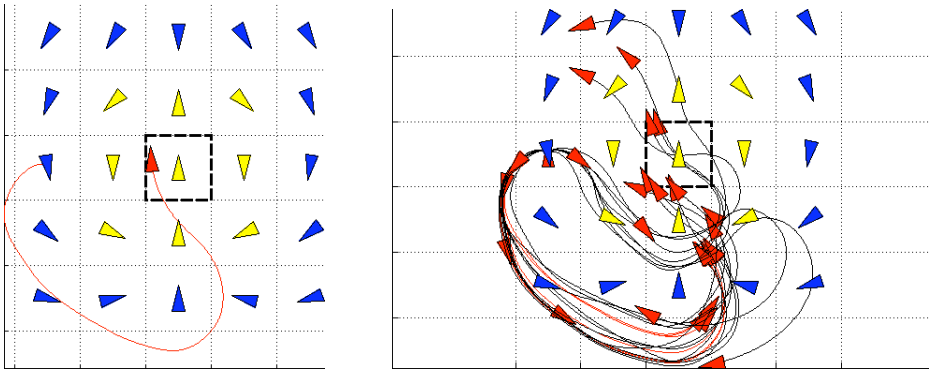
(d) Iteración 55: Estado del Proceso

Figura 5.2

En la figura 5.2d se puede observar como las hormigas recolectoras –trayectoria roja– esta marcando la zona de la solución, el proceso se esta desplazando hacia la zona marcada por la nueva solución. Aunque la búsqueda según la solución anterior no está totalmente descartada porque todavía siguen

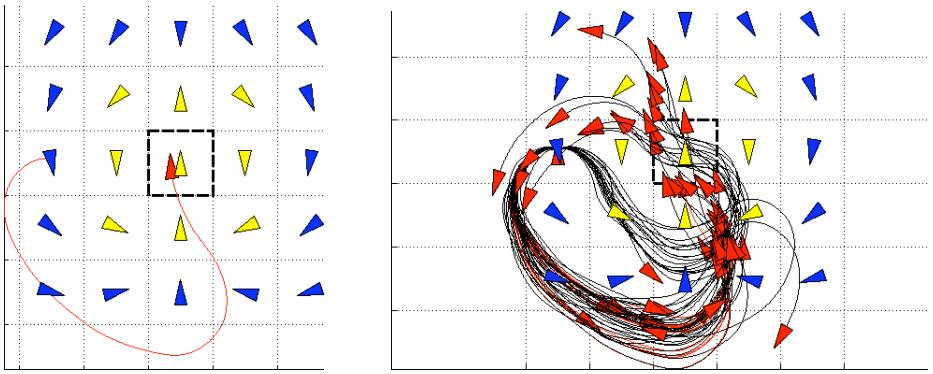
existiendo hormigas que recorren la zona.

Hay que señalar que el ACE comienza con una cierta comida disponible que gasta en las primeras exploraciones, con lo cual hasta que no se encuentra una solución estable, que pueda ser recorrida frecuentemente por las recolectoras, la ganancia de comida será escasa. Por tanto la población de hormigas disponible para la búsqueda será pequeña, ésta ira creciendo a medida que la solución se estabilice. A esta razón se debe la reducida población de hormigas que aparece en el iteración 55, figura 5.2d.



(a) Iteración 68: Mejor Solución

(b) Iteración 68: Estado del Proceso



(c) Iteración 80: Mejor Solución

(d) Iteración 80: Estado del Proceso

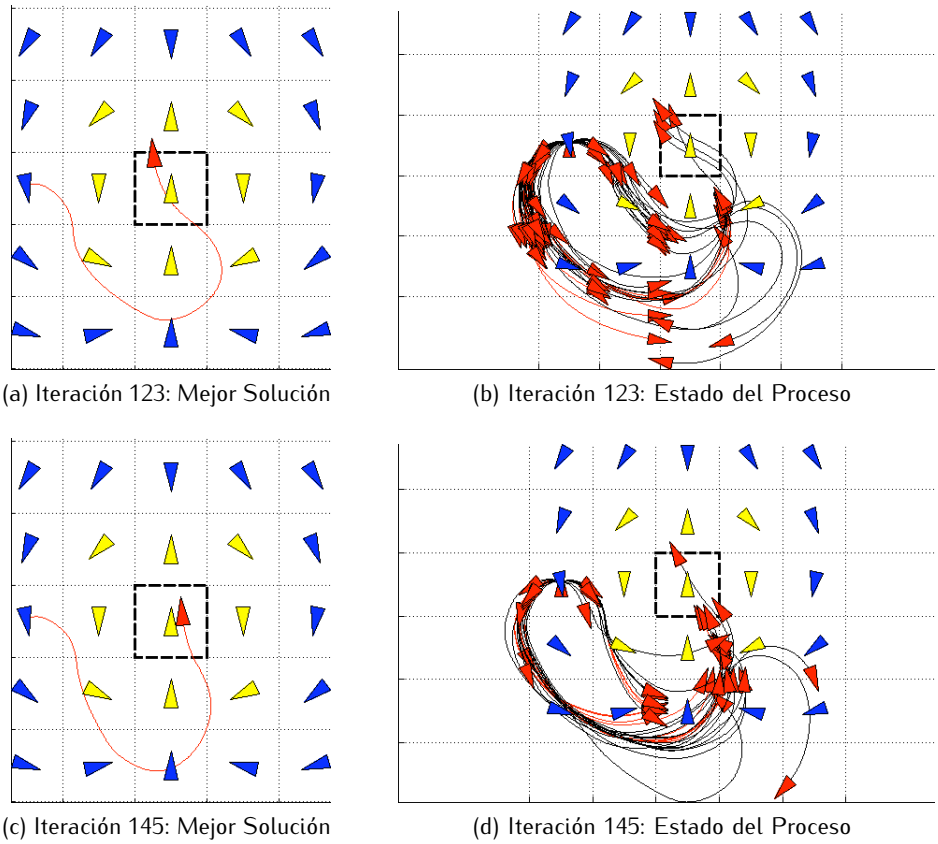
Figura 5.3

En la iteración 68, figura 5.3b, la solución se ha mejorado y el proceso se ha estabilizado un poco. Gracias a esto se puede observar que la población de hormigas ha crecido, aún así el ratio de exploradoras–recolectoras es relativamente bajo. Se puede observar como la zona donde se encontraba la solución anterior dispone todavía de hormigas exploradoras.

Ya en la iteración 80, figura 5.3b, el proceso es mucho más estable como demuestra el tamaño de la población de hormigas. La solución continua mejorando y se puede ver como la zona donde se encontraba la solución previa no ha sido descartada para la exploración. Es decir, siguen existiendo hormigas que patrullan esa zona en busca de una mejor solución, aunque las recolectoras recorren únicamente la nueva zona de exploración.

Esto se debe primero a que las zonas de búsqueda son independientes, únicamente coinciden en el punto de partida, por tanto la información generada acerca de esa zona de búsqueda no se pierde. Puede ser reutilizada por las hormigas exploradoras que ignoren la feromona.

En segundo lugar el ratio de exploradoras–recolectoras ha aumentado, se generan muchas más hormigas exploradoras por pocas recolectoras. De tal modo la probabilidad de que una recolectora escoja la zona anterior –correspondiente a la solución de la iteración 38– es muy escasa, ya que los valores de feromona de esta zona son bajos comparados con la otra –correspondiente a la solución de la iteración 80–. Las exploradoras al existir en mayor cantidad aumenta la probabilidad de que exploren la zona marcada con una valor más bajo de feromona.



(a) Iteración 123: Mejor Solución

(b) Iteración 123: Estado del Proceso

(c) Iteración 145: Mejor Solución

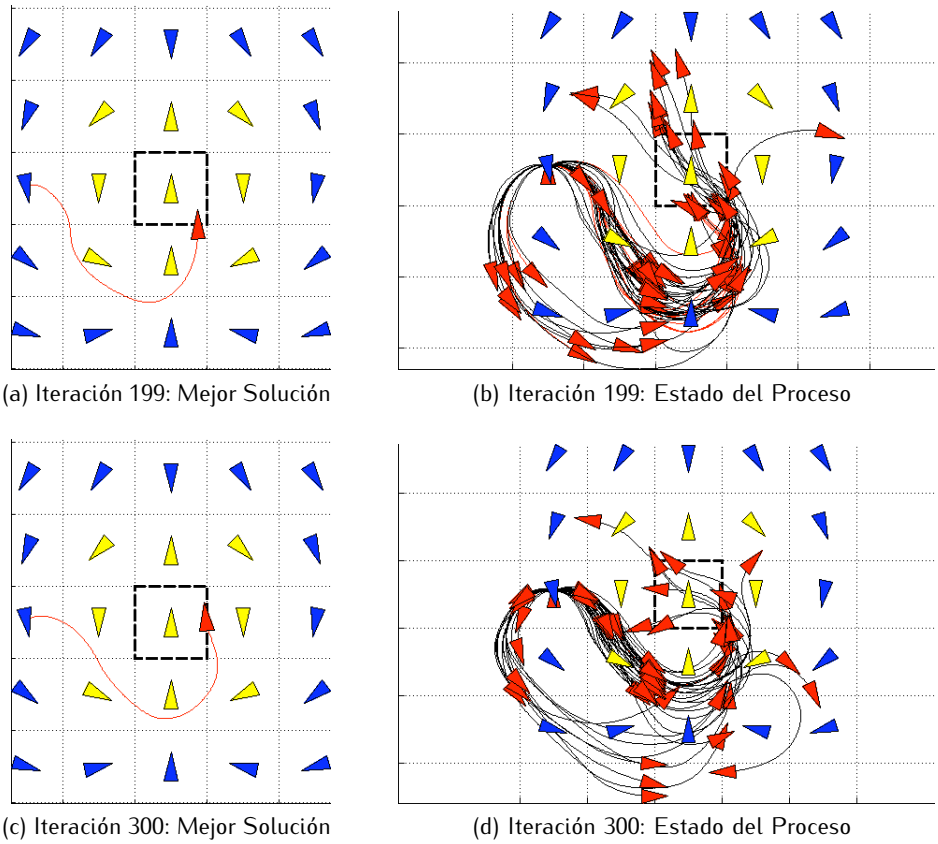
(d) Iteración 145: Estado del Proceso

Figura 5.4

En la iteración 123 aparece una nueva solución,figura 5.4a, en la zona de búsqueda conocida debido a la solución alcanzada en la iteración 38. Como se puede observar en la figura 5.4b la zona más poblada sigue siendo la de la solución anterior –solución en la iteración 80–. Aunque el número de exploradoras donde ha aparecido la nueva solución es considerable. Esto es producto a que ambas zonas se han explorado en paralelo, aunque las mejores soluciones perteneciesen a la zona de la solución alcanzada en la iteración 80, el conocimiento de la otra zona ha sido mejorado hasta el punto de dar con una solución alternativa.

