GENERACIÓN DE COMPORTAMIENTOS DE ENJAMBRE EN ROBOTS MÓVILES A TRAVÉS DEL USO DEL APRENDIZAJE POR REFUERZO.

WILSON ORLANDO QUESADA MONCAYO

UNIVERSIDAD SANTO TOMÁS
FACULTAD DE INGENIERÍA ELECTRÓNICA
BOGOTA D.C.
2018

GENERACIÓN DE COMPORTAMIENTOS DE ENJAMBRE EN ROBOTS MÓVILES A TRAVÉS DEL USO DEL APRENDIZAJE POR REFUERZO.

WILSON ORLANDO QUESADA MONCAYO

PROPUESTA DE TRABAJO DE GRADO PARA OPTAR AL TÍTULO DE INGENIERO ELECTRÓNICO

DIRECTOR:

Ing. JUAN MANUEL CALDERÓN CHÁVEZ Ph.D.

UNIVERSIDAD SANTO TOMÁS
FACULTAD DE INGENIERÍA ELECTRÓNICA
BOGOTA D.C.
2018

DEDICATORIA Y AGRADECIMIENTO

En primer lugar quiero agradecer a mi Dios todo poderoso porque me ha brindado la sabiduría y me ha permitido realizar y culminar mis estudios de pregrado. Dedico este trabajo a mi Padre Celestial y lo consagro para el reino de Dios. Quiero agradecer a mis padres, Josué Quesada, María Moncayo y a mi hermana Vivian Quesada, porque me han apoyado completamente y por creer en mí. A Diana Marcela Gallego por ser como una segunda madre para mí y por sus consejos, a Gonzalo García, Mariana García, Andrea Rojas y a todo el equipo del ministerio, que son mi segunda familia y mis consejeros, porque me han acompañado durante todo el proceso, me han enseñado a ser humilde y por la paciencia. Finalmente quiero agradecer al Ing. Juan Manuel Calderón Chávez por sus enseñanzas, paciencia, consejos y por ayudarme en la realización de este proyecto.

RESUMEN

En este trabajo se hace uso de técnicas de aprendizaje por refuerzo (Q-Learning) con el objetivo de entrenar un grupo de robots para generar comportamientos de enjambre.

Se presentan dos posibles soluciones con diferentes enfoques. En la primera solución propuesta se establecen los estados del robot en función de la distancia de sus dos vecinos más cercanos. En la segunda solución propuesta se definen un radio de atracción y otro radio de repulsión, y los estados se establecen según la cantidad de vecinos dentro de cada uno de los radios divididos en los cuatro cuadrantes locales del robot. Para cada solución propuesta se definen las acciones del robot y se propone una política de premios y castigos. Cada robot se conecta con sus vecinos una vez que ha alcanzado una distancia prudente. Se hace uso de teoría de grafos para medir la conectividad del enjambre y saber si la topología del grafo que forma el enjambre al final de la simulación es conexo o no. En este trabajo se asume que la comunicación de cada agente con sus vecinos ya está resuelta. Se realizan varias pruebas en Matlab para cada una de las soluciones propuestas variando el número de robots del enjambre. Finalmente se prueba la segunda solución propuesta en V-rep usando robots cuadricópteros virtuales.

Este documento está estructurado de la siguiente forma: En el capítulo 1 y 2 se define el problema y la justificación. El capítulo 3 y 4 contienen una revisión de trabajos relacionados con robótica de enjambre y se definen los objetivos del proyecto. En el capítulo 5 se presentan los conceptos teóricos necesarios utilizados en el desarrollo de este proyecto. En el capítulo 6 y 7 se muestra el diseño metodológico, la administración del proyecto, cronograma de actividades y presupuesto para el proyecto. En el capítulo 8 se muestra el trabajo previo a la realización de este proyecto usando lógica difusa. En el capítulo 9 y 10 se muestra el diseño del proyecto, el planteamiento de las soluciones propuestas, - las pruebas y resultados de las dos soluciones. Finalmente, las conclusiones se muestran en el capítulo 11.

TABLA DE CONTENIDO

INTRODUCCIÓN

A través del tiempo se han generado numerosos desastres naturales, como terremotos, inundaciones, ciclones, entre otros, en diferentes partes del mundo que han provocado numerosos daños y muertes. Entre 1994 y 2013 se han registrado 6873 catástrofes en todo el mundo y casi 1,35 millones de muertos a causa de los desastres naturales [1].

Después de los desastres muchos de los sobrevivientes quedan heridos y/o atrapados bajo escombros de derrumbes, en áreas que siguen siendo peligrosas incluso para los rescatistas debido a la inestabilidad del terreno, gases tóxicos, daños en el sistema eléctrico, entre otros. Por lo que se torna muy difícil localizar a las víctimas y dar apoyo, ya que el equipo de rescate debe tener cuidado de guardar su propia vida y procurar su propia seguridad.

Este problema retrasa el proceso de búsqueda, aumenta las muertes y lesiones en las victimas, por lo que se han desarrollado diferentes métodos que actualmente se involucran en el proceso de búsqueda en desastres:

- Búsqueda física, en la que se utiliza evaluaciones visuales y vocales para ubicar a las víctimas. Este método puede ignorar victimas inconscientes, se ve afectado por condiciones del lugar (visibilidad, temperatura, hora del día) y se restringe a áreas relativamente seguras para el equipo de rescate [2].
- Búsqueda canina, en la que perros entrenados tratan de localizar a las víctimas atrapadas por medio del olfato; también pueden detectar victimas inconscientes. Este método se ve afectado por la disponibilidad y capacidad del equipo canino ya que puede que los perros no estén disponibles, y aunque estén disponibles necesitan descansar periódicamente, además de la accesibilidad del sitio y el tiempo que puede tomar transportar los perros al lugar del desastre [2].
- Búsqueda electrónica, es el método en el que se hace uso de dispositivos como cámaras, micrófonos y equipos sísmicos, que permiten cubrir grandes áreas y ayudan al equipo de rescate a encontrar víctimas atrapadas. Este método también tiene limitaciones al momento de encontrar victimas inconscientes y en el despliegue [2].

Si bien estos métodos creados pueden aportar en mejorar el proceso de búsqueda y rescate de víctimas, es importante contar con nuevos enfoques más eficaces, seguros y rápidos que permitan dar apoyo a las victimas sin exponer la seguridad ni la vida del equipo de rescate. Por ejemplo, el uso de robots en zonas de desastres para apoyar al equipo de rescatistas puede optimizar tareas tales: como exploración, mapeo, reconocimiento del terreno, ubicación de víctimas, reducir el tiempo de búsqueda, entre otros. El uso de la robótica cooperativa es una excelente alternativa para la colaboración en tareas de búsqueda y rescate ya que éste enfoque aporta grandes ventajas, como que los robots se desplieguen abarcando grandes áreas, la búsqueda puede continuar aun si un robot llega a fallar o queda atascado, los robots pueden formar una red de comunicación para la organización de equipos de rescate y telemedicina, entre otros.

Es importante que los robots además de ser autónomos tengan la capacidad de adaptarse y que posean habilidades de navegación colectiva para la exploración, por lo cual el presente trabajo pretende desarrollar habilidades de navegación para que los robots aprendan a comportarse y evolucionar como un enjambre.

Este trabajo se desarrolló gracias a la colaboración y el apoyo del Grupo de Investigación y Desarrollo en Robótica de la Universidad Santo Tomas (G.E.D.). El grupo de investigación trabaja principalmente tres líneas de investigación denominadas, Sistemas Inteligentes, Visión Artificial y Robótica. Dentro de estas líneas de investigación el grupo ha desarrollado diferentes proyectos de investigación enfocados a robótica cooperativa, robótica educativa, inteligencia artificial y robots humanoides. El grupo ha participado activamente en la iniciativa RoboCup en donde se desarrolló un equipo de futbol de robots para la liga “Small Size” tal como se muestra en los trabajos [3, 4, 5, 6]. Se desarrollaron estrategias de control y lógica difusa para la reducción de la fuerza de impacto en robots humanoides cuando estos caen o saltan [7, 8, 9, 10,11]. Además han trabajado en el uso de la robótica para incentivar a estudiantes de colegio a optar por carreras afines a la matemática y la ingeniería [12, 13]. Actualmente se desarrollan proyectos de investigación enfocados en la robótica de enjambre y sistemas de aprendizaje automático tal como se demuestra en el desarrollo del presente trabajo.

1. PLANTEAMIENTO DEL PROBLEMA

1.1. PREGUNTA PROBLEMA

El presente trabajo planea dar solución al siguiente problema:

¿Cómo entrenar múltiples robots móviles para que aprendan a comportarse como un enjambre y navegar en entornos desconocidos?

1.2. PLANTEAMIENTO

En el campo de la robótica, el enfoque hacia la coordinación de un gran número de robots para generar comportamientos colectivos inspirados en el estudio de animales sociales, es conocido como robótica de enjambre.

Es importante desarrollar algoritmos que permitan a los robots no solamente ser autónomos, sino tener también la capacidad de aprender y adaptarse, como por ejemplo los algoritmos de aprendizaje por refuerzo o RL por sus siglas en ingles.

Sin embargo, el aprendizaje de refuerzo (RL) con múltiples robots no ha sido ampliamente estudiado y existen muy pocos trabajos con alcance limitado en el campo, ya que abordar un problema de robótica de enjambre con aprendizaje de refuerzo es un tema desafiante. El problema principal es como descomponer la recompensa global de todo el enjambre en recompensas individuales para cada robot. Este problema es conocido como asignación de crédito espacial. En [14] Mataric afronta este problema con solo 4 robots.

El uso de múltiples robots coordinados tiene aplicaciones en un gran número de tareas que aportan a solucionar problemáticas de la humanidad como en la agricultura, en construcciones, en el sector industrial, entre otras. El uso de la robótica colectiva también es principalmente útil en tareas de búsqueda y rescate en entornos caóticos o terrenos inestables como derrumbes, también puede ayudar a completar la búsqueda en menor tiempo sin arriesgar la seguridad del equipo de rescatistas.

El presente trabajo propone la aplicación de algoritmos de aprendizaje por refuerzo (Q-learning) en robótica de enjambre para que los robots aprendan a comportarse y evolucionar como enjambre en ambientes desconocidos.

2. JUSTIFICACIÓN

Durante más de cincuenta años los robots han sido usados para realizar tareas que generalmente resultan ser sucias, aburridas y/o peligrosas [15]. La aplicación de la robótica juega un papel de gran importancia en diferentes sectores como la industria, automotriz, medicina o en aplicaciones en las que es necesario acceder a lugares peligrosos para los humanos y de difícil acceso [16].

El uso de múltiples robots independientes trabajando de forma cooperativa para conseguir un objetivo común tiene claras ventajas, a diferencia de usar un único robot sofisticado y costoso [15]:

- La misión puede continuar, aun cuando un agente del enjambre llegue a fallar.
- El enjambre puede cubrir grandes áreas distribuyendo los agentes en el espacio.
- Incluso se puede generar sistemas de robots que sean flexibles y adaptables.

La robótica colectiva tiene diferentes aplicaciones como por ejemplo en la gestión de fabricación y almacén, para cobertura de red y monitoreo de desastres, robots de construcción, robótica agrícola, entre otros [15], siendo una aplicación importante la búsqueda y rescate en áreas afectadas por catástrofes naturales. La robótica de enjambre surge como una alternativa prometedora en el proceso de búsqueda de víctimas en zonas de desastre para salvar vidas.