Las exploradoras que han generando trayectorias–solución inferiores a la mejor solución conocida, han ido incorporando a la tabla dichas trayectorias. De tal modo que han ido reclutando y manteniendo una población de hormigas que explore la zona correspondiente a la solución en la iteración 38, donde se encuentra también la mejor solución actual.

Ya en la iteración 145, figura 5.4c y figura 5.4d, la situación ha cambiado: el ACE ha reconocido la zona donde se encuentra la mejor solución y cada vez destina más hormigas a esa zona. Como se puede observar existen hormigas recolectoras en ambas regiones, lo que indican que la estabilidad del proceso está cambiando en favor de la nueva zona, como este desplazamiento es gradual se pueden explotar ambas zonas simultáneamente. Además se puede observar que la inestabilidad provocada por la aparición de soluciones en una zona de búsqueda nueva tiene influencia en la población de hormigas, ésta ha disminuido a consecuencia de que ha disminuido el flujo de comida.



(a) Iteración 199: Mejor Solución

(b) Iteración 199: Estado del Proceso

(c) Iteración 300: Mejor Solución

(d) Iteración 300: Estado del Proceso

Figura 5.5

En la iteración 199, figura 5.5a, se puede apreciar como la ultima solución aparecida – solución en la iteración 145– se ha ido mejorando y como la zona de búsqueda marcada por ésta se ha hecho predominante. La figura 5.5b muestra como todavía la zona anterior no ha desaparecido del proceso, sigue siendo explotada por hormigas recolectoras y exploradoras. Por así decirlo, el sistema ha destinado muchos recursos en ella, la ha explorado durante más de 100 iteraciones, tiene que

"asegurarse" que puede descartarla sin riesgo a equivocarse. Es decir, la resistencia al descarte es proporcional al trabajo invertido.

La iteración 300, figura 5.5c y figura 5.5d, no ofrece mucha variación respecto a la 199, la solución continúa mejorando y la nueva zona de búsqueda sigue siendo predominante mientras la anterior ha pasado ha un segundo plano. Se puede observar por la ausencia de hormigas recolectoras que la estabilidad del sistema esta bastante avanzada, las reservas de comida deben estar totalmente cubiertas para que no haya hormigas recolectoras.

En este ejemplo puede verse como el sistema es capaz de evitar lo mínimos locales, gracias a que es capaz de mantener en exploración varias zonas simultáneamente. Esto se consigue gracias a que cuando aparece una solución nueva, esta altera la estabilidad del proceso ya que el flujo de comida está vinculado a la calidad de la solución. Por tanto si se está explotando una solución, ha alcanzado un equilibrio entre la población y los recursos que dicha solución le provee; el que aparezca una solución mejor altera los recursos disponibles de la solución que se esta explotando actualmente, lo que conlleva una rotura de la estabilidad. El sistema intenta recuperarla, para lo cual ambas soluciones compiten entre sí, la mejor solución proveerá de más recursos por tanto podrá mantener una mayor población de búsqueda en su zona. Pero la solución desplazada conservará recursos y también podrá mantener una población de búsqueda. A medida que la diferencia entre soluciones vaya creciendo el sistema convergerá a una u otra, en los casos de situaciones equivalentes la convergencia es muy lenta e incluso ambas zonas pueden explotarse simultáneamente.

También puede verse claramente como el sistema se auto-organiza, reacciona a las perturbaciones que le genera la aparición de nuevas soluciones y la competición entre zonas de búsqueda. El sistema concretamente reacciona a la inestabilidad, reacciona ajustando –disminuyendo– el número de hormigas para evitar que la inestabilidad aumente. A medida que va recuperando la estabilidad la población de hormigas aumenta hasta alcanzar el límite fijado. Esta auto-organización es una de las características que exhiben los sistemas complejos.

5.2. Comparación con soluciones analíticas

Cuando se diseñan métodos de cálculo por aproximación es necesario comprobar si efectivamente el método obtiene la solución en los casos conocidos, es una forma de asegurar que se esta operando de manera correcta al menos en las situaciones conocidas. De este modo aumentan las garantías de que efectivamente se estén obteniendo soluciones correctas para casos desconocidos. Realizar esta comparación es de todo punto necesaria, pues únicamente en situaciones conocidas se puede evaluar si efectivamente el método se comporta adecuadamente. En nuestro caso existen soluciones óptimas conocidas, aquellas donde la trayectoria solución es la línea recta.

Para evaluar si el sistema alcanza la solución optima en los casos conocidos definimos el siguiente objetivo: $[X = 0, Y = \delta, \theta = 90^{o}, Velocidad = --]$. El estado de partida es el de reposo, el barco se sitúa en el origen de coordenadas (0,0) y con una orientación de 90º. Así variando la distancia al objetivo (δ) en la dirección del eje Y podemos plantear distintos escenarios.

Para calcular la solución analítica, se fijan respectivamente las consignas de rumbo y velocidad a 90º y la velocidad a la máxima disponible. A continuación se resuelve el modelo hasta que el barco alcanza la región del espacio discreto donde esta definido el objetivo. A partir de esta solución se obtiene el tiempo de la solución analítica.

Se han llevado a cabo 90 ejecuciones independientes para cada escenario, cada una de ellas con un limite de 1000 iteraciones y una población máxima de 200 individuos. Como error aceptable permitimos una desviación de 4º y 1 m/s respectivamente de las condiciones prefijadas como objetivo. Así obtenemos los tiempos correspondientes a la mejor solución encontrada de entre las 90 ejecuciones y la media de las 90 soluciones obtenidas.

Para ilustrar el procedimiento, en la figura 5.6, se sobreimpresionan las 90 soluciones para la situación donde la distancia al objetivo es de 50m. El código de colores varía del azul al rojo, indicando la calidad de una solución respecto del resto, siendo el azul la peor calidad y el rojo la mejor.

En la figura 5.7 se muestra una comparación entre los resultados obtenidos mediante todas las ejecuciones independientes. La primera gráfica muestra el número de soluciones obtenidas, la segunda el tiempo(s) de la mejor solución obtenida y la tercera el tiempo(s) de la peor solución obtenida en cada ejecución.

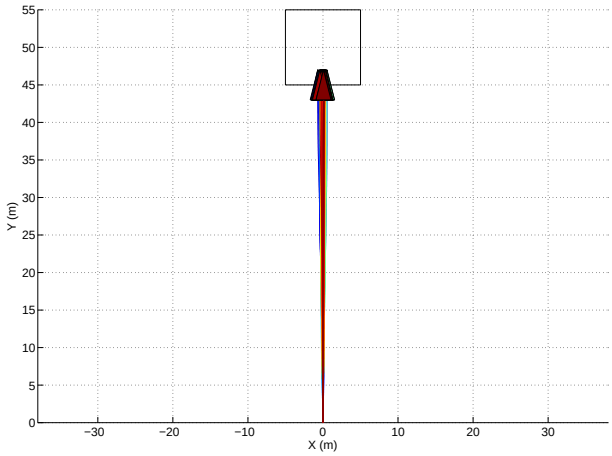


Figura 5.6: 90 mejores soluciones para el caso de $posición = (0,50), \theta = 90, Velocidad = --$

En la tabla 5.1 y en la figura 5.8 se puede ver el resultado de la comparación. A partir de estos se puede observar como en muchos casos la mejor soluciones coincide con la solución calculada de manera analítica y la media de las soluciones es muy próxima a la solución analítica, sobre todo en distancias cortas (<100m). El error cometido aumenta en las distancias largas debido a que el numero de decisiones aumenta y por tanto la optimización es más difícil. Por lo que es posible que el número de iteraciones y el limite de población, prefijados, fuesen insuficientes en situaciones que presentan distancias largas ($\geq 100m$).

Se puede concluir que el sistema funciona correctamente, es capaz de obtener soluciones muy próximas al óptimo en situaciones conocidas. Además el hecho de que la media de las soluciones se aproxime al óptimo indica que el proceso es convergente, ya que cada ejecución parte de una primera solución distinta.

Por tanto podemos considerar –en principio– fiables la soluciones que obtengamos para escenarios donde la solución no es conocida de antemano.

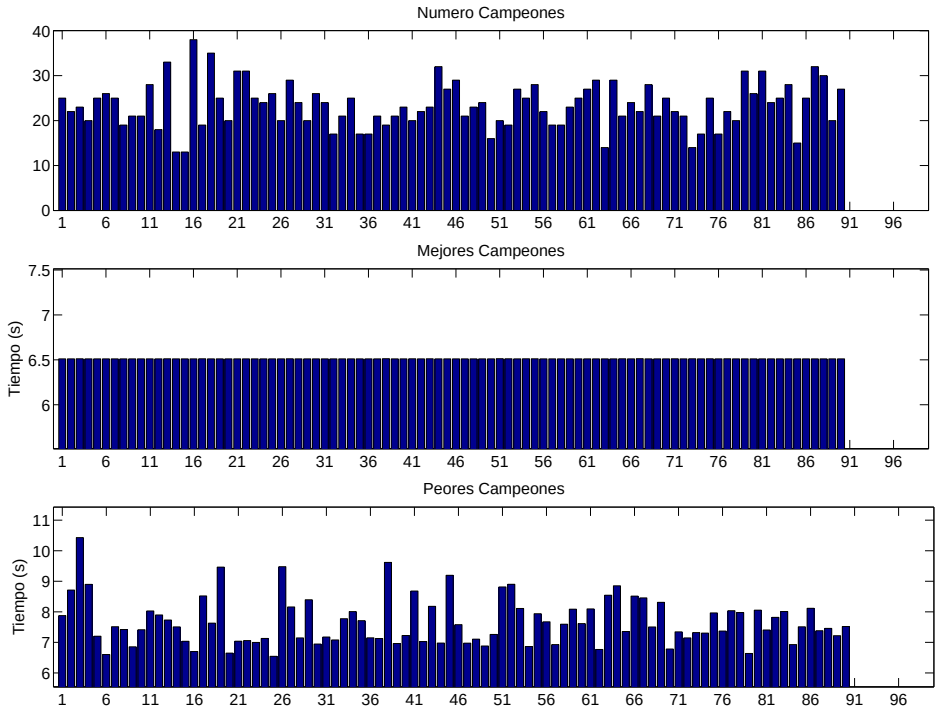


Figura 5.7: Comparación de 90 ejecuciones para el caso de $posición = (0,50), \theta = 90, Velocidad = --$

Cuadro 5.1: Comparación en tiempo(s) con la solución analítica para diferentes distancias(m)

Distancia(m)	Tiempo Medio(s)	Tiempo Mejor(s)	Tiempo Analítico(s)
20	3,56806466714380	3,56806466714380	3,56815339908185
50	6,51095464481695	6,51017655778067	6,51068202325209
70	8,09169121271584	8,08861360319354	8,08848123460212
100	10,2756573732549	10,2639010758927	10,2591967851685
150	13,8243150718650	13,7443282462004	13,7128103697154
200	17,4550793048750	17,2847582653421	17,1497683823365

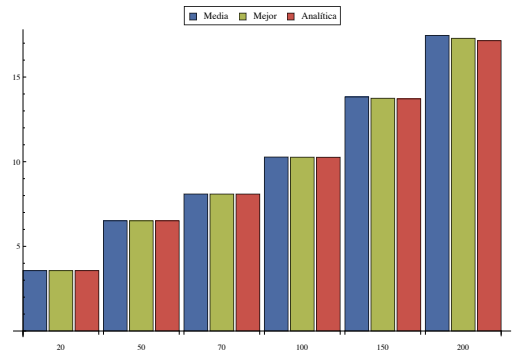


Figura 5.8: Comparación en tiempo(s) con la solución analítica para diferentes distancias(m)

5.3. Casos de estudio

5.3.1. Experimento aislado

La condiciones de cada experimento se han fijado para todos los casos de estudio a 1000 iteraciones y una población limite máxima de 200 hormigas. A su vez permitimos una desviación de 4º en el rumbo objetivo y 1 m/s en la velocidad objetivo. La posición objetivo se considera alcanzada cuando el barco llega a la región del espacio discreto correspondiente con la posición objetivo. En todos los casos el estado de partida se fija en el origen, con una orientación de 90º y partiendo del reposo.

La figura 5.9 ilustra un experimento con el siguiente objetivo: $posición = (0, -20), orientación = 90^0, velocidad = --$. El código de colores varía del azul al rojo, indicando la calidad de una solución respecto del resto, siendo el azul la peor calidad y el rojo la mejor. En la figura, las primeras soluciones son las trayectorias exteriores, a medida que el proceso continúa se encuentran mejores soluciones: las trayectorias interiores.

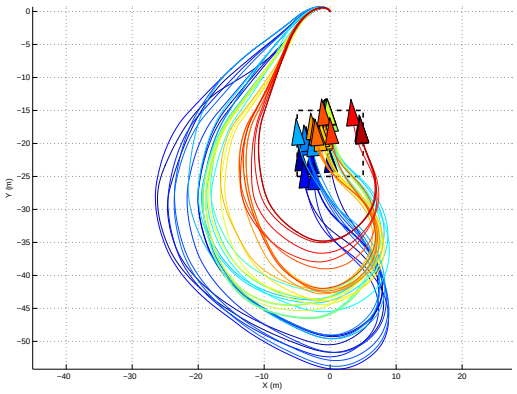


Figura 5.9: Trayectorias obtenidas para un experimento aislado, con el objetivo definido como: ($x = 0, y = -20, \theta = 90^o, velocidad = --$)

La figura 5.10 muestra los valores de la velocidad que lleva el barco, justo antes de que se tome una nueva decisión. Es decir, cada vez que el barco cambia de posición en el espacio discreto.

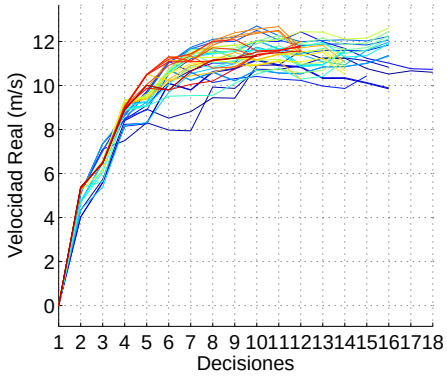


Figura 5.10: Evolución de la velocidad del barco para un experimento aislado, con el objetivo definido como: $(x = 0, y = -20, \theta = 90^o, velocidad = --)$

La figura 5.11 muestra la evolución de la población de hormigas durante el experimento.

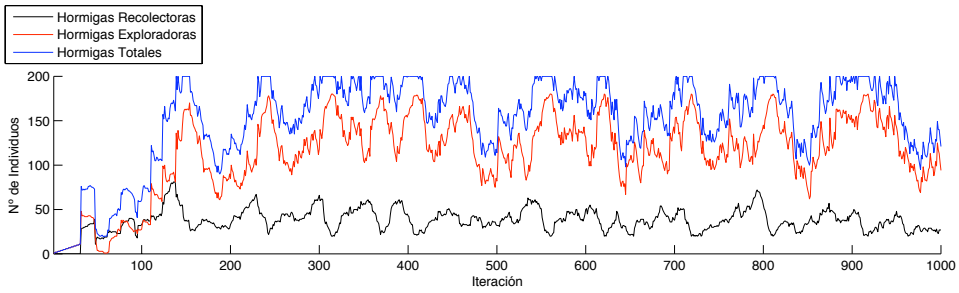


Figura 5.11: Evolución de la población de hormigas para un experimento aislado, con el objetivo definido como: $(x = 0, y = -20, \theta = 90^o, velocidad = --)$

5.3.2. Lote de experimentos

Para verificar la convergencia del método hacia la solución optima, se ejecutan lotes de experimentos para cada caso de estudio. Cada lote consiste en 100 experimentos aislados.

Posición=(0,-20),Orientación=90º,Velocidad=–

La figura 5.12 muestra las 20 mejores trayectorias obtenidas, para el mismo objetivo con el que se ejemplificó el experimento aislado. Obviamente, para el escenario particular propuesto existen dos soluciones óptimas simétricas –una correspondiente a girar a izquierdas, en la primera decisiones, y la otra girar a derechas–. El ACE es capaz de aproximar ambas.

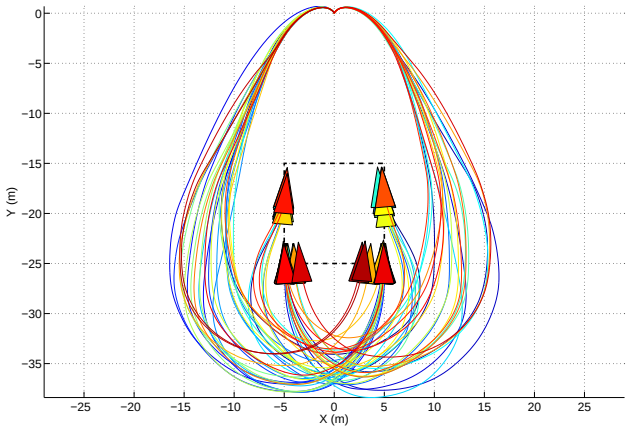


Figura 5.12: 20 mejores trayectorias obtenidas para un lote experimentos, con el objetivo definido como: $(x = 0, y = -20, \theta = 90^o, velocidad = --)$

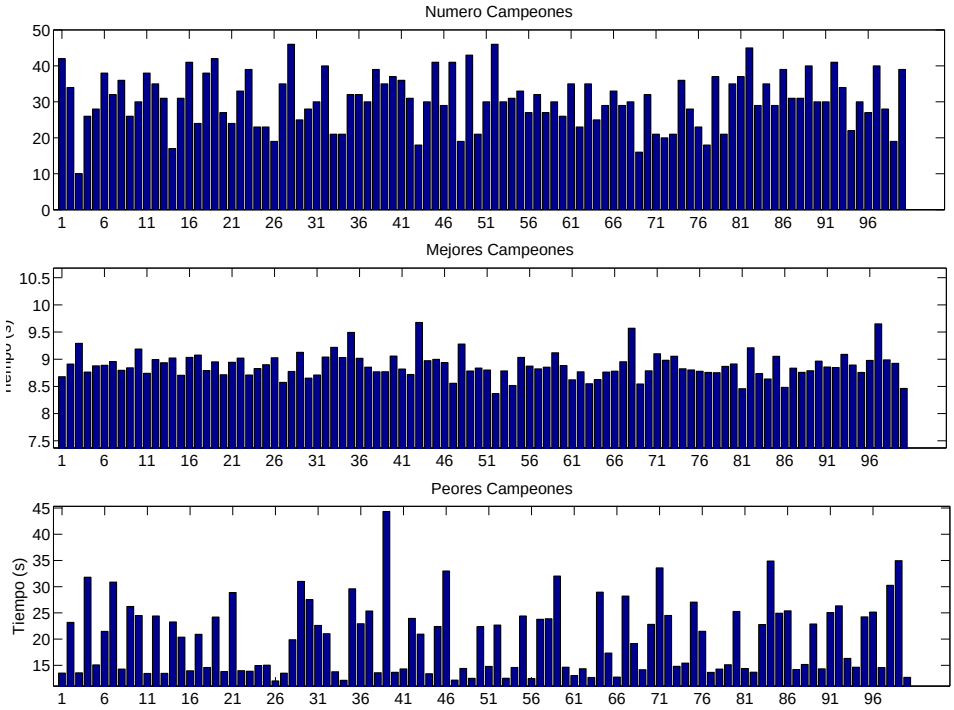


Figura 5.13: Comparación de los resultados obtenidos para un lote experimentos, con el objetivo definido como: $(x = 0, y = -20, \theta = 90^o, velocidad = --)$

La figura 5.13 muestra comparaciones del lote de experimentos, cada columna corresponde a un experimento aislado. La primera gráfica muestra el número de soluciones obtenidas, la segunda el tiempo(s) de la mejor solución obtenida y la tercera el tiempo(s) de la peor solución obtenida.

El cuadro 5.2 muestra las estadísticas correspondientes al lote de experimentos. Como puede ob-

Cuadro 5.2: Estadísticas del lote experimentos, para el objetivo definido como: $(x = 0, y = -20, \theta = 90^o, velocidad = --)$

	Nº de Soluciones	Media	Moda	Desviación Estándar	Mejor Solución
Mejores Soluciones	100	8.8856	8.3686	0.23308	8.3686
Peores Soluciones	100	20.0815	12.0445	6.9261	12.0445

servarse la desviación estándar en las peores soluciones es considerablemente mayor que en el caso de las mejores soluciones, esto indica que el proceso converge a soluciones similares independientemente de las condiciones iniciales –primera solución obtenida–. Señalar que la diferencia máxima entre las mejores soluciones es inferior al segundo.