Una de las características importantes y deseables en un robot es, además de ser autónomo, que sea capaz de aprender y adaptarse. Las técnicas de aprendizaje automático, como el aprendizaje por refuerzo, son especialmente útiles cuando es difícil planificar las acciones correctas [16]. Por ejemplo en entornos caóticos y desconocidos o cuando las circunstancias del terreno pueden cambiar, como terrenos en áreas de desastres, es muy importante que el sistema aparte de ser autónomo tenga capacidad de aprendizaje para que se pueda adaptar.

Por otra parte, el uso de la robótica de enjambre en tareas de búsqueda y rescate puede aportar grandes ventajas como por ejemplo aumentar significativamente la velocidad y efectividad de la búsqueda, los robots pueden alcanzar lugares estrechos y de difícil acceso para el equipo de rescatistas, pueden realizar mapeo del lugar y ayudar al equipo de recate en el proceso de la búsqueda o transportar suministros [17]. Con este proyecto de grado se busca producir un comportamiento colectivo partiendo del comportamiento individual de cada agente del enjambre usando algoritmos de aprendizaje por refuerzo (Q-learning).

3. ANTECEDENTES.

En [18] Brambilla presenta una revisión de la ingeniería de enjambre. Para estudiar los trabajos relacionados con robótica de enjambre, se proponen dos taxonomías principales (ver Fig. 1) que son: Según el método, en el que se clasifican los trabajos de acuerdo con el método utilizado para diseñar o analizar el enjambre de robots, y según el comportamiento colectivo, en el que se clasifican los trabajos de acuerdo al comportamiento básico del enjambre.

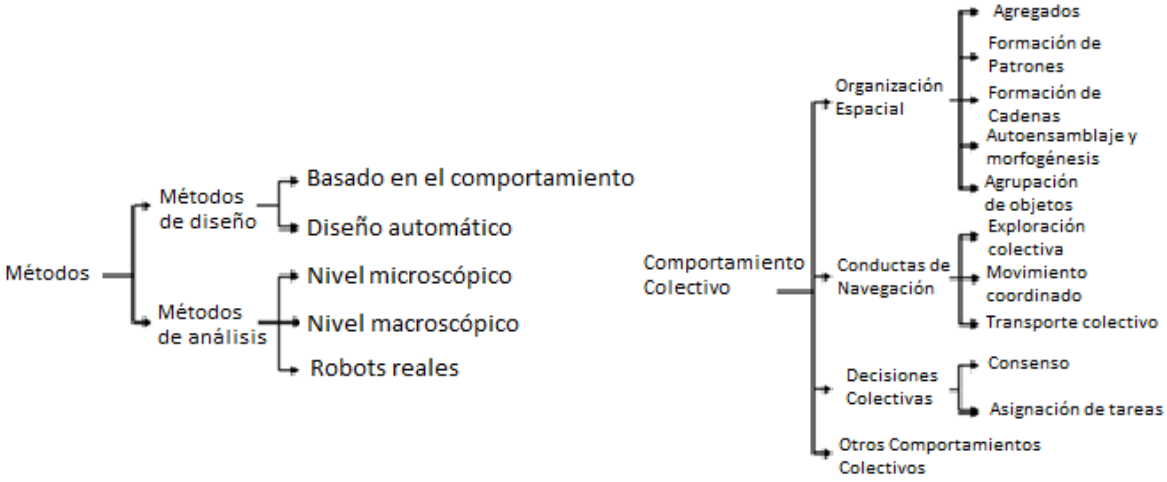


Figura 1. Diagrama de taxonomías propuestas por Brambilla.

Brambilla propone dos diferentes métodos de diseño para robótica de enjambre: diseño basado en el comportamiento y diseño automático. En el primer método, se ajusta el comportamiento individualmente para cada robot de forma iterativa hasta obtener un comportamiento colectivo deseado. En el diseño automático, Brambilla lo clasifica en dos categorías: robótica evolutiva y aprendizaje de refuerzo de robots múltiples, siendo este último el objeto de estudio de este trabajo.

Considerando el enfoque de este proyecto de grado, se buscaron proyectos que estudiaron temas similares a los abordados en este trabajo. José León León en su trabajo: **simulación de enjambres de robots en labores de exploración para detección de posibles víctimas** [17, 19, 20] el cual fue presentado en 2017 como Tesis de Maestría de la Facultad de Ingeniería Electrónica de la Universidad Santo Tomás de Colombia, propone y simula un algoritmo para la navegación y el consenso de múltiples agentes enfocado a localizar víctimas en desastres. El autor propone para la navegación un algoritmo bio-inspirado en las abejas basado en fuerzas virtuales de atracción y repulsión, y para el consenso un protocolo distribuido para tomar una decisión cuando se disponga de la evidencia suficiente.

Por otro lado en Yanguas et al. [21] Se realiza un algoritmo simulado con la capacidad de asignar de manera optimizada tareas de búsqueda y rescate de víctimas en zona de desastre, haciendo uso de un grupo de robots heterogéneos (con diferente capacidades de velocidades para navegaren el entorno). En el trabajo se plantean 4 tareas esenciales la exploración del entorno, la navegación de robots en un espacio no convexo (un entorno con obstáculos que no permite hacer uso de distancias euclidianas), detección de víctimas y clasificación de víctimas entre victima con capacidad

de movilizar o aquellas con requerimiento de suplementos médicos debido a la imposibilidad de moverse.

En julio de 2004 Youcef Zennir presentó su trabajo, **Apprentissage Par Renforcement Et Systemes Distribués : Application À L'apprentissage De La Marche D'un Robot Hexapode** [16], en el Instituto Nacional De Ciencias Aplicadas De Lyon como Tesis Doctoral, en la cual plantea el uso de técnicas de aprendizaje de refuerzo para que un robot hexápodo aprenda a caminar. El autor adopta un enfoque de Q-learning distribuido considerando cada extremidad del robot como agentes independientes que realizan el proceso de aprendizaje teniendo en cuenta o no a los otros agentes. Finalmente el robot simulado logra caminar como un fenómeno emergente de los movimientos individuales de las extremidades; el robot aprende a recorrer una trayectoria mientras mantiene su postura.

En julio del 2013 fue publicado en la conferencia de control: **Proceedings of the 32nd Chinese Control Conference,** en Xi'an, China, el trabajo **The Improved Q-Learning Algorithm based on Pheromone Mechanism for Swarm Robot System [22]**, en el cual se propone un algoritmo mejorado de Q-Learning utilizando mecanismos de feromonas de colonias de hormigas para planificación de rutas en robótica de enjambre. Los autores simulan el algoritmo en la plataforma Player/Stage y lograron demostrar que el algoritmo propuesto tiene una alta eficiencia comparado con otros algoritmos como el Q-Learning y el PSO (Particle Swarm Optimization).

En septiembre del 2017 Adrian Šošić, Abdelhak M. Zoubir y Heinz Koeppl publicaron en la revista Springer Science el trabajo **Reinforcement learning in a continuum of agents** [23]**,** en el cual los autores proponen un modelo de comportamiento colectivo basado en una descripción continua de un grupo de agentes, partiendo del supuesto de que cada agente tiene acceso únicamente a información local y está parcialmente informado sobre el estado del sistema global, los agentes construyen estructuras de control a partir de la retroalimentación de recompensa local o global para abrir el camino al control distribuido autoaprendido.

En el trabajo de Wilson Quesada: **Leader-follower formation for UAV robot swarm based on fuzzy logic theory** [25], el cual fue presentado en la conferencia Internacional sobre Inteligencia Artificial y Soft Computing ICAISC 2018, se propone un algoritmo basado en la lógica difusa para guiar un enjambre de robots siguiendo un robot líder mientras se mantiene la formación. El sistema propuesto se encarga de calcular la velocidad y dirección de movimiento para cada robot del enjambre. El algoritmo fue programado y evaluado en Matlab, para posteriormente implementarlo y simularlo en V-Rep. En la simulación del algoritmo se generaron los siguientes comportamientos: Primero cuando el líder está quieto, los otros robots se reúnen alrededor de él dejándolo en el centro. El otro comportamiento generado es cuando el líder está en movimiento, los otros robots lo siguen a la nueva ubicación y lo rodean.

Los trabajos consultados previamente abordan problemas relacionados con la robótica de enjambre en diferentes escenarios y usando diferentes enfoques y estrategias para la programación del algoritmo. Este trabajo propone el uso de algoritmos de aprendizaje de refuerzo como el Q-Learning para enseñar a múltiples robots a comportarse colectivamente como un enjambre.

4. OBJETIVOS

4.1. OBJETIVO GENERAL.

El objetivo de la presente propuesta es:

- Generar un comportamiento de enjambre en múltiples robots móviles usando algoritmos de aprendizaje por refuerzo (Q-learning).

4.2. OBJETIVOS ESPECÍFICOS

- Definir el conjunto posible de estados, acciones y la política de exploración para seleccionar la acción adecuada según el grado de aprendizaje de cada uno de los robots que conforman el enjambre.
- Establecer la política de premios y castigos que direccionen el aprendizaje de los agentes hacia un comportamiento de enjambre.
- Simular el algoritmo de aprendizaje para un robot de forma individual, en un enjambre robótico utilizando software como Matlab o V–REP (Los programas pueden cambiar si se considera necesario).
- Evaluar los resultados obtenidos de la interacción de los robots del enjambre entre ellos y el entorno.

5. MARCO TEÓRICO

Para la realización y el desarrollo de este proyecto es necesario conocer varios términos y conceptos que ayudaran a un mejor entendimiento e interpretación del mismo.

A continuación, se introducirán dichos conceptos y términos necesarios, que son: Ciencias de la computación, aprendizaje automático, aprendizaje por refuerzo, Q-Learning y Robótica de enjambre.

5.1. CIENCIAS DE LA COMPUTACIÓN

Es la ciencia que se enfoca en el estudio del procesamiento y transformación de la información, y su interacción con el mundo. Peter Denning, la define de la siguiente manera [25]:

> "El cuerpo de conocimiento de las ciencias de la computación se describe con frecuencia como el estudio sistemático de los procesos algorítmicos que describen y transforman la información: su teoría, análisis, diseño, eficiencia, implementación y aplicación. La pregunta fundamental que subyace a toda la computación es: ¿Qué puede ser (eficientemente) automatizado?". [25]

5.2. APRENDIZAJE AUTOMÁTICO

También conocido como machine learning o aprendizaje de máquinas, es una rama de las ciencias de la computación, o más específicamente de la inteligencia artificial, cuyo objetivo principal es dotar a las máquinas y equipos de cómputo con la capacidad de aprender. Lo que se busca es generar métodos generales que permitan predecir comportamientos futuros usando información de lo que ha ocurrido en el pasado. [26]

Los algoritmos de aprendizaje automático se pueden clasificar en diferentes subcategorías dependiendo de cómo se aborde el problema de aprendizaje. Las dos subcategorías principales del aprendizaje automático son el aprendizaje supervisado y el aprendizaje no supervisado, que se explican a continuación:

- **Aprendizaje supervisado:**
  Consiste en aprender de un conjunto de ejemplos bien conocidos, como datos de entrada etiquetados con las salidas correspondientes, que son proporcionados por un supervisor externo. El objetivo principal de este tipo de aprendizaje es que el sistema generalice la respuesta y pueda actuar correctamente ante datos de entrada desconocidos con los que no se entrenó. [27]

- **Aprendizaje no supervisado:**
  En este tipo de aprendizaje el sistema recibe como entradas un conjunto de datos de los que no se conocen su clasificación correcta. El sistema trata de encontrar estructuras o patrones ocultos existentes en los datos de entrada para etiquetarlos. [27]

A demás del aprendizaje supervisado y el aprendizaje no supervisado, existen otros tipos del aprendizaje automático. Un tercer paradigma, del que se hablara con mayor detalle más adelante, es el aprendizaje por refuerzo.