Posición(0,20), Orientación = 270º y Velocidad = 4

La figura 5.14 muestra las 20 mejores trayectorias obtenidas. Obviamente, para el escenario particular propuesto existen dos soluciones óptimas simétricas –una correspondiente a girar a izquierdas, en la primera decisiones, y la otra girar a derechas–. El ACE es capaz de aproximar las dos.

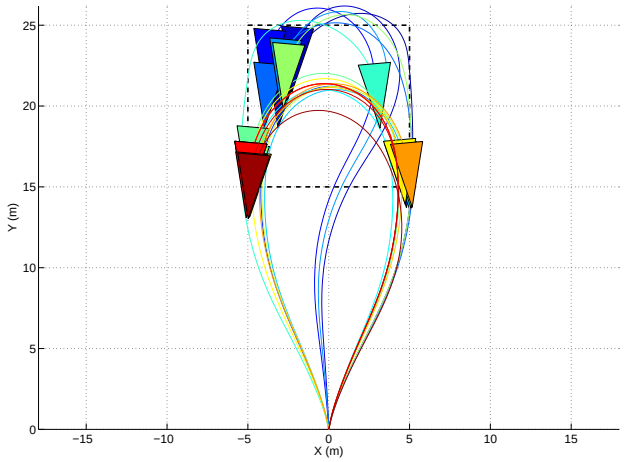


Figura 5.14: 20 mejores trayectorias obtenidas para un lote experimentos, con el objetivo definido como: $(x = 0, y = 20, \theta = 270^o, velocidad = 4)$

La figura 5.15 muestra la evolución de la velocidad del barco para las diferentes trayectorias (representadas en la figura 5.14), la velocidad se representa según cada instante de decisión que corresponde con una posición del espacio discreto –región de 10x10m del espacio real–.

La figura 5.16 muestra las consignas de velocidad obtenidas para cada trayectoria, se representan igualmente frente a instantes de decisión.

Como se puede observar la trayectorias mejores resultan de un manejo de la aceleración bastante intuitivo, consiste en acelerar bruscamente para luego frenar. Las hormigas realizan esto marcando

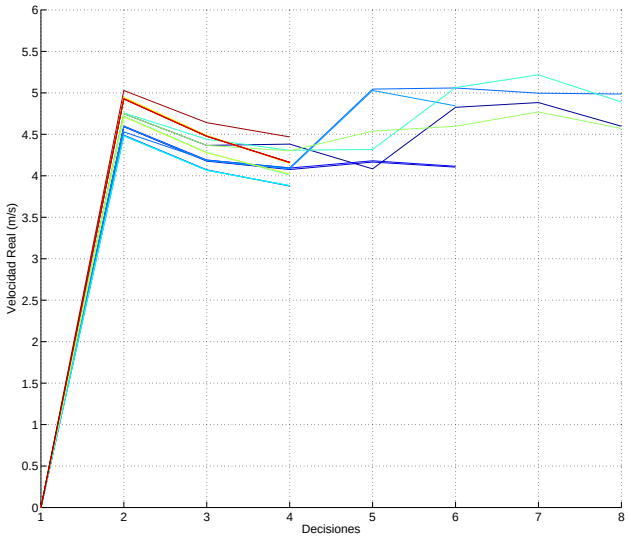


Figura 5.15: Velocidad del barco correspondiente a diferentes trayectorias, con el objetivo definido como: $(x = 0, y = 20, \theta = 270^o, velocidad = 4)$

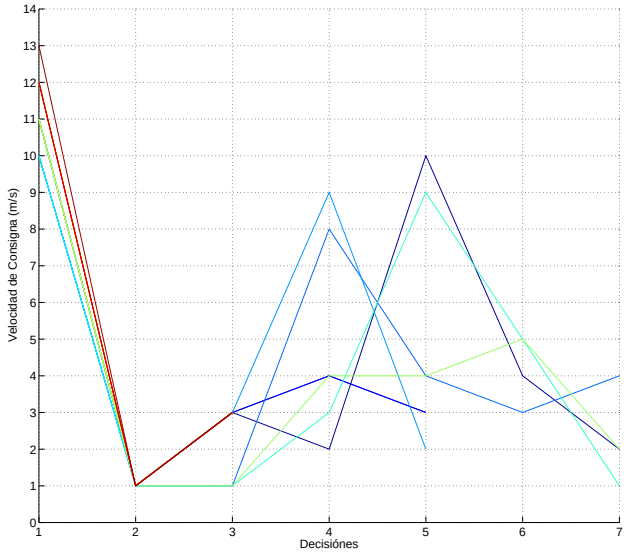


Figura 5.16: Consignas de velocidad obtenidas correspondientes a diferentes trayectorias, con el objetivo definido como: $(x = 0, y = 20, \theta = 270^o, velocidad = 4)$

como primera consigna una velocidad muy próxima a la máxima disponible. Para frenar la nave marcan como velocidad de consigna la mínima disponible y disminuyen la frenada marcando como última velocidad de consigna una muy próxima a la velocidad objetivo.

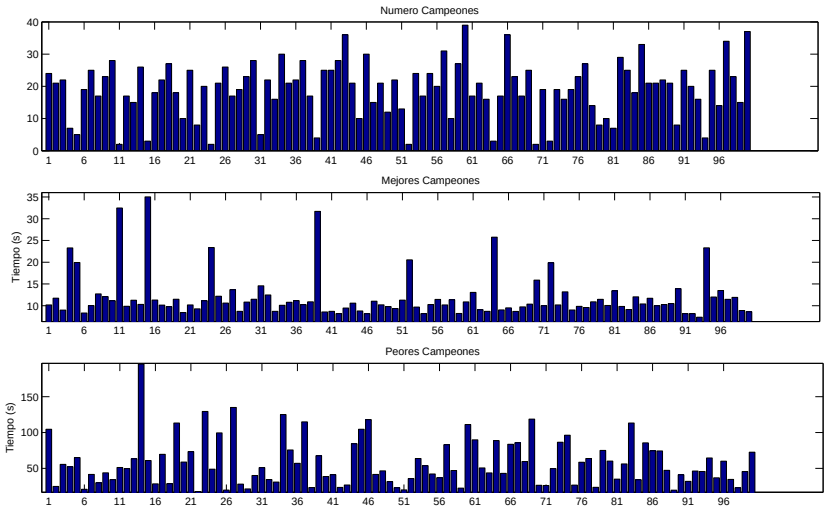


Figura 5.17: Comparación de los resultados obtenidos para un lote experimentos, con el objetivo definido como: $(x = 0, y = 20, \theta = 270^o, velocidad = 4)$

La figura 5.17 muestra comparaciones del lote de experimentos, cada columna corresponde a un experimento único. La primera gráfica muestra el número de soluciones obtenidas, la segunda el tiempo(s) de la mejor solución obtenida y la tercera el tiempo(s) de la peor solución obtenida.

Cuadro 5.3: Estadísticas del lote experimentos, para el objetivo definido como: $(x = 0, y = 20, \theta = 270^o, velocidad = 4)$

	Nº de Soluciones	Media	Moda	Desviación Estándar	Mejor Solución
Mejores Soluciones	100	11.9398	8.1831	5.0873	7.3659
Peores Soluciones	100	57.7725	17.6921	32.4745	17.6921

Se puede observar como en varios casos el número de campeones es muy reducido y a su vez la mejor solución es bastante mala, en comparación con el resto. En estos ocurre que la población de hormigas se ha extinguido en las primeras iteraciones, con lo que no han completado las 1000. Nos referimos a esta situación como experimento fallido. La extinción se produce porque no hay comida disponible y ninguna de las hormigas recolectoras consigue dar con trayectoria que alcance el objetivo, con lo que no existe posibilidad de que la población crezca. La recolectoras no alcanzan el objetivo en los casos donde se forman bucles en las trayectorias. Dichos bucles normalmente son corregidos

por las hormigas exploradoras, pero existen casos donde son complejos y requieren más tiempo para corregirlos. Durante la corrección del bucle no hay ganancia de alimento, luego si este proceso se retrasa la población se extingue.

La cuadro 5.3 muestra las estadísticas correspondientes al lote de experimentos. Como puede observarse los experimentos fallidos también han sido incluidos en la tabla. Aún así en la gráfica puede verse como los experimentos no fallidos presentan resultados similares.

Posición(0,100), Orientación = 270º y Velocidad = –

La figura 5.18 muestra las 100 mejores trayectorias obtenidas. De nuevo, para el escenario particular propuesto existen dos soluciones óptimas simétricas –una correspondiente a girar a izquierdas, en la primera decisiones, y la otra girar a derechas–. El ACE es capaz de aproximar ambas.

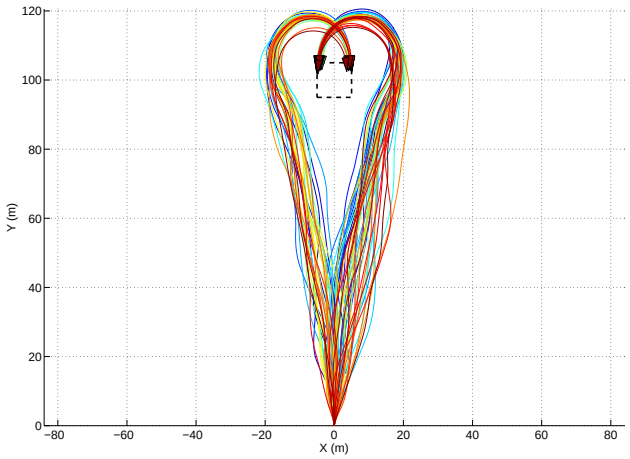


Figura 5.18: 100 mejores trayectorias obtenidas para un lote experimentos, con el objetivo definido como: $(x = 0, y = 100, \theta = 270^o, velocidad = --)$

Es interesante observar que todas las trayectorias consisten en trayectorias oblicuas para obtener un buen ángulo de giro cuando se llega a la zona de aproximación al objetivo. De este modo se consigue que el giro pueda realizarse a mayor velocidad y el resultado de la trayectoria sea más rápida que por ejemplo trayectorias que aproximen al objetivo en línea recta.

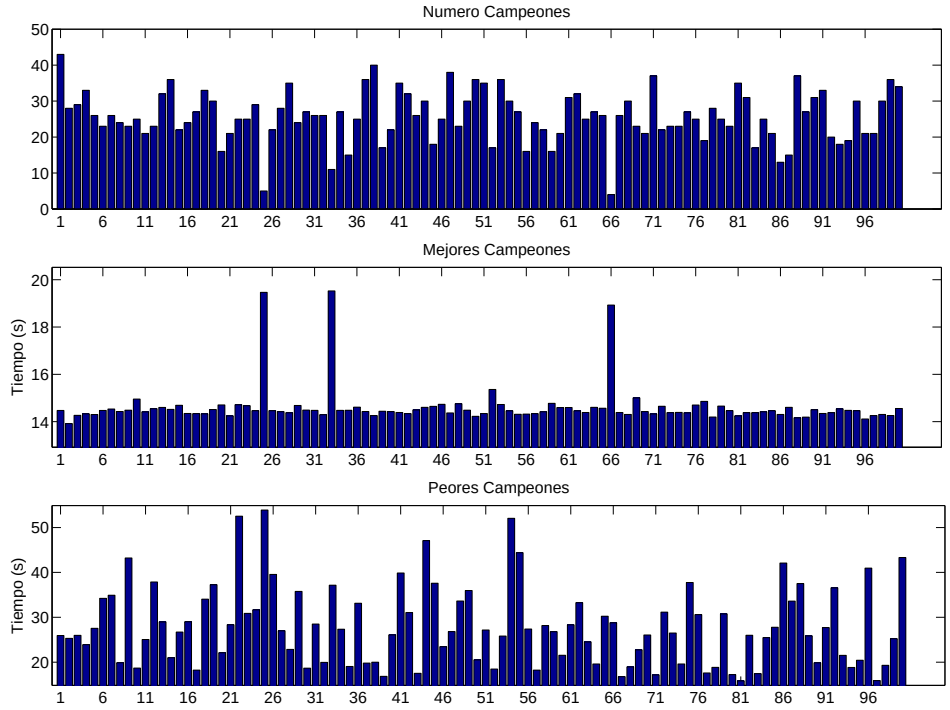


Figura 5.19: Comparación de los resultados obtenidos para un lote experimentos, con el objetivo definido como: $(x = 0, y = 100, \theta = 270^o, velocidad = --)$

La figura 5.19 muestra comparaciones del lote de experimentos, cada columna corresponde a un experimento único. La primera gráfica muestra el número de soluciones obtenidas, la segunda el tiempo(s) de la mejor solución obtenida y la tercera el tiempo(s) de la peor solución obtenida. Se puede observar la convergencia del sistema independientemente de la solución inicial encontrada y cómo existen unos pocos casos donde el sistema ha fallado.

Cuadro 5.4: Estadísticas del lote experimentos, para el objetivo definido como: $(x = 0, y = 100, \theta = 270^o, velocidad = --)$

	Nº de Soluciones	Media	Moda	Desviación Estándar	Mejor Solución
Mejores Soluciones	100	14.6126	13.9184	0.8541	13.9184
Peores Soluciones	100	27.9021	15.8373	8.7192	15.8373

El cuadro 5.4 muestra las estadísticas correspondientes al lote de experimentos.

Posición(0,-20), Orientación = 270º y Velocidad = –

La figura 5.20 muestra las 100 mejores trayectorias obtenidas. La figura 5.21 muestra las consignas

de rumbo correspondientes a las 100 trayectorias obtenidas, puede observarse perfectamente como queda capturada la simetría del escenario propuesto.

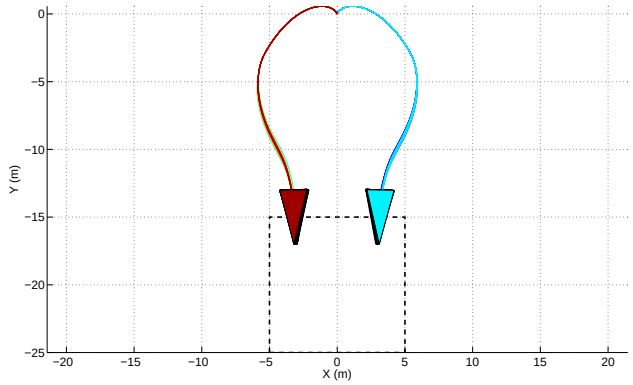


Figura 5.20: 100 mejores trayectorias obtenidas para un lote experimentos, con el objetivo definido como: $(x = 0, y = -200, \theta = 270^o, velocidad = --)$

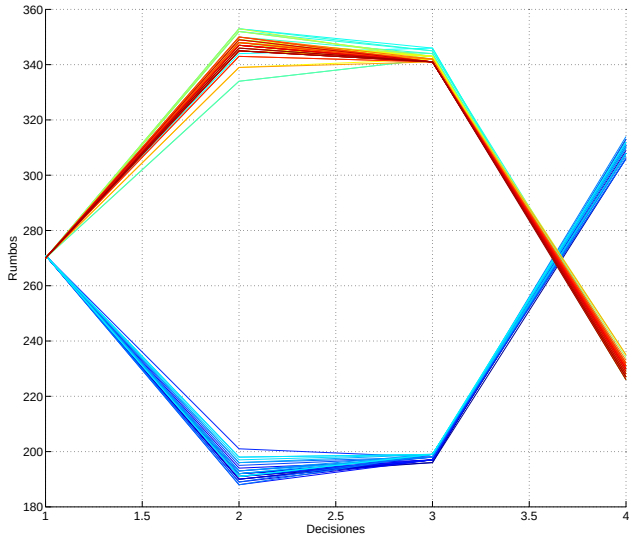


Figura 5.21: Consignas de rumbo obtenidas correspondientes a diferentes trayectorias, con el objetivo definido como: $(x = 0, y = -20, \theta = 270^o, velocidad = --)$

La figura 5.22 muestra comparaciones del lote de experimentos, cada columna corresponde a un experimento único. La primera gráfica muestra el número de soluciones obtenidas, la segunda el tiempo(s) de la mejor solución obtenida y la tercera el tiempo(s) de la peor solución obtenida. Se puede observar la convergencia del sistema, independientemente de la solución inicial encontrada, hacia soluciones prácticamente idénticas en tiempo.

Es interesante observar que de todo el lote de experimentos únicamente se obtienen dos trayectorias simétricas. Es decir, el sistema claramente converge, esto es importante porque debido a que

es capaz de mantener diferentes zonas de búsqueda se corría el riesgo de que el sistema no fuese convergente, evidentemente esto no es así. Además en caso de existir múltiples óptimos el sistema puede aproximar tanto uno como otro, lo que indica que no existe ningún sesgo. Por último señalar la calidad de las aproximaciones de los óptimos conseguidas para este caso.

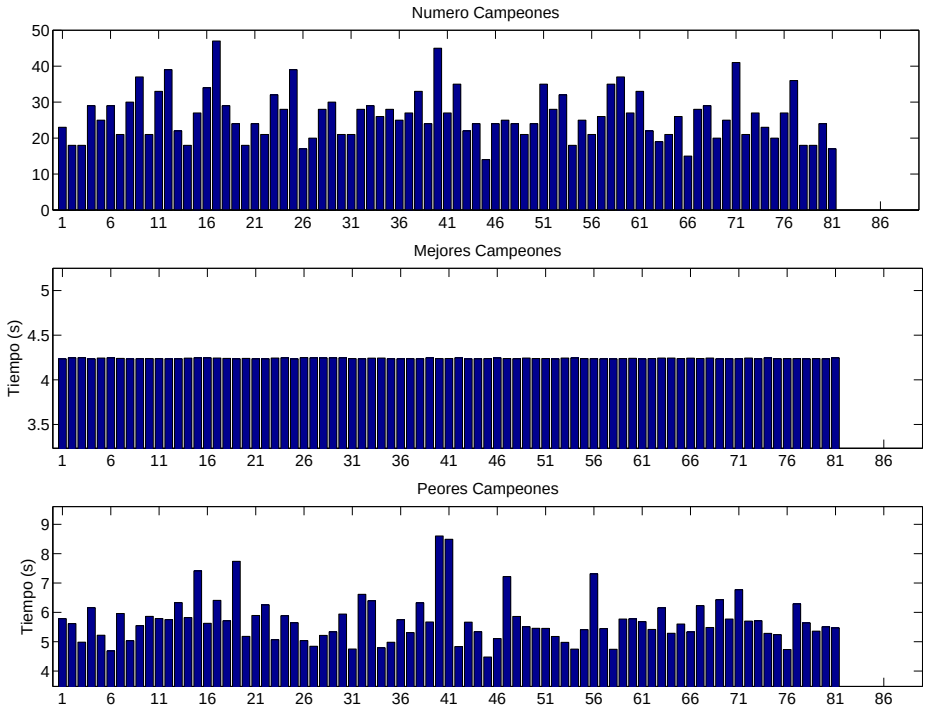


Figura 5.22: Comparación de los resultados obtenidos para un lote experimentos, con el objetivo definido como: $(x = 0, y = -20, \theta = 270^o, velocidad = --)$

Cuadro 5.5: Estadísticas del lote experimentos, para el objetivo definido como: $(x = 0, y = -20, \theta = 270^o, velocidad = --)$

	Nº de Soluciones	Media	Moda	Desviación Estándar	Mejor Solución
Mejores Soluciones	100	4.2397	4.2343	0.0045777	4.2341
Peores Soluciones	100	5.7147	4.4786	0.77938	4.4786

El cuadro 5.5 muestra las estadísticas correspondientes al lote de experimentos. Es interesante observar como la desviación estándar es mínima entre las mejores soluciones obtenidas. También se puede observar como las peores soluciones obtenidas están muy próximas a la solución mejor, esto indica la buena aproximación que provee el autómata a la hora de guiar las búsquedas para casos sencillos, ya que incluso las peores soluciones descubiertas son soluciones más que aceptables.

Posición(0,-100), Orientación = 90º y Velocidad = –

La figura 5.23 muestra las 100 mejores trayectorias obtenidas.