5.3. APRENDIZAJE POR REFUERZO

El aprendizaje por refuerzo o RL, por sus siglas en inglés (reinforcement learning), es otro enfoque importante de aprendizaje automático que se basa en la interacción del agente con el entorno. El agente aprende a comportarse a través del ensayo y error, de los resultados obtenidos de las acciones tomadas, es decir, aprende de la experiencia. Lo que se busca es asignar acciones a situaciones, con el fin de maximizar una señal de recompensa numérica. [27]

El agente de aprendizaje no sabe de antemano que acciones tomar, sino que debe probar cada acción y descubrir cuales provocan la mayor recompensa en diferentes estados posibles.

En el aprendizaje por refuerzo se pueden identificar algunos términos principales, además del agente y el entorno, como: política, señal de recompensa, función de valor, estados, acciones y modelo del entorno, las cuales se explicarán a continuación [27, 28]:

- Agente: Es el aprendiz, el que toma las decisiones y ejecuta acciones.
- Entorno o medio ambiente: Es todo lo que esta fuera del agente, el mundo en el cual puede operar y se mueve.
- Estado: Es la situación actual del agente en el entorno.
- Acciones: Es el conjunto de todos los posibles movimientos y decisiones que el agente puede tomar.
- Señal de recompensa: Es un valor numérico generado por el entorno que sirve de retroalimentación al agente y permite medir el éxito o fracaso de las acciones tomadas.
- Política: Es el método por el cual se asignan los estados percibidos del entorno a las acciones que se deben tomar en esos estados. Es lo que define el comportamiento del agente en determinado estado.
- Función de valor: Es la estimación de la recompensa total acumulada en el futuro al tomar una acción en un estado dado, y sirve para determinar qué es lo que más le conviene al agente en largo plazo.
- Modelo del entorno: Permite realizar predicciones sobre cómo se comportará el entorno ante un estado y una acción para estimar el próximo estado y la próxima recompensa. Algunos sistemas de aprendizaje de refuerzo usan modelos del entorno y otros no.

Para entender mejor el aprendizaje por refuerzo se ilustrará con el siguiente ejemplo:

Cuando un niño está aprendiendo a caminar se topa con varios factores a tener en cuenta como: primero intentar levantarse y permanecer quieto, equilibrar el peso corporal, decidir cuál pie debe mover y donde ponerlo. [29]

En este ejemplo el niño es el agente que intenta caminar en el entorno, que es el mundo que lo rodea (la superficie sobre la que camina), y su estado con las acciones que toma. Cada vez que el niño pierde el equilibrio y cae al piso intentando levantarse o dar un paso, recibe una recompensa negativa que es el sentimiento de dolor por el golpe y la frustración por no lograr levantarse o dar el paso. Cuando el niño logra completar con éxito sub módulos de la tarea principal, recibe una recompensa positiva, que puede ser la satisfacción que siente por tener éxito.

Este es un ejemplo simplificado del RL que permite ver como el agente aprende mientras interactúa con el entorno, y tiene objetivos claros y definidos.

Proceso de decisión de Markov

Para formalizar un problema de aprendizaje por refuerzo, lo más común es representarlo como un proceso de decisión de Markov (MDP).

En el ejemplo anterior se tenía un agente (el niño), en un entorno (el mundo que lo rodea). El agente interactúa continuamente con el entorno por medio de las acciones que toma (dar un paso, inclinar su cuerpo y mover sus brazos para equilibrarse, etc.). Estas acciones modifican el entorno, el cual responde al agente presentándole nuevos estados y recompensas.

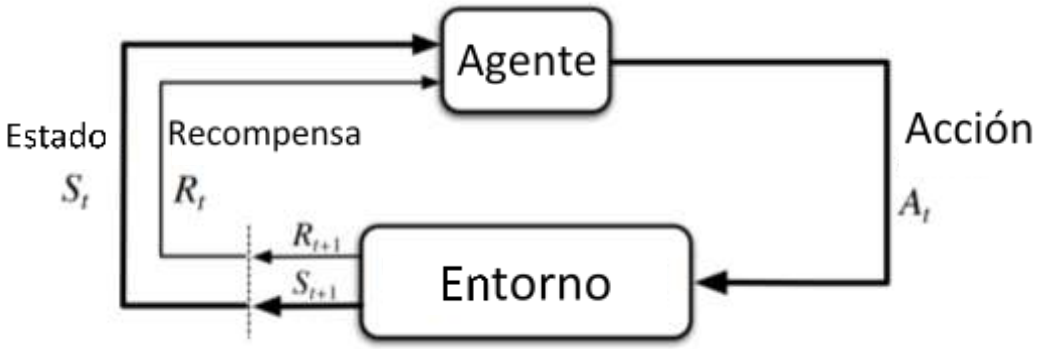


Figura 2: Interacción agente-entorno en un proceso de decisión de Markov [27].

El proceso de decisión de Markov o MDP lo conforma el conjunto de estados, acciones y la política para el cambio de un estado a otro. En secuencias de tiempo con pasos discretos, $t = 0,1,2,3 \ldots$ se presentan los estados S_t, sobre el cual el agente tomará una acción A_t, y en el siguiente paso recibirá una recompensa R_{t+1}, y un nuevo estado S_{t+1}. En un MDP finito se tiene secuencias finitas de estados, acciones y recompensas [27, 30]:

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3 \ldots S_{n-1}, A_{n-1}, R_n, S_n$$

El agente debe tener en cuenta no solo las recompensas inmediatas, sino también las futuras, si se desea tener un buen desempeño en el largo plazo.

Para calcular la recompensa total desde un punto de tiempo t se expresa como:

$$G_t = R_t + R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_n$$

Para las recompensas futuras se utiliza un factor de descuento (γ). Por lo que G_t queda como sigue:

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \gamma^3 R_{t+3} + \cdots + \gamma^{n-t} R_n$$

$$G_t = R_t + \gamma(R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots)$$

$$G_t = R_t + \gamma G_{t+1}$$

El factor de descuento es un valor entre 0 y 1 ($0 \leq \gamma \leq 1$). Entre más pequeño sea el valor de descuento menos se tendrán en cuenta las recompensas futuras. El agente tratará de seleccionar las acciones que maximicen las recompensas futuras descontadas. Si $\gamma = 0$, se dice que el agente es "miope", es decir, que solo se preocupa por maximizar la recompensa inmediata sin tener en cuenta las recompensas futuras [27, 30].

5.4. Q – LEARNING

El método del aprendizaje de refuerzo que se utilizará en este trabajo es el de Q-Learning, para el cual se define una función $Q(s, a)$ [30]:

$$Q(s_t, a_t) = max(R_{t+1})$$

La cual representa la recompensa futura máxima recibida cuando se toma la acción a en el estado s. El nombre de la función Q viene por "quality", representa la calidad de tomar una acción *a* estando en un estado s.

El objetivo es obtener el mejor puntaje posible al final. Aquí se hace uso de una política "optima" que permite conocer la acción que maximiza la recompensa para el estado s, y se define como sigue:

$$\pi^*(s) = \underset{a}{argmax} Q(s, a)$$

Donde π^* representa la política optima, con la que se elige la acción en cada estado.

La función $Q(s, a)$ se puede expresar recursivamente en términos del valor Q del siguiente estado s', como sigue:

$$Q(s, a) = r + \gamma \underset{a'}{max} Q(s', a')$$

La ecuación anterior es conocida como la ecuación de Bellman, que indica que la recompensa futura máxima es la recompensa recibida por el agente para el estado s más una fracción de la recompensa futura máxima del siguiente estado s'.

La idea de Q-Learning es aproximar iterativamente la función Q usando la ecuación de Bellman. La ecuación Q-Learning está dada como sigue:

$$Q_{t+1}(S_t, A_t) = Q_t(S_t, A_t) + \alpha \left[r_{t+1} + \gamma \underset{a}{max} Q_t(S_{t+1}, a) - Q_t(S_t, A_t) \right]$$

Donde α es la velocidad de aprendizaje, indica que tanto peso tiene la diferencia entre el valor Q anterior y el nuevo.

El algoritmo Q-Learning es el siguiente [27, 30]:

Inicializar $Q(S, A)$ arbitrariamente
Repetir por cada episodio:
    Inicializar **S**
    Repetir para cada paso del episodio:
        Elija A de S utilizando la política derivada de Q
        Tome la acción A, recompensa R, estado futuro S'
        $Q(S, A) \leftarrow Q(S, A) + \alpha \left[R + \gamma \underset{a}{max} Q(S', a) - Q(S, A) \right]$
        S ← S'

5.5. ROBÓTICA DE ENJAMBRE

Es un enfoque novedoso de la robótica que está inspirado en la observación de comportamientos auto-organizados de animales sociales (peces, hormigas, abejas, termitas, aves, entre otros) que busca coordinar un gran número de robots relativamente simples, de tal forma que puedan interactuar localmente entre ellos y el entorno para generar comportamientos colectivos inteligentes [18].

En general el funcionamiento de los enjambres naturales a nivel de sistema ha demostrado ser robusto, escalable y flexible. Estas tres propiedades son la principal motivación para los sistemas robóticos de enjambre y se explican con más detalle posteriormente [31]:

- **Robustez:**
  En el contexto de la robótica de enjambre, se refiere a la capacidad que tiene el sistema para continuar funcionando a pesar de pérdidas o fallas de algunos agentes, o perturbaciones en el ambiente. Esta característica de robustez es posible gracias a la redundancia de agentes, lo que permite que si un agente falla otro pueda reemplazarlo en la misma tarea, y la ausencia de un líder, ya que si el líder falla implicaría un fallo para todo el sistema de enjambre a menos que se pueda asignar a otro líder de forma automática.
- **Escalabilidad:**
  Se refiere a la capacidad que tiene el sistema robótico de enjambre para mantener el correcto funcionamiento con diferentes tamaños de grupo. Es decir, si se introducen nuevos agentes en el enjambre o si se eliminan, el rendimiento general del grupo no debería de verse muy afectado.
- **Flexibilidad:**
  Es la capacidad que tiene el sistema robótico de enjambre para realizar un gran número tareas de diferente naturaleza y operar en diferentes entornos.

Existen algunas características de diseño principales que se tiene en cuenta para determinar si un trabajo pertenece al campo de la robótica de enjambre o a otros enfoques de múltiples robots. A continuación, se muestra la lista propuesta por Brambilla [18], aunque también se pueden encontrar puntos de vista similares en [15].

- Los robots son autónomos.
- Los robots están situados en el entorno y pueden actuar para modificarlo.
- Las capacidades de detección y comunicación de los robots son locales.
- Los robots no tienen acceso al control centralizado ni al conocimiento global.
- Los robots cooperan para abordar una tarea determinada.

6. DISEÑO METODOLÓGICO

A continuación se describirán las etapas y los diferentes pasos a seguir para la realización del proyecto. El presente proyecto se llevará a cabo en 3 etapas que son: elaboración y redacción del documento, desarrollo práctico, evaluación y análisis de resultados.