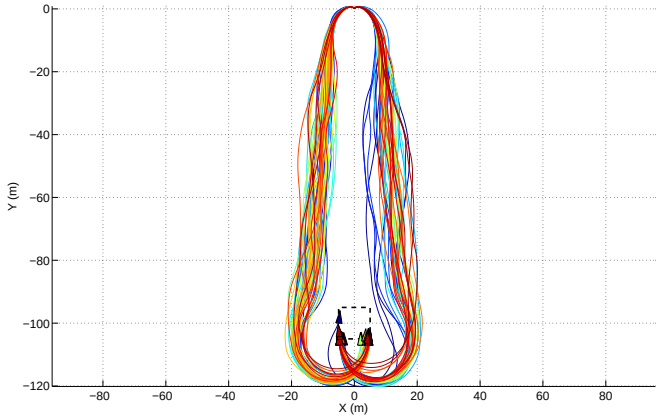


Figura 5.23: 100 mejores trayectorias obtenidas para un lote experimentos, con el objetivo definido como: $(x = 0, y = -100, \theta = 90^o, velocidad = --)$

La figura 5.24 muestra las consignas de las 100 trayectorias obtenidas, es interesante observar como la variación en las consignas para la parte de la trayectoria correspondiente a la navegación presenta mucha menor dispersión que la parte correspondiente a la maniobra.

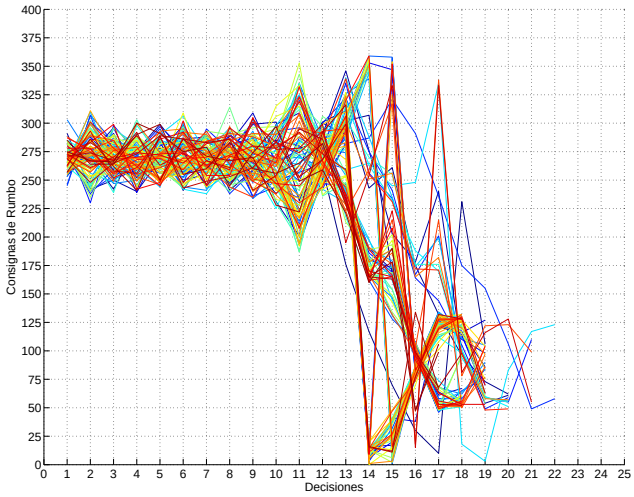


Figura 5.24: Consignas de rumbo obtenidas correspondientes a diferentes trayectorias, con el objetivo definido como: $(x = 0, y = -100, \theta = 90^o, velocidad = --)$

Al igual que ocurría con el caso "gemelo"[1] (Posición = (0,100), Orientación = 270º y Velocidad = –) todas las trayectorias consisten en trayectorias oblicuas para obtener un buen ángulo de giro cuando se llega a la zona de aproximación al objetivo. De este modo se consigue que el giro pueda realizarse

[1] La relación entre el estado final y el inicial no es la misma. En un caso es 90º–270º y en el otro 90º–90º

a mayor velocidad y el resultado de la trayectoria sea más rápida que por ejemplo trayectorias que aproximen al objetivo en línea recta.

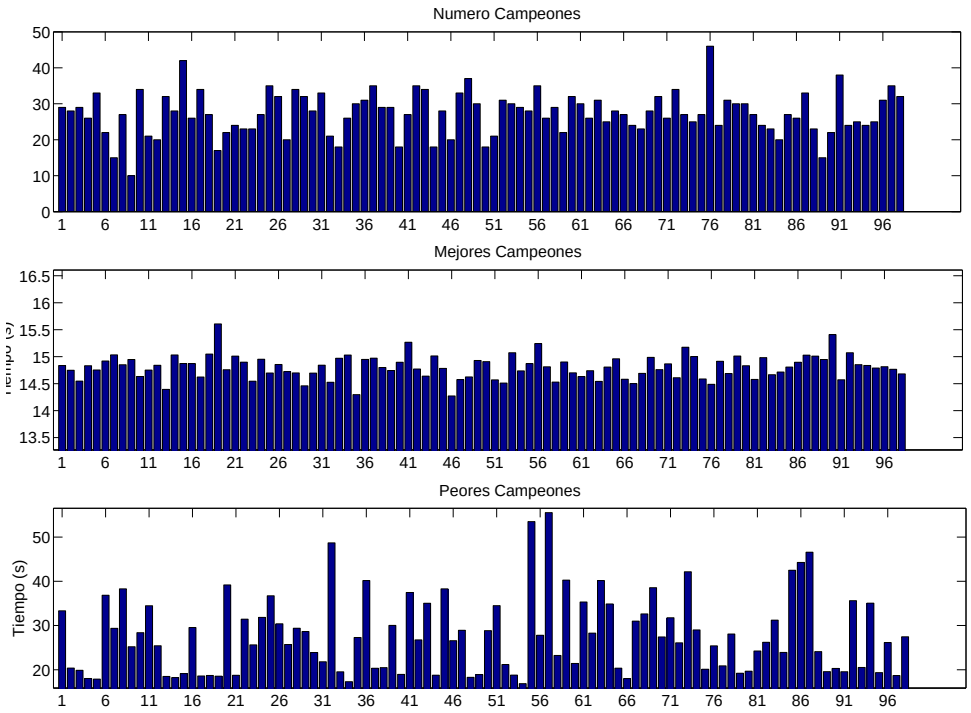


Figura 5.25: Comparación de los resultados obtenidos para un lote experimentos, con el objetivo definido como: $(x = 0, y = -100, \theta = 90^o, velocidad = --)$

La figura 5.25 muestra comparaciones del lote de experimentos, cada columna corresponde a un experimento único. La primera gráfica muestra el número de soluciones obtenidas, la segunda el tiempo(s) de la mejor solución obtenida y la tercera el tiempo(s) de la peor solución obtenida. Se puede observar la convergencia del sistema independientemente de la solución inicial encontrada.

Cuadro 5.6: Estadísticas del lote experimentos, para el objetivo definido como: $(x = 0, y = -100, \theta = 90^o, velocidad = --)$

	Nº de Soluciones	Media	Moda	Desviación Estándar	Mejor Solución
Mejores Soluciones	100	14.8062	14.2723	0.21858	14.2723
Peores Soluciones	100	27.752	16.8595	8.7059	16.8595

El cuadro 5.6 muestra las estadísticas correspondientes al lote de experimentos.

Hay que destacar que como era de esperar la solución obtenida en este escenario y su "gemelo" (Posición = (0,100), Orientación = 270º y Velocidad = –) es prácticamente idéntica. La diferencia

entre las medias de las mejores soluciones para ambos casos es mínima, la media de las mejores trayectorias en la "situación simétrica" era de 14.6126s y en el escenario actual de 14.8062s.

Posición(20,0), Orientación = 0º y Velocidad = –

La figura 5.26 muestra las 100 mejores trayectorias obtenidas. En este escenario se ve perfectamente como el sistema captura la única posible trayectoria óptima, todos los experimentos singulares convergen hacia una solución única.

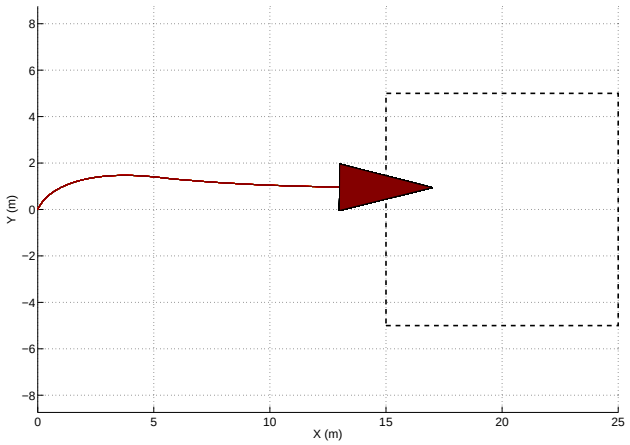


Figura 5.26: 100 mejores trayectorias obtenidas para un lote experimentos, con el objetivo definido como: $(x = 20, y = 0, \theta = 0^o, velocidad = --)$

La figura 5.27 muestra comparaciones del lote de experimentos, cada columna corresponde a un experimento único. La primera gráfica muestra el número de soluciones obtenidas, la segunda el tiempo(s) de la mejor solución obtenida y la tercera el tiempo(s) de la peor solución obtenida. Se puede observar la completa convergencia del sistema, independientemente de la solución inicial encontrada, hacia la solución única.

El cuadro 5.8 muestra las estadísticas correspondientes al lote de experimentos. Es interesante observar como la media coincide con la moda, para las mejores soluciones, y como la desviación estándar es cero. Si bien es cierto que la moda de las peores soluciones esta muy próxima al valor de las mejores soluciones, lo que indica la buena aproximación del autómata, hay que resaltar que el sistema incluye también al autómata, no es únicamente el ACE. Aún así aunque la solución primera sea muy próxima hay que ajustarla a la óptima, en ningún experimento singular la solución óptima se ha encontrado como primera solución.

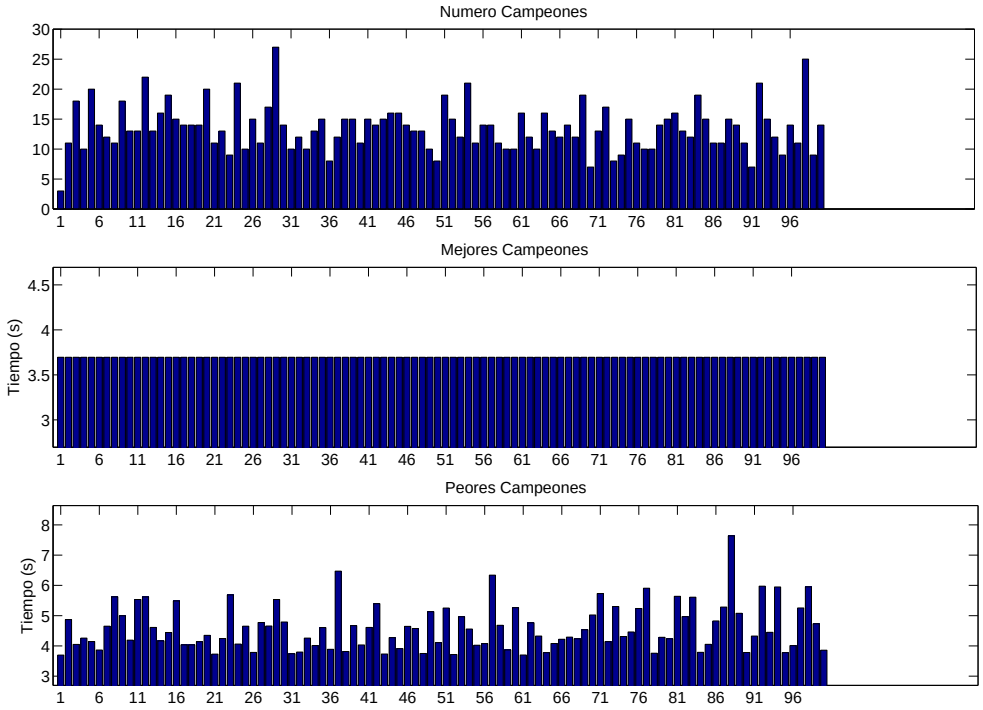


Figura 5.27: Comparación de los resultados obtenidos para un lote experimentos, con el objetivo definido como: $(x = 20, y = 0, \theta = 0^o, velocidad = --)$