6.1. DESARROLLO PRÁCTICO

En esta etapa se construye el algoritmo y se realizan pruebas y experimentos de su funcionamiento en un entorno simulado. Se subdivide en tareas específicas, que son:

- **Diseño del algoritmo.**
  En esta fase se determina las entradas, las matrices de los estados, acciones y recompensas del algoritmo q-learning para el aprendizaje de los agentes del enjambre, y se procede a realizar la primera versión del código para un único agente.
- **Simulación de algoritmo diseñado.**
  Con base al algoritmo diseñado para un único agente se implementa la simulación con varios agentes que interactúen entre ellos utilizando software como Matlab o V–REP.
- **Ejecución y pruebas de simulación.**
  Se simula el algoritmo para varios agentes, realizando diferentes pruebas con diferentes números de robots y se observa el proceso de aprendizaje con indicadores de escalabilidad.

6.2. ANÁLISIS DE RESULTADOS

En esta etapa se analiza los resultados obtenidos de las diferentes simulaciones realizadas y según el proceso realizado en la etapa anterior se realiza el respectivo informe.

6.3. ELABORACIÓN Y REDACCIÓN DEL DOCUMENTO.

En esta etapa se trata todo lo relacionado con la elaboración del presente documento. Inicia con la elaboración del anteproyecto y concluye con la entrega del documento final. A continuación se presentan las tareas específicas en que se subdivide esta etapa:

- **Revisión bibliográfica.**
  Se busca, selecciona y clasifican los diferentes documentos usados como material bibliográfico para la construcción del anteproyecto.
- **Definición y planteamiento del problema.**
  Se describe, delimita y define el problema específico a tratar en esta investigación.
- **Redacción de la justificación.**
  Se exponen las razones principales que promueven el desarrollo del presente proyecto.
- **Redacción de antecedentes.**
  Se expone brevemente algunos trabajos con fines similares.
- **Definición de los objetivos del proyecto.**
  Se determina un objetivo general, que es la meta y el propósito del proyecto y, y se trazan los objetivos específicos a seguir durante el proyecto, que contribuyen a la realización del objetivo general.
- **Propuesta de la administración del proyecto.**
  Se propone el cronograma de actividades, en donde se ordena y se estima la duración aproximada en el tiempo para la realización de las actividades propuestas, y se establece el presupuesto, en donde se estima de la cantidad de dinero que se requiere para llevar a cabo el proyecto.

7. ADMINISTRACIÓN DEL PROYECTO

7.1. CRONOGRAMA DE ACTIVIDADES.

Actividades		Marzo				Abril				Mayo				Junio				Julio				Agosto				Septiembre				Octubre			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Elaboracion y redaccion del documento				**10 Semanas**																													
1	**Revisión bibliográfica**																																
2	**Definición y planteamiento del problema**																																
3	**Redacción de la justificación**																																
4	**Redacción de antecedentes**																																
5	**Definición de los objetivos del proyecto**																																
6	**Propuesta de la administración del proyecto**																																
Desarrollo práctico															**15 Semanas**																		
1	**Diseño del algoritmo**																																
2	**Diseño simulación**																																
3	**Ejecución y pruebas de simulación**																																
Análisis de resultados																														**4 Semanas**			
1	**Informe**																																
2	**Sustentación**																																

7.2. PRESUPUESTO

Rubros Financiables	Fuente	Total (COP)
Material Bibliográfico	Recurso institucional	$ 1.000.000
Equipos	Recurso institucional	$ 4.000.000
Software	Recurso institucional	$ 1.000.000
Asesoría	Recurso institucional	$ 1.500.000
Total (COP)		$ 7.500.000

Material Bibliográfico	Fuente	Total (COP)
Consulta base de datos y libros	Recurso institucional	$ 1.000.000
TOTAL		$ 1.000.000

Equipos	Cantidad	Fuente	Total (COP)
Equipo de computo	1	Recurso institucional	$ 4.000.000
TOTAL			$ 4.000.000

Software	Cantidad	Fuente	Total (COP)
Licencia Matlab	1	Recurso institucional	$ 1.000.000
V-Rep	1	Open source	$ -
TOTAL			$ 1.000.000

Asesoría				
Investigador	**Función en el proyecto**	**Horas**	**Costo Hora (COP)**	**Total (COP)**
Juan Manuel Calderón Chávez	Director	30	$ 50.000	$ 1.500.000
TOTAL				$ 1.500.000

8. TRABAJO PRELIMINAR

En este capítulo se mostrará el trabajo que se ha desarrollado previamente a este proyecto de grado, publicado como: “Formación líder-seguidor para enjambre de robots UAV basado en teoría de lógica difusa” [24], el cual inició como un trabajo de la electiva “SISTEMAS BIOINSPIRADOS”, en el que se propuso un algoritmo basado en un enfoque de lógica difusa para mantener la formación entre los robots que siguen al robot líder sin colisionar con otros agentes del grupo, y que ha inspirado a la realización de este proyecto de grado con un enfoque de aprendizaje por refuerzo.

8.1. Navegación del enjambre.

El enjambre de robots tiene que seguir a un robot líder sin colisionar con los otros robots, pero no muy lejos del robot más cercano. Para cumplir este requisito, se propuso un sistema basado en lógica difusa y este algoritmo se simuló utilizando MATLAB y V-Rep edición educativa para simulaciones de robot de drones. Este software permite la simulación de robots reales como drones, humanoides y varios robots móviles.

8.2. Variables iniciales.

Se tienen en cuenta tres variables de entrada: el número de robots sin incluir el robot líder (Nr), la distancia óptima al robot más cercano (Lr) y la distancia óptima al líder (Ll). Las pruebas de simulación se realizaron con diferentes valores para cada una de las variables de entrada.

8.3. Sistema difuso.

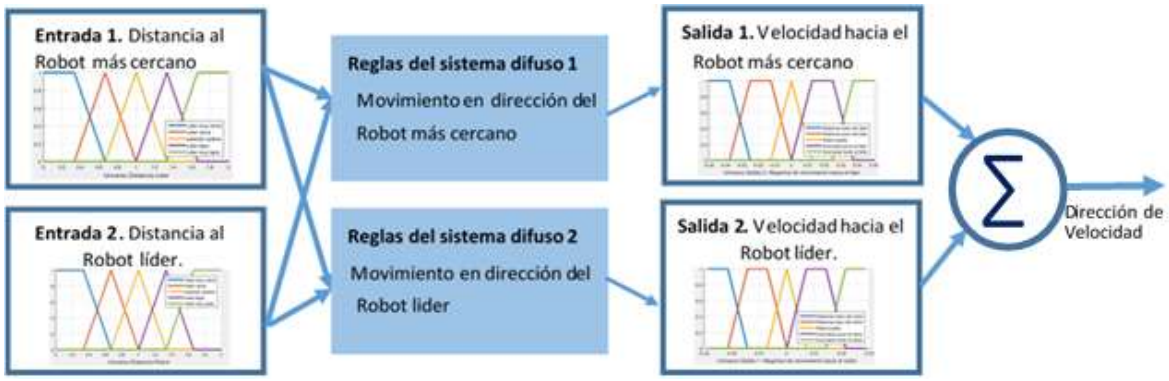


Fig. 8.1. Diagrama del sistema difuso completo.

Se propone un sistema difuso con el objetivo de cumplir los requisitos del desafío. Los robots deben navegar siguiendo a un líder y, al mismo tiempo, deben evitar colisionar con otros robots del enjambre. El sistema propuesto tiene dos subsistemas difusos usando el enfoque de Mamdani. El sistema completo tiene dos salidas, una que determina la velocidad hacia el robot más cercano y la segunda salida determina la velocidad hacia el líder. Además, cada sistema difuso tiene dos entradas: la primera es la distancia desde el robot al líder y la segunda es la distancia desde el robot al robot más cercano, por lo que se definieron dos subsistemas difusos para la salida 1, la salida 2 respectivamente. La figura 8.1 muestra el esquema del sistema completo, que incluye un sumador vectorial de las salidas para determinar la velocidad final y la dirección del robot de referencia.

Variables lingüísticas: Para el control difuso se definieron una serie de variables lingüísticas para los conjuntos difusos de las entradas y salidas, que se muestran en la tabla 1.

Tabla 1. Variables lingüísticas

Entradas	Salidas
MC = Muy cerca	RH = retirarse harto del robot
C = cerca	RP= retirarse poco del robot
PO = Posición optima	Q=Quieto
L = Lejos	AP = acercarse poco al robot
ML = muy lejos	AH = acercarse harto al robot

Entradas: El sistema difuso tiene dos entradas, la primera depende de la distancia al robot más cercano y la segunda es la distancia del Líder. Ambas entradas están compuestas por cinco conjuntos difusos definidos como Muy cerca, Cerca, Óptimo, Lejos y Muy lejos, como se muestra en la tabla 1. Estos conjuntos difusos se definen mediante una función difusa triangular para los tres conjuntos centrales y dos funciones trapezoidales para los conjuntos Muy Cercanos y Muy Lejos, como se muestra en la Figura 8.2. El universo del discurso para las entradas se define por la distancia óptima al robot líder (Ll) y el robot más cercano (Lr). Para estos, el valor óptimo se ubica en 1, la distancia más cercana es cero y la distancia más lejana es 2.

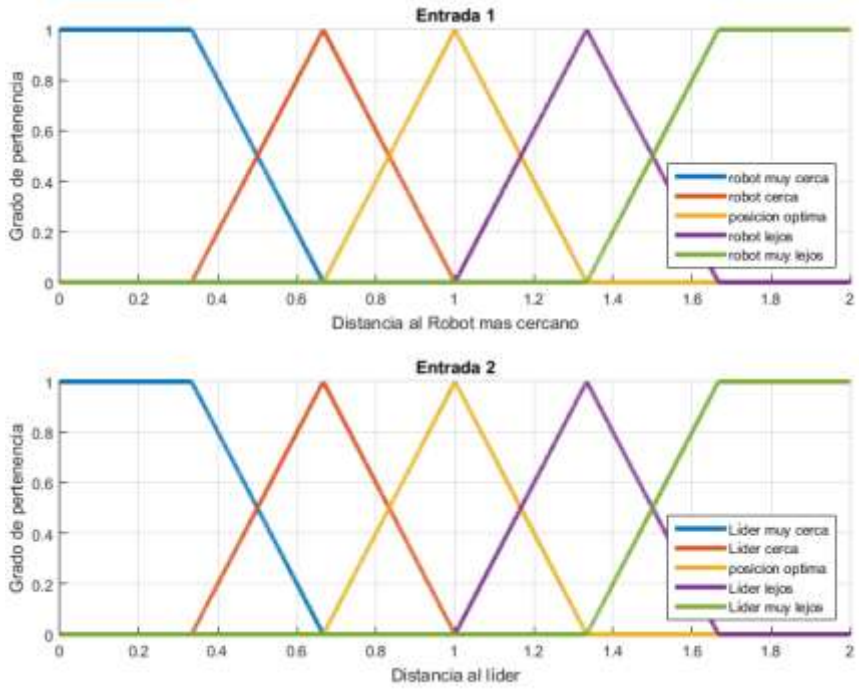


Fig. 8.2. Entradas del sistema difuso.

Salidas: el sistema difuso tiene dos salidas para determinar la velocidad de desplazamiento hacia el robot más cercano y el líder, respectivamente. Ambas salidas tienen cinco conjuntos difusos definidos como: (ver tabla 1) Retirarse Mucho, Retirarse Poco, Quieto, Acercarse Poco y

Acercarse Mucho. Estos conjuntos están representados por funciones trapezoidales, excepto el conjunto difuso " Quieto" que usa una función triangular como se muestra en la figura 8.3.

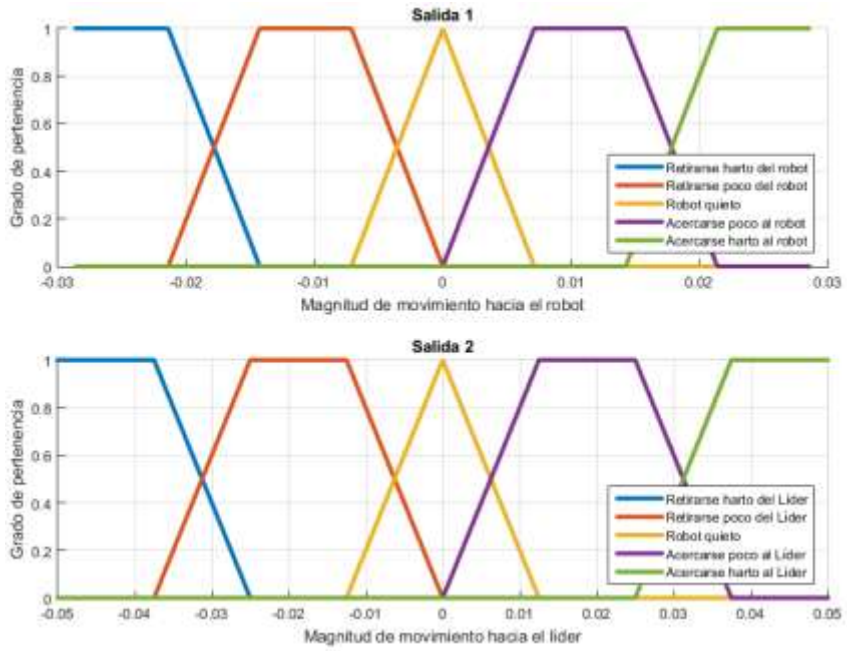


Fig. 8.3. Entradas del sistema difuso.

Cálculo de la distancia: el robot que está ejecutando la acción de movimiento se toma como robot de referencia. Se calculan todas las distancias entre cada robot de enjambre y el robot de referencia. La distancia más pequeña se toma como distancia al robot más cercano (Lr). Esta distancia es la primera entrada del sistema difuso. La distancia entre el robot de referencia y el robot líder (Ll) también se calcula, ya que esta es la segunda entrada del sistema difuso. El cálculo de distancias mencionado anteriormente se realiza para cada uno de los robots que forman el enjambre. El cálculo de la distancia se realiza utilizando la fórmula 1 de distancia en el plano cartesiano.

$$Ll, Lr = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad (1)$$

Reglas del sistema difuso: dado que el sistema tiene dos salidas, hay dos conjuntos de reglas de decisión. Las reglas de cada conjunto calculan el producto mínimo entre la pertenencia de dos conjuntos, cada conjunto pertenece a una entrada del sistema diferente. La salida calculada con las reglas pertenece a uno de los conjuntos de salida del sistema. La forma general de la regla se muestra en la ecuación 2.

$$\boldsymbol{Si}\ (entrada\ (1) es\ RP\ (i))\boldsymbol{y}\ (entrada\ (2) es\ RP\ (j))\boldsymbol{entonces}\ salida\ es\ RP\ (k) \qquad (2)$$

Donde i y j son los índices para la función de membresía de la entrada 1 y 2 respectivamente, y k es el índice de la salida del conjunto difuso que se está evaluando.

Las reglas de conjuntos difusos para las salidas 1 y 2 se muestran en las tablas 2 y 3, respectivamente.

Tabla 2. Conjunto de reglas salida 1: Movimiento en la dirección del robot más cercana.

		Distancia al robot más cercano				
		Muy cerca	**Cerca**	**Optima**	**Lejos**	**Muy Lejos**
Distancia al Líder	**Muy cerca**	RH	RP	Q	AP	AH
	Cerca	RH	Q	Q	Q	AH
	Optima	RH	Q	Q	Q	AH
	Lejos	RH	Q	Q	Q	AH
	Muy Lejos	RH	RP	Q	AP	AH

Tabla 3. Conjunto de reglas salida 2: Velocidad en la dirección del líder.

		Distancia al robot más cercano				
		Muy cerca	**Cerca**	**Optima**	**Lejos**	**Muy Lejos**
Distancia al Líder	**Muy cerca**	RH	RH	RH	RH	RH
	Cerca	RP	Q	Q	RP	RP
	Optima	Q	Q	Q	Q	Q
	Lejos	AP	AP	AP	AP	AP
	Muy Lejos	AH	AH	AH	AH	AH

Finalmente, para generar los datos de salida, se requiere el proceso de defuzzificación. Este método calcula una única cantidad escalar a partir de un conjunto difuso de salida difusa. Este trabajo utilizó el método bisector para el proceso de defuzzificación.

Cálculo final de la velocidad: La salida del sistema difuso que se describió en la sección anterior, representa la magnitud del movimiento hacia el robot más cercano y hacia el líder, ya que son dos sistemas difusos para dos salidas. Si la magnitud es negativa, significa que el robot debe alejarse del robot más cercano o del líder, respectivamente. Las dos salidas se suman vectorialmente, teniendo en cuenta el ángulo con respecto al líder y el robot cercano, para generar el vector de movimiento que apunta a la dirección en la que debe moverse el robot. Finalmente, la nueva posición de los robots se actualiza de acuerdo con la suma vectorial de las salidas del sistema difuso.

8.4. Experimentos y resultados

Se realizaron varios experimentos con el objetivo de probar el sistema de navegación propuesto basado en un enfoque de lógica difusa.

Posición inicial del robot

Dado que los robots no están inicialmente organizados en una formación específica, los agentes robóticos se ubicaron inicialmente al azar dentro del espacio de navegación. El algoritmo no depende de la posición inicial de los robots y, en su lugar, intenta organizarlos a través del proceso de navegación.

Experimentos

Se realizaron cinco experimentos, donde los tres primeros muestran el comportamiento del enjambre simulado utilizando MATLAB. Estos tres experimentos intentan mostrar cómo se agrupan los robots en torno al líder y cómo los robots siguen al líder cuando se mueve a un lugar diferente. En estos experimentos, el robot líder es una estrella roja y los robots enjambre están representados por círculos azules. Los dos últimos experimentos se realizaron utilizando VRep (software de simulación de robótica). Estos dos experimentos describieron la auto-organización de los drones alrededor del líder utilizando el algoritmo propuesto, y cómo el enjambre de drones navega siguiendo al líder cuando se está moviendo.

Experimento 1: Este experimento se realizó utilizando 8 robots. Estos robots fueron localizados aleatoriamente en un universo de 10X10 unidades de área. La Figura 8.4 muestra la secuencia sobre cómo los robots navegan desde la posición aleatoria original hasta el punto más cercano al líder. Al final de la figura 8.4 se muestra cómo se organiza el enjambre de robot alrededor del líder.

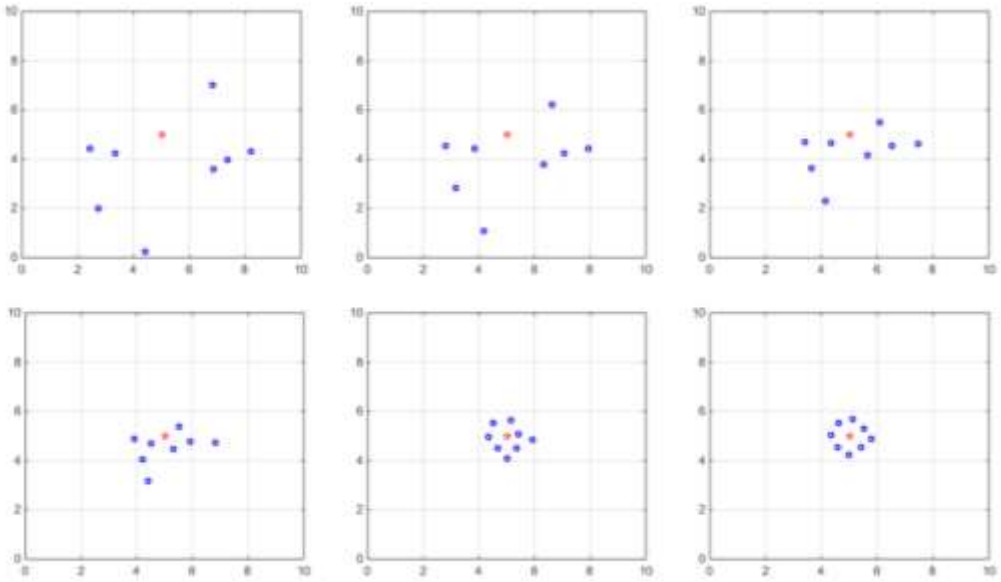


Fig. 8.4. Experimento 1

Experimento 2: este experimento utiliza 40 robots inicialmente localizados al azar. La Figura 8.5 muestra la secuencia completa sobre cómo el enjambre navega a través del campo que se acerca al líder. Una vez que los robots han llegado al punto más cercano al líder, muestra cómo el enjambre se organiza en capas alrededor del líder y mantiene la distancia entre los agentes del enjambre.

Experimento 3: Este experimento muestra al enjambre siguiendo al robot líder. El experimento se realizó con 15 agentes y se localizaron aleatoriamente al comienzo de la simulación. En La Figura 8.6 se muestra la secuencia de cómo los robots navegan cerca del líder y cuando el líder comienza a desplazarse a otra posición, todos los robots navegan siguiéndolo y manteniendo la formación establecida inicialmente. La traza de cada robot se muestra en líneas verdes en la

figura 8.6. Como se explicó inicialmente, la identificación del robot líder en rojo y los robots enjambre son azules.