Cuadro 5.7: Estadísticas del lote experimentos, para el objetivo definido como: $(x = 20, y = 0, \theta = 0^o, velocidad = --)$

	Nº de Soluciones	Media	Moda	Desviación Estándar	Mejor Solución
Mejores Soluciones	100	3.6957	3.6957	0	3.6957
Peores Soluciones	100	4.6012	3.6968	0.75361	3.6968

Posición(20,0), Orientación = 90º y Velocidad = –

La figura 5.28 muestra las 20 mejores trayectorias obtenidas. La figura 5.29 muestra las consignas de rumbos responsables de la generación de las trayectorias anteriores. La figura 5.30 muestra la velocidad del barco para cada uno de las 20 trayectorias obtenidas, según el instante de decisión concreto. Este es escenario que presente una maniobra compleja, además sabemos que presenta un mínimo local (ver sección 5.1). Hay que señalar que ninguna de la soluciones devueltas por el sistema, para aquellos casos donde no ha fallado, pertenece a la zona de los mínimos locales.

La figura 5.31 muestra comparaciones del lote de experimentos, cada columna corresponde a un experimento único. La primera gráfica muestra el número de soluciones obtenidas, la segunda el tiem-

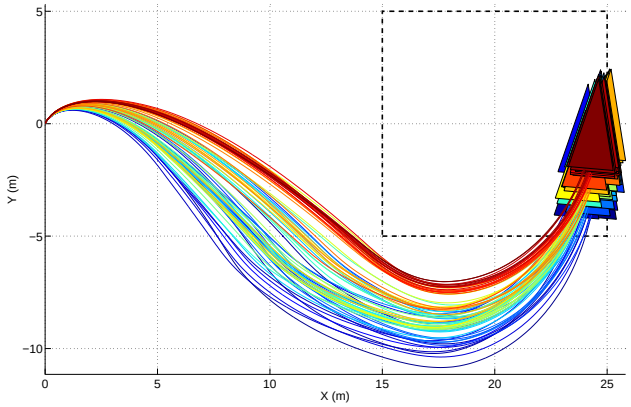


Figura 5.28: 20 mejores trayectorias obtenidas para un lote experimentos, con el objetivo definido como: $(x = 20, y = 0, \theta = 90^o, velocidad = --)$

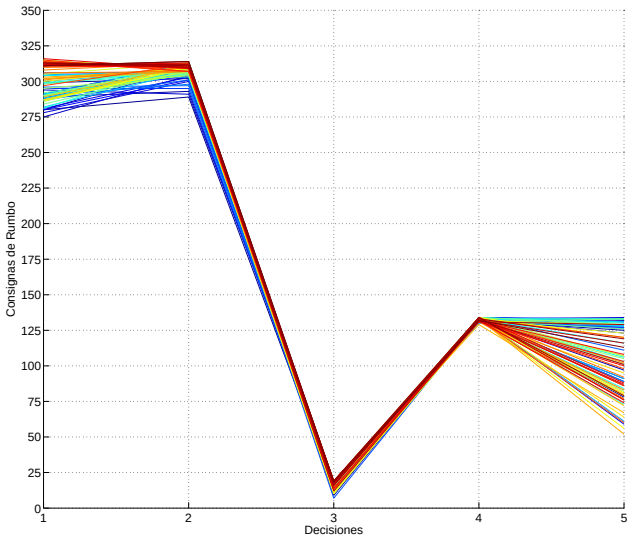


Figura 5.29: Consignas de rumbo obtenidas correspondientes a diferentes trayectorias, con el objetivo definido como: $(x = 20, y = 0, \theta = 90^o, velocidad = --)$

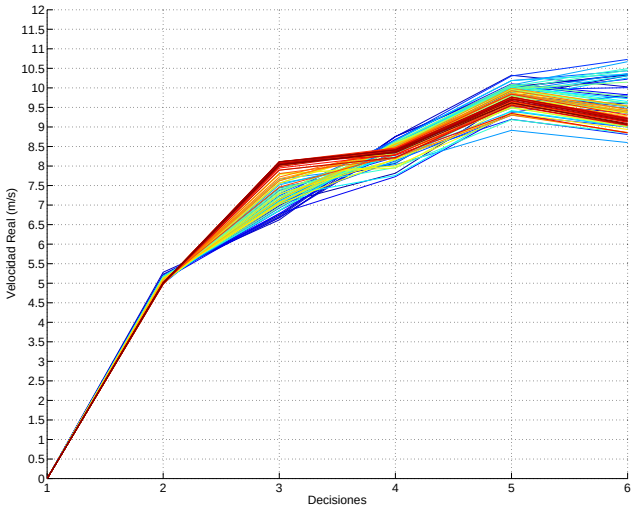


Figura 5.30: Velocidad del barco correspondiente a diferentes trayectorias, con el objetivo definido como: $(x = 20, y = 0, \theta = 90^o, velocidad = --)$

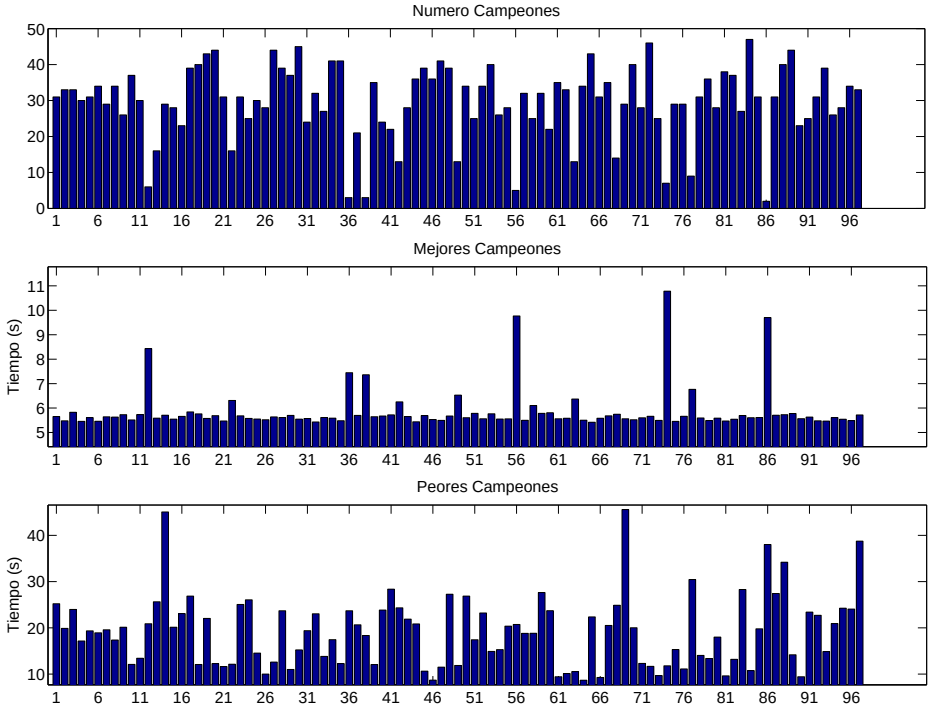


Figura 5.31: Comparación de los resultados obtenidos para un lote experimentos, con el objetivo definido como: $(x = 20, y = 0, \theta = 90^o, velocidad = --)$

po(s) de la mejor solución obtenida y la tercera el tiempo(s) de la peor solución obtenida. Nuevamente se pueden observar fallos en algunos de los experimentos aislados y la convergencia del sistema hacia trayectorias equivalentes en tiempo.

Cuadro 5.8: Estadísticas del lote experimentos, para el objetivo definido como: ($x = 20, y = 0, \theta = 90^o, velocidad = --$)

	Nº de Soluciones	Media	Moda	Desviación Estándar	Mejor Solución
Mejores Soluciones	100	5.8593	5.4204	0.87758	5.4204
Peores Soluciones	100	19.1536	8.6826	7.6527	8.6826

El cuadro 5.8 muestra las estadísticas correspondientes al lote de experimentos. Es interesante observar como en este escenario, al ser más complejo, se lleva a cabo una mayor optimización de las trayectorias. Los valores relativos a las mejores soluciones son,entre sí, bastante aproximados en relación a la dispersión presente en las primeras soluciones.

Posición(50,50), Orientación = 225º y Velocidad = –

La figura 5.23 muestra las 20 mejores trayectorias obtenidas.

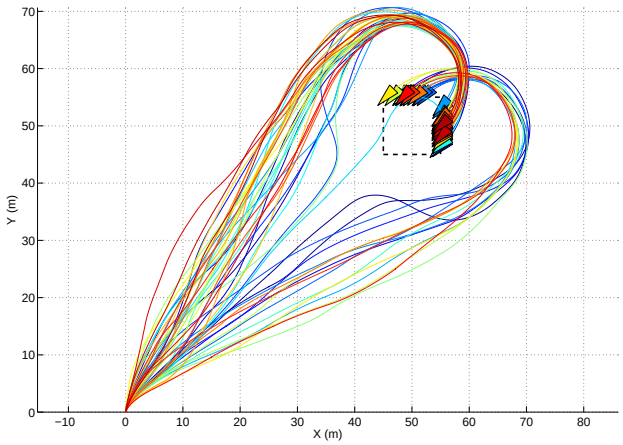


Figura 5.32: 20 mejores trayectorias obtenidas para un lote experimentos, con el objetivo definido como: ($x = 50, y = 50, \theta = 225^o, velocidad = --$)

La figura 5.31 muestra comparaciones del lote de experimentos, cada columna corresponde a un experimento único. La primera gráfica muestra el número de soluciones obtenidas, la segunda el tiempo(s) de la mejor solución obtenida y la tercera el tiempo(s) de la peor solución obtenida. Nuevamente se pueden observar fallos en algunos de los experimentos singulares y la convergencia del sistema hacia trayectorias equivalentes en tiempo.