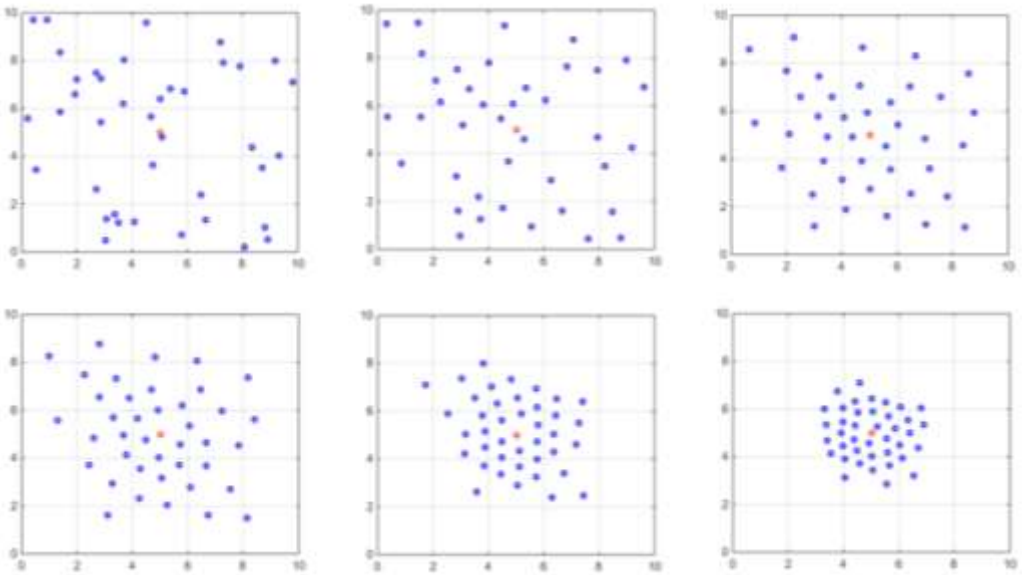


Fig. 8.5. Experimento 2

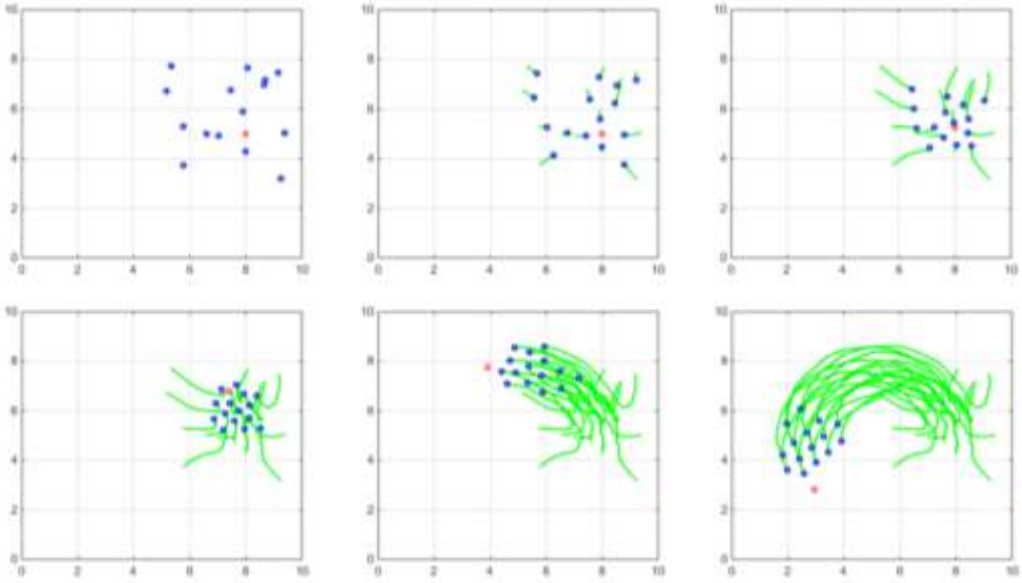


Fig. 8.6. Experimento 3

Experimento 4: Este experimento prueba el algoritmo propuesto utilizando VRep. La Figura 8.7 muestra el enjambre de drones navegando hacia el líder. La posición inicial de cada dron fue asignada al azar. Al final de la secuencia se muestra la posición final del enjambre de drones, muestra cómo los drones rodean al líder y mantienen la distancia al líder y la distancia entre otros drones.

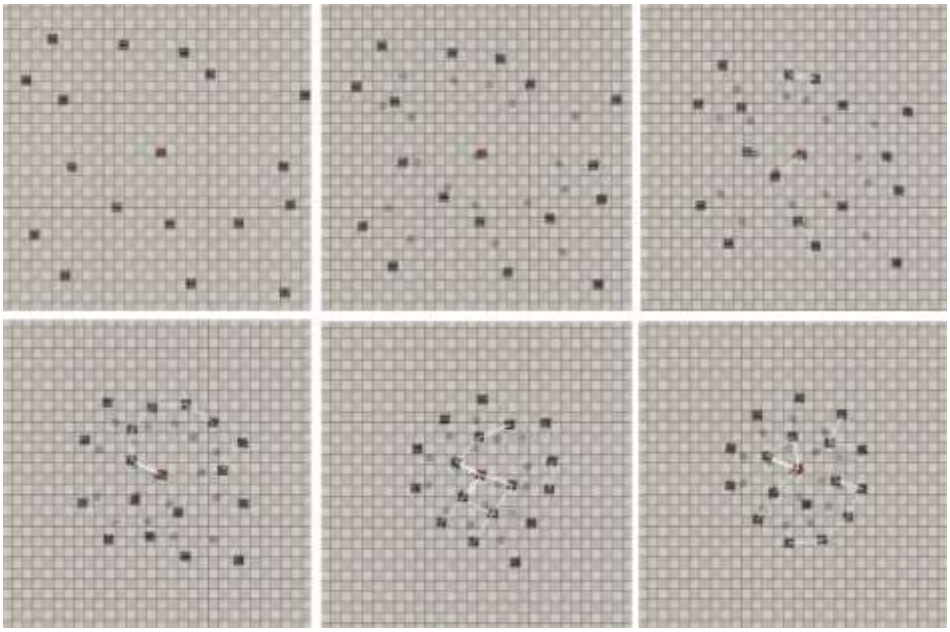


Fig. 8.7. Experimento 4

Experimento 5: Este es el último experimento y utiliza 15 drones para evaluar la navegación del enjambre de UAVs siguiendo al líder. La posición inicial de cada dron se definió aleatoriamente como experimentos previos. La Figura 8.8 muestra la secuencia de navegación donde los drones se acercan al líder y lo siguen a través de la navegación libre, solo siguiéndolo. La simulación muestra al enjambre manteniendo la formación y la distancia entre todos los robots del enjambre y evitando chocar con otros miembros del enjambre mientras navegan siguiendo al líder.

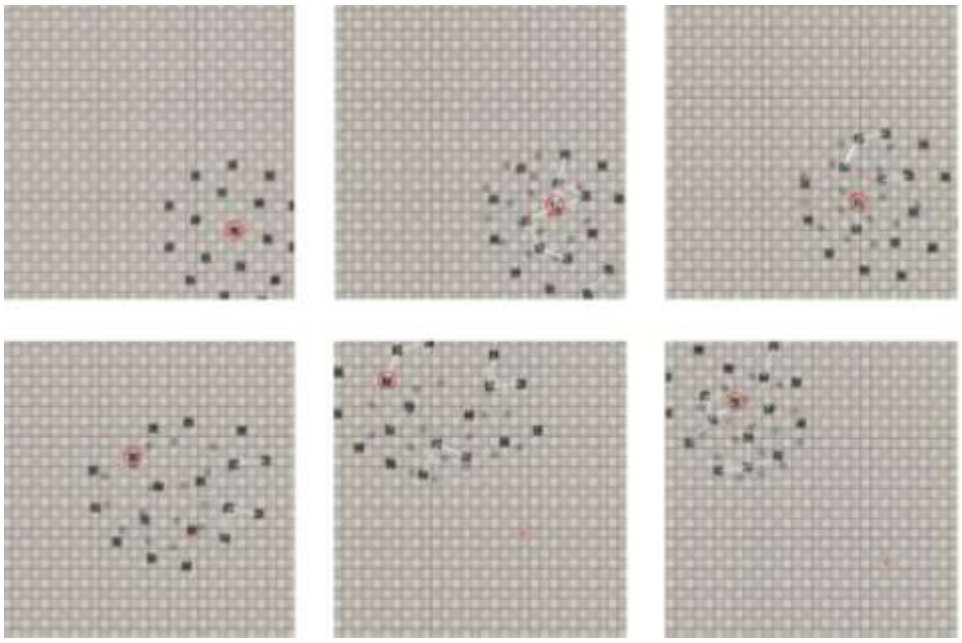


Fig. 8.8. Experimento 5

9. DISEÑO Y DESARROLLO DEL PROYECTO

En este capítulo se plantean dos soluciones con enfoques diferentes haciendo uso del aprendizaje por refuerzo. Se propone el uso de una función de exploración para abordar el dilema de la exploración-explotación y favorecer al proceso de aprendizaje de cada agente del enjambre. La política de exploración generada se utiliza en las dos soluciones propuestas. En este trabajo no se estudia el problema de la comunicación, y se asume que la comunicación de cada agente con sus vecinos ya está resuelta.

9.1. Enfoque Q-Learning

En las dos soluciones que se proponen se utilizó la ecuación clásica de Q-learning, que se muestra en (9.1.1) y se explicó en el capítulo 5 en la sub sección de Q-learning.

$$Q_{t+1}(S_t, A_t) = Q_t(S_t, A_t) + \alpha \left[r_{t+1} + \gamma \max_a Q_t(S_{t+1}, a) - Q_t(S_t, A_t) \right] \quad (9.1.1)$$

9.2. Política de exploración

La política es lo que determina cual será el comportamiento del agente en un determinado estado. Esta política le permite al agente escoger entre probar alguna acción nueva (exploración) o tomar la mejor acción conocida hasta el momento (explotación). La función de exploración que se propone para este trabajo facilita la posibilidad de explorar muchas opciones al inicio del proceso de aprendizaje, pero conforme pasa el tiempo, el agente escogerá la acción con mejor valor Q. En otras palabras, el agente iniciará el aprendizaje realizando mucha exploración, y una vez que ha aprendido, terminará ejecutando solo explotación. La función de exploración se describe en la ecuación 9.2.1.

$$a_t = \text{argmax}\left(Q(s_t, a) + \frac{k_p}{nP(s_t, a) + 1} \right) \quad (9.2.1)$$

Donde a_t es la acción a realizar, $Q(s_t, a)$ es el valor Q del el estado actual para todo el conjunto de acciones que el agente puede tomar, $nP(s_t, a)$ guarda el histórico de la cantidad de veces que se ha seleccionado la acción a en el estado s_t, y k_p es un parámetro constante tal que $k_p \gg Q(s_t, a)$. El aprendizaje comienza con todos los valores de la matriz $Q(s_t, a)$ y $nP(s_t, a)$ en cero, pues aún no ha sido explorada ninguna acción, por lo que la función queda con k_p dividido en 1. Conforme pasa el tiempo $nP(s_t, a)$ crece proporcionalmente al número de veces que se ha seleccionado el par estado-acción (s_t, a), y la función tiende a escoger la acción menos explorada para el estado s_t. Cuando $nP(s_t, a)$ es lo suficientemente grande, el factor $\frac{k_p}{nP(s_t,a)+1}$ es insignificante frente al valor de $Q(s_t, a)$ y la función tiende a seleccionar la mejor acción conocida, es decir, al final la política solamente hace explotación. Esta política de exploración se utiliza en las dos soluciones que se plantean a continuación.

9.3. Primera solución propuesta

En esta primera solución se asume que cada agente puede detectar la posición de sus dos vecinos más cercanos. Los estados se definen en función de la distancia de separación entre sus dos vecinos más cercanos en lugar de la ubicación espacial del robot. Se toma como referencia al robot que ejecuta la acción, y a sus dos vecinos más cercanos se les llama robot A

y robot B. La distancia del robot de referencia al Robot A o B se denota como RA o RB respectivamente, siendo RA<RB. En la sección 9.3.1 se presenta otra versión de esta solución modificada incluyendo un robot enemigo del enjambre, al cual los agentes deberán aprender a evitar.

Estados. Los estados se establecen con las distancias RA y RB, como se explicó anteriormente. Como los estados son discretos, las distancias RA y RB también se deben discretizar. En este caso se discretizaron las distancias RA y RB en 3 niveles, siendo 0 una distancia muy cerca, 1 una distancia buena y 2 que el robot está muy lejos.

Tabla 4. Estados de la primera solución propuesta

Estado	RA	RB
1	0	0
2	0	1
3	0	2
4	1	0
5	1	1
6	1	2
7	2	0
8	2	1
9	2	2

Con dos robots vecinos y tres niveles de distancia se forman 3^2 estados como se muestra en la tabla 4.

Acciones. Las acciones son el conjunto de todos movimientos y decisiones que el agente puede tomar, como girar a la izquierda o a la derecha, avanzar o retroceder. En este trabajo se asume que los robots del enjambre son holonómicos, que son robots cuyo número de grados de libertad controlables es igual al número de grados de libertad totales. Para estos caso las acciones normalmente son un número finito de posibles direcciones en las que el robot se puede mover, sin embargo para este planteamiento se definen 5 acciones que son: acercarse al robot A (acción 1), retirarse del robot A (acción 2), acercarse al robot B (acción 3), retirarse del robot B (acción 4) o permanecer quieto (acción 5).

Política de premios y castigos. La política de recompensa se basa en reducir o aumentar la distancia de RA y RB lo suficiente hasta mantenerla en el nivel discretizado 1, que equivale a una distancia buena como se explicó anteriormente. La política de recompensa se representa como una función a trozos como sigue:

$$R_{t-1} = \begin{cases} RA - (Nd - 1), & a_t = 1 \\ -RA, & a_t = 2 \\ RB - (Nd - 1), & a_t = 3 \\ -RB, & a_t = 4 \\ -1, & a_t = 5, s_t \neq 5 \\ 100, & a_t = 5, s_t = 5 \end{cases}$$

Donde a_t es la acción seleccionada, s_t es el estado actual y Nd son los niveles de discretización de la distancia RA y RB. Para este caso $Nd = 3$.

En el algoritmo 9.3.1 se describen los pasos para esta solución.

Algoritmo 9.3.1: Algoritmo de la primera solución.

I. **Paso 1:** Calcular todas las distancias entre los robots del enjambre y para cada robot mapear las distancias discretizadas de los dos robots más cercanos (distancia RA y RB, según el planteamiento descrito en esta sección).
II. **Paso 2:** Con las distancias discretizadas del paso 1 calcular el estado actual para cada robot.
III. **Paso 3:** Con la política de exploración de la ecuación (9.2.1), se decide la acción a realizar para cada robot y se ejecuta.
IV. **Paso 4:** De acuerdo con la política de premios y castigos, se actualizan los valores de la matriz Q con el estado actual y el estado anterior usando la ecuación (9.1.1). Se actualizan los estados y se regresa al paso 1.

9.3.1. Versión de la primera solución con un enemigo

Ahora se hará una modificación al planteamiento anterior para incluir un enemigo del enjambre, el cual los agentes deberán aprender a evitar. Para esto se toma la distancia al robot enemigo y se discretiza de la misma forma que se hizo con RA y RB. El conjunto de estados al incluir un enemigo se muestran en la tabla 5, donde la X representa condiciones de no importa con el fin de simplificar la cantidad de estados.

Tabla 5. Estados de la primera solución propuesta con un enemigo

Estado	RE	RA	RB
1	0	X	X
2	1	X	X
3	2	0	0
4	2	0	1
5	2	0	2
6	2	1	0
7	2	1	1
8	2	1	2
9	2	2	0
10	2	2	1
11	2	2	2

Para esta modificación se agrega una acción al conjunto de acciones descrito anteriormente que será la sexta acción: retirarse del enemigo (acción 6).

En la versión clásica de Q-Learning se asocia una recompensa a cada par estado- acción. La nueva política de recompensas se muestra en la tabla 6, donde se tienen las 6 acciones con sus siglas correspondientes, como se describe a continuación:

ARA: acercarse al robot A (acción 1)
RRA: retirarse del robot A (acción 2)
ARB: acercarse al robot B (acción 3)
RRB: retirarse del robot B (acción 4)
Q: permanecer quieto (acción 5)
RRE: retirarse del enemigo (acción 6)

Tabla 6. Recompensas para la primera solución con un enemigo

Estado	ARA	RRA	ARB	RRB	Q	RRE
1	-10	-10	-10	-10	-10	10
2	0	0	0	0	0	0
3	-2	0	-2	0	-1	-1
4	-2	0	-1	-1	-1	-1
5	-2	0	0	-2	-1	-1
6	-1	-1	-2	0	-1	-1
7	-1	-1	-1	-1	100	-1
8	-1	-1	0	-2	-1	-1
9	0	-2	-2	0	-1	-1
10	0	-2	-1	-1	-1	-1
11	0	-2	0	-2	-1	-1

En el algoritmo 9.3.1.1 se describen los pasos para esta solución.

Algoritmo 9.3.1.1: Algoritmo de la primera solución con un enemigo.

I. **Paso 1:** Calcular todas las distancias entre todos los robots del enjambre incluyendo el robot enemigo, y para cada robot mapear las distancias discretizadas del robot enemigo y de los dos robots más cercanos.
II. **Paso 2:** Con las distancias discretizadas del paso 1 calcular el estado actual para cada robot.
III. **Paso 3:** Con la política de exploración de la ecuación (9.2.1), se decide la acción a realizar para cada robot y se ejecuta.
IV. **Paso 4:** De acuerdo con la política de premios y castigos de la tabla 6, se actualizan los valores de la matriz Q con el estado actual y el estado anterior usando la ecuación (9.1.1). Se actualizan los estados y se regresa al paso 1.

9.4. Segunda solución propuesta

Para esta solución se hace uso de varios conceptos presentados por F.L. Lewis en [32]. Se define un radio de repulsión rr, en el que cada agente busca mantenerse alejado de otros agentes para evitar colisionar con sus vecinos. El vecindario de colisión para el agente i se define en (9.4.1).

$$\boldsymbol{N_i^c = \{j : r_{ij} \leq rr\}} \quad (\mathbf{9.4.1})$$

Fuente [32].

Se define un segundo radio de interacción $ri > rr$, en el que el agente puede ver a los vecinos. Todos los agentes fuera del radio de interacción ri no son percibidos por el robot de referencia i^{th}. El vecindario de interacción se define en (9.4.2).

$$\boldsymbol{N_i = \{j : r_{ij} \leq \mathrm{ri}\}} \quad (\mathbf{9.4.2})$$

Fuente [32].

Donde r_{ij} es el la distancia euclidiana del agente i^{th} al agente j^{th}, y se define en (9.4.3), con $\big(p_i(t), q_i(t)\big)$ como la posición del agente i^{th} en el plano (x, y) y x_i el vector de posición del agente i^{th}.

$$\boldsymbol{r_{ij} = \|x_j - x_i\| = \sqrt{(p_j - p_i)^2 + (q_j - q_i)^2}} \quad (\mathbf{9.4.3})$$

Fuente [32].

Se asume que cada agente puede detectar la posición de sus vecinos dentro de su radio de interacción.

Estados. Los estados se establecen con la cantidad de robots dentro de ri y la cantidad de robots dentro de rr por cuadrante. Se definen subconjuntos de $\boldsymbol{N_i}$ para cada cuadrante local del i^{th} robot de referencia. El vecindario de interacción del agente i^{th} para el primer cuadrante se define como $\boldsymbol{N_{i_1} = \{j : j \in N_i \,;\, p_j > p_i \,;\, q_i > q_i\}}$, para el segundo cuadrante $\boldsymbol{N_{i_2} = \{j : j \in N_i \,;\, p_j < p_i \,;\, q_i > q_i\}}$, para el tercer cuadrante se define como $\boldsymbol{N_{i_3} = \{j : j \in N_i \,;\, p_j < p_i \,;\, q_i < q_i\}}$ y finalmente $\boldsymbol{N_{i_4} = \{j : j \in N_i \,;\, p_j > p_i \,;\, q_i < q_i\}}$ para el cuarto cuadrante. De forma general, el vecindario de interacción del i^{th} agente para el cuadrante C se denota como $\boldsymbol{N_{i_C}}$.

De la misma forma se definen subconjuntos del vecindario de colisión como:

$$\boldsymbol{N_{i\,1}^c = \{j : j \in N_i^c \,;\, p_j > p_i \,;\, q_i > q_i\}}$$
$$\boldsymbol{N_{i\,2}^c = \{j : j \in N_i^c \,;\, p_j < p_i \,;\, q_i > q_i\}}$$
$$\boldsymbol{N_{i\,3}^c = \{j : j \in N_i^c \,;\, p_j < p_i \,;\, q_i < q_i\}}$$
$$\boldsymbol{N_{i\,4}^c = \{j : j \in N_i^c \,;\, p_j > p_i \,;\, q_i < q_i\}}$$

Para cada cuadrante del robot i^{th} respectivamente. De forma general, el vecindario de colisión del agente i^{th} para el cuadrante C se denota como $\boldsymbol{N_{i\,C}^c}$.

Los estados están definidos por la cardinalidad de los subconjuntos $\boldsymbol{N_{i_C}}$ y $\boldsymbol{N^{c}_{i\,C}}$, definidos anteriormente. Es decir, se cuenta para cada cuadrante del robot i los j^{th} vecinos dentro de los radios ri y rr.
Para simplificar los estados se toma el número del cuadrante con el máximo número de robots para cada radio.

Número del cuadrante para ri

$$Cri = argmax\left(card(\boldsymbol{N_{i_1}}), card(\boldsymbol{N_{i_2}}), card(\boldsymbol{N_{i_3}}), card(\boldsymbol{N_{i_4}})\right)$$

Número del cuadrante para rr

$$Crr = argmax\left(card(\boldsymbol{N^{c}_{i\,1}}), card(\boldsymbol{N^{c}_{i_2}}), card(\boldsymbol{N^{c}_{i_3}}), card(\boldsymbol{N^{c}_{i_4}})\right)$$

En la tabla 7 se muestra el conjunto de estados para esta solución propuesta.

Tabla 7. Estados del segundo sistema propuesto

Estado	Ri	Rr	Cuadrante ri	Cuadrante rr
1	0	0	X	X
2	1	0	0	X
3	1	0	1	X
4	1	0	2	X
5	1	0	3	X
6	X	1	X	0
7	X	1	X	1
8	X	1	X	2
9	X	1	X	3

Donde la x representa condiciones de no importa con el fin de simplificar la cantidad de estados. En la columna de ri y rr, se coloca 0 para representar que no hay ningún robot en el cuadrante para cada radio respectivamente, y 1 para indicar cuál es el cuadrante con mayor cantidad de robots para cada radio. El estado 1 es para cuando el agente está aislado del grupo y no hay vecinos dentro del radio ri. Las columna Cuadrante ri y Cuadrante rr son para representar los cuatro cuadrantes locales del robot de referencia para cada uno de los radios, donde 0, 1, 2 y 3 representan el primer, segundo, tercer y cuarto cuadrante respectivamente.

En la figura 9.4.1 se muestra un ejemplo para el estado 2, donde se muestra con un punto blanco al robot de referencia ubicado en la posición (0,0) y se muestra la división de los cuatro cuadrantes locales del robot de referencia. La zona de color rojo representa el área dentro del radio de repulsión o vecindario de colisión. La zona de color verde representa el vecindario de interacción, que es el área dentro del radio de interacción y fuera del radio de repulsión. Los otros robots del enjambre se representan con puntos de color verde claro. Para este ejemplo no hay vecinos dentro del radio de repulsión, y se tiene que el cuadrante con mayor cantidad de vecinos en el vecindario de interacción es el primer cuadrante con cuatro robots. Según la tabla 7, el estado 2 es el único estado que no tiene vecinos dentro del radio de repulsión y cuyo cuadrante con máximo número de robots en el

vecindario de interacción es el primer cuadrante (cuadrante 0 según la tabla), por lo tanto el estado para para el agente de referencia de la figura 9.4.1 es el estado 2.