El cuadro 5.9 muestra las estadísticas correspondientes al lote de experimentos. Es posible alcanzar el objetivo mediante trayectorias con tiempos muy similares, pero con trazados distintos. Una

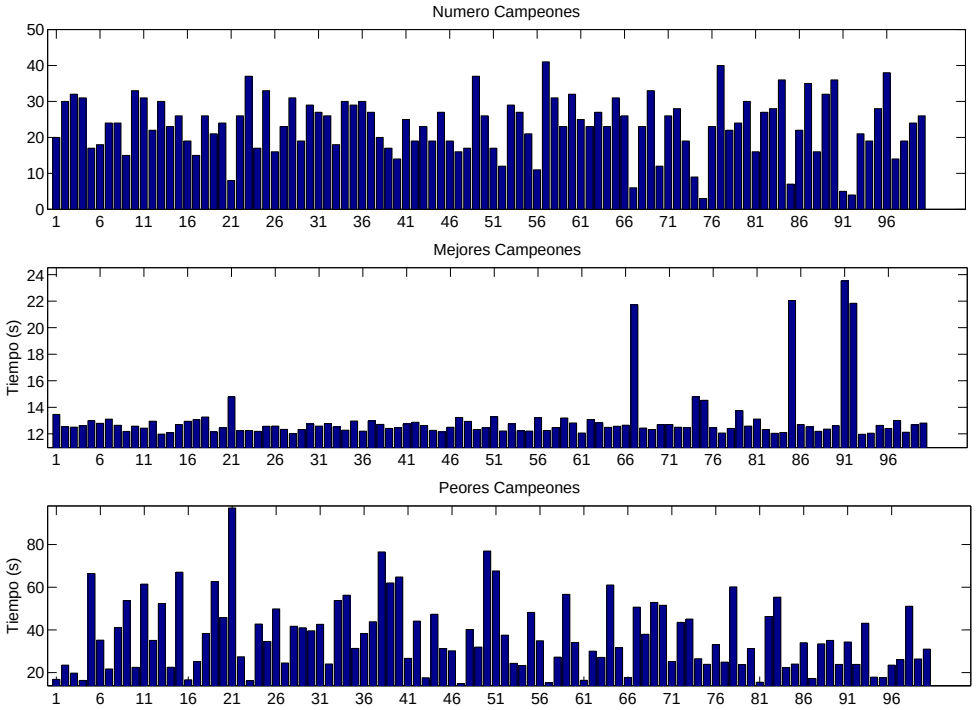


Figura 5.33: Comparación de los resultados obtenidos para un lote experimentos, con el objetivo definido como: ($x = 50, y = 50, \theta = 225^o, velocidad = --$)

Cuadro 5.9: Estadísticas del lote experimentos, para el objetivo definido como: ($x = 50, y = 5, \theta = 225^o, velocidad = --$)

	Nº de Soluciones	Media	Moda	Desviación Estándar	Mejor Solución
Mejores Soluciones	100	13.0204	11.9641	1.9721	11.9641
Peores Soluciones	100	36.9778	14.83	16.5025	14.83

posibilidad es aproximar el objetivo por la izquierda y por la derecha.

5.4. Discusión

Un primer aspecto a destacar es la capacidad del sistema para encontrar soluciones óptimas simétricas en los escenarios donde existan, esta capacidad indica la ausencia de sesgo en las elecciones del sistema. En principio cualquiera de las dos soluciones simétricas pueden ser descubiertas, que se opte por una u otra es una cuestión de azar. A medida que se descubran más soluciones en una región, más hormigas se destinarán a la búsqueda de soluciones en dicha región, de tal modo que en los casos simétricos existe un punto de inflexión donde la búsqueda se decanta por una región en vez de la otra. Una vez que se ha encontrado la primera solución, con la comida disponible al inicio se generan un conjunto de hormigas exploradoras. Las hormigas exploradoras tienen más probabilidades de explorar la solución encontrada. Como consecuencia la solución optima encontrada predomina y para que cambiase a la solución simétrica, tendría que darse el caso de que esta fuese encontrada directamente por una exploradora. Lo que es improbable.

Un segundo aspecto que enlaza con el anterior es la capacidad del sistema para evitar los mínimos locales. En ninguno de los experimentos que han completado el número de iteraciones, la solución final estaba contenida en una zona de mínimos locales. Hay que señalar que llamamos zonas de mínimos locales a aquellas donde existiendo solución, estas no aproximan la solución óptima. Esto se consigue gracias a la capacidad del sistema para explorar múltiples zonas simultáneamente.

De lo dicho anteriormente se puede concluir que efectivamente el sistema parece aproximar las trayectorias óptimas. Como base para esta afirmación hay que señalar la comparación con la solución analítica, los casos donde el lote de experimentos ha encontrado una solución única o soluciones simétricas y la ausencia de soluciones en mínimos locales.

también a raíz de estos experimentos quedan patentes dos aspectos mejorables. El primero es que el sistema no ha sido ajustado para cada escenario concreto, todos los experimentos se han realizado con la misma configuración básica. En algunos escenarios parece ser escasa en número de iteraciones y tamaño de la población, y en otros probablemente estén sobre dimensionados.

El segundo aspecto tiene que ver con las situaciones donde el experimento falla, porque la población se extingue y no completa las iteraciones pedidas. Si bien esta situación es fácilmente resoluble, basta con reiniciar el alimento disponible cuando la población se ha extinguido, esto no nos parece muy adecuado. La razón es doble, por un lado habría que conseguir que fuese más robusto (esta es una de las líneas de desarrollo futuro), y por otro que las extinciones advierten de la complejidad del problema, o dicho de otro modo: qué tipo de escenarios son los más costosos de resolver.

Conclusiones

- Se ha desarrollado un planificador de maniobras en mar abierto para barcos autónomos que incluye la dinámica de estos. La planificación se realiza en lazo abierto, previamente a la ejecución de la maniobra.
- El desarrollo se basa en el empleo de algoritmos bioinspirados, desarrollados a partir del Ant Colony Optimization. La extensión permite aplicar este método de solución discreto a la planificación de movimiento.
- El algoritmo desarrollado permite obtener un juego de valores de consigna para los actuadores del barco cuyo empleo permite a este realizar maniobras complejas en mar abierto. Además permite encontrar maniobras viables en un periodo de tiempo corto y con suficiente tiempo, mejorarlas para obtener maniobras próximas al óptimo. El criterio de optimización seguido es el empleo de un tiempo minimo para el recorrido de la trayectoria.
- Se ha probado exhaustivamente el sistema desarrollado para distintos escenarios, variando la complejidad de la maniobra. En todos lo casos se ha conseguido la convergencia del método, así como soluciones próximas al óptimo.
- La carga computacional del método es alta, dependiendo del escenario. Los casos donde la distancia al objetivo excede los 100m el tiempo de computo excede las 5 horas, utilizando la versión R2007a de Matlab en un computador PowerPC G5 (Single Core) con 2GB de Memoria RAM y sistema operativo Mac OS X (10.5.7). Teniendo en cuenta que el calculo no se ha paralelizado.
- Como líneas de desarrollo futuras se incluye: la reducción del tiempo computacional mediante técnicas de paralelización. La inclusión de obstáculos (costas, aguas poco profundas, etc..) en el escenario de maniobra. Así como el empleo del algoritmo desarrollado en otros escenarios de planificación de movimiento.
- Parte del trabajo realizado ha sido aceptado como una comunicación en 8th IFAC Conference on Manoeuvring and Control of Marine Craft / MCMC09 (2009).

Bibliografía

[1] J. Holland, *Adaptation in natural and artificial systems*. MIT Press Cambridge, MA, USA, 1992.

[2] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[3] S. M. La Valle, *Planning algorithms*. Cambridge University Press, 2006.

[4] M. Dorigo and T. Stützle, *Ant colony optimization*. MIT press, 2004.

[5] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence: from natural to artificial systems*. Oxford University Press, USA, 1999.

[6] S. Goss, S. Aron, J. Deneubourg, and J. Pasteels, "Self-organized shortcuts in the argentine ant," *Naturwissenschaften*, vol. 76, no. 12, pp. 579–581, 1989.

[7] G. Pólya, *How to solve it: A new aspect of mathematical method*. Princeton University Press Princeton, NJ, 1971.

[8] T. I. Fossen, *Marine Control systems*. MARINE CYBERNETICS AS, Trondheim, Norway, 2002.

[9] J. R. Dormand and P. J. Prince, "A family of embedded runge-kutta formulae," *J. Comp. Appl. Math.*, vol. 6, pp. 579–581, 1980.

[10] S. Wolfram, *A New Kind of Science*. Wolfram Media, May 2002.

[11] A. Ilachinski, *Cellular automata: A discrete universe*. World Scientific, 2001.

[12] M. Gardner, "Mathematical games: The fantastic combinations of John Conway's new solitaire game 'Life'," *Scientific American*, vol. 223, no. 4, pp. 120–123, 1970.

[13] B. Ruiz, F. Gutiérrez, P. Guerrero, and J. Gallardo, *Razonando con Haskell (Un curso sobre programación funcional)*. 2004.

[14] D. Gordon, *Ants at Work: How an Insect Society Is Organized*. WW Norton & Company, 2000.

[15] R. A. Brooks, "Elephants don't play chess," *Robotics and Autonomous Systems*, vol. 6, pp. 3–15, June 1990.

[16] R. Brooks, "Intelligence without reason," *Artificial intelligence: critical concepts*, vol. 3, 1991.

[17] R. Brooks, "Intelligence without representation," *Cambridge, MA: MIT Press*, 1997.

[18] S. Camazine, N. Franks, J. Sneyd, E. Bonabeau, J. Deneubourg, and G. Theraula, *Self-Organization in Biological Systems*. Princeton University Press Princeton, NJ, USA, 2001.

[19] T. Seeley, S. Camazine, and J. Sneyd, "Collective decision-making in honey bees: how colonies choose among nectar sources," *Behavioral Ecology and Sociobiology*, vol. 28, no. 4, pp. 277–290, 1991.

[20] A. Land and A. Doig, "An automatic method of solving discrete programming problems," *Econometrica: Journal of the Econometric Society*, pp. 497–520, 1960.

[21] N. Oliet, Y. Mallén, and J. López, *Estructuras de datos y métodos algorítmicos: Ejercicios resueltos*. Pearson Educación, 2004.