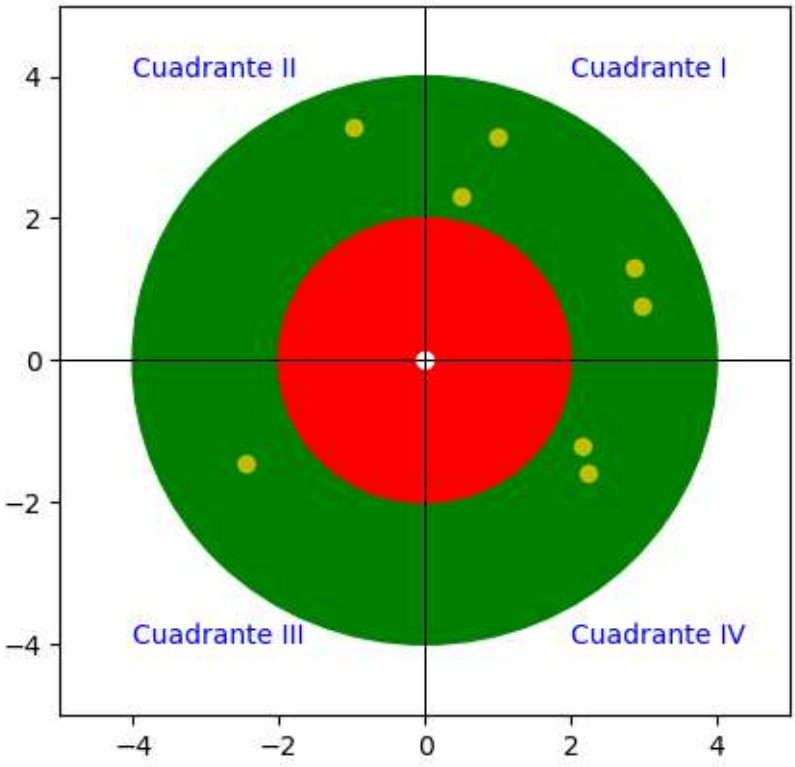


Fig. 9.4.1. Ejemplo estado 2.

Fuente autor.

Acciones. Para este planteamiento se definen 8 acciones, que son acercarse al cuadrante 1 de ri (acción 1), acercarse al cuadrante 2 de ri (acción 2), acercarse al cuadrante 3 de ri (acción 3), acercarse al cuadrante 4 de ri (acción 4), retirarse del cuadrante 1 de rr (acción 5), retirarse del cuadrante 2 de rr (acción 6), retirarse del cuadrante 3 de rr (acción 7) y retirarse del cuadrante 4 de rr (acción 8).

En la figura 9.4.2 se muestra las primeras cuatro acciones mencionadas anteriormente que consisten en acercarse al cuadrante correspondiente del radio de interacción. La dirección de movimiento para cualquiera de estas acciones se calcula con la ecuación mostrada en (9.4.4).
En la figura 9.4.3 se muestran las últimas cuatro acciones que consisten en alejarse del cuadrante correspondiente del radio de repulsión. El ángulo de movimiento final para cualquiera de estas acciones se calcula con la sumatoria mostrada en la ecuación (9.4.5).
Al seleccionar alguna acción de movimiento hacia cualquiera de los cuadrantes, la ejecución del movimiento se realiza en la dirección del vector resultante de la sumatoria de todos los j^{th} agentes dentro del radio ri o rr para el cuadrante de la acción seleccionada.

Es decir, si por ejemplo la acción seleccionada es acercarse al cuadrante 1 de ri (acción 1), el movimiento que se debe ejecutar efectivamente es en dirección del cuadrante 1 pero la dirección o el ángulo del movimiento final se calcula con (9.4.4).
Por otra parte, si la acción seleccionada es retirarse del cuadrante 1 de rr (acción 5), el movimiento que se debe ejecutar es en dirección del cuadrante 3, según como indican el sentido de las flechas de la figura 9.4.3, y el movimiento final se calcula con la ecuación mostrada en (9.4.5).
Los pasos seguidos en esta solución y el proceso de ejecución de la acción seleccionada con la política de exploración se describen en el algoritmo 9.4.1. En $(9.4.4)$ se muestra el protocolo de control para las acciones de ri y en $(9.4.5)$ para las acciones de rr.

$$\boldsymbol{u_{iC} = \sum_{j \in N_{i_C} \setminus N^c_{i_C}} k \frac{(x_j - x_i)}{r_{ij}}} \qquad \boldsymbol{(9.4.4)}$$

$$\boldsymbol{u^c_{iC} = -\sum_{j \in N^c_{i_C}} k \frac{(x_j - x_i)}{r_{ij}}} \qquad \boldsymbol{(9.4.5)}$$

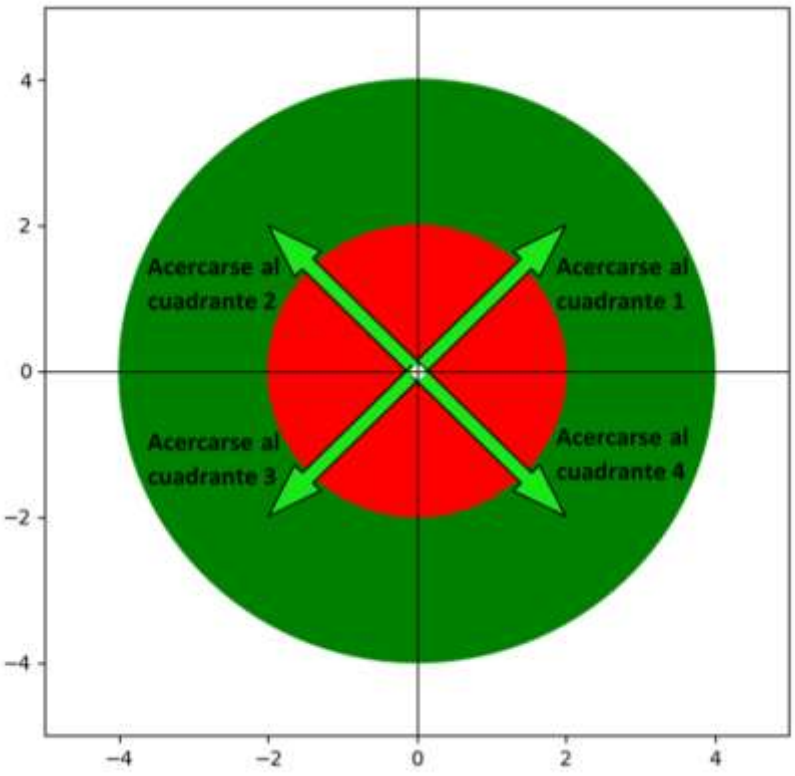


Fig. 9.4.2. Acciones en relación al radio de interacción.

Fuente autor.

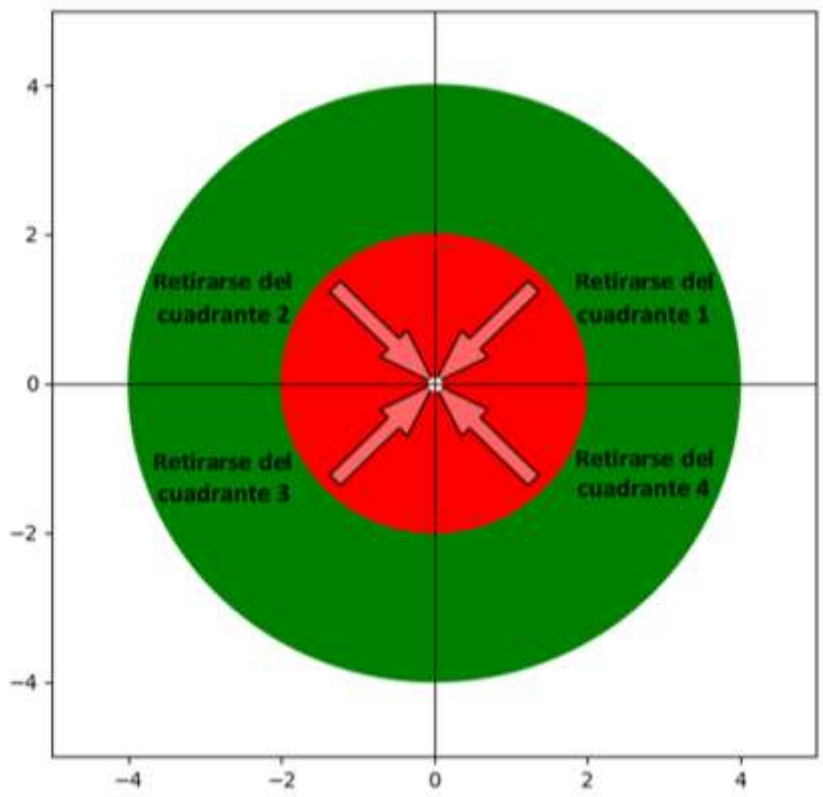


Fig. 9.4.3. Acciones en relación al radio de repulsión.

Fuente autor.

Algoritmo 9.4.1: Algoritmo de la segunda solución.

I. **Paso 1:** Calcular los subconjuntos por cuadrante del vecindario de interacción $\boldsymbol{N_{i_C}}$ y los subconjuntos por cuadrante del vecindario de colisión $\boldsymbol{N_{i_C}^c}$ para todos los robot.

II. **Paso 2:** Calcular la cardinalidad de los subconjuntos $\boldsymbol{N_{i_C}}$ y $\boldsymbol{N_{i_C}^c}$, contando la cantidad de vecinos que pertenecen a cada conjunto, y calcular por cada robot el número del cuadrante con el máximo número de robots para cada radio.

III. **Paso 3:** Con los resultados del paso 2 calcular el estado actual de cada robot teniendo en cuenta la tabla 7.

IV. **Paso 4:** Con la política de exploración de la ecuación (9.2.1), se decide la acción a realizar para cada robot.

V. **Paso 5:** Ejecutar la acción seleccionada teniendo en cuenta que para las acciones relacionadas con acercarse a los cuadrantes de ri, que son las primeras cuatro acciones, se utiliza la ley de control que se muestra en la ecuación (9.4.4). Para las acciones relacionadas con alejarse de los cuadrantes de rr (últimas cuatro acciones), se utiliza la ecuación (9.4.5).

VI. **Paso 6:** De acuerdo con la política de premios y castigos se actualizan los valores de la matriz Q con el estado actual y el estado anterior usando la ecuación (9.1.1). Se actualizan los estados y se regresa al paso 1.

Política de premios y castigos. La política de recompensas se define con la siguiente ecuación:

$$R_{t-1} = \begin{cases} 0 \; si \; s_t = 1 \\ 1 \; si \; a_t = s_t - 1 \\ -1 \; cualquier \; otro \; caso \end{cases}$$

9.4.1. Versión de la segunda solución con un enemigo

Al igual que en la primera solución propuesta, se hará una pequeña modificación al planteamiento anterior para incluir un enemigo al enjambre. Para esto se incluye un nuevo estado, que es para indicar cuando el robot enemigo entra en el radio $1.5 * rr$ del robot de referencia. El conjunto de estados al incluir un enemigo se muestran en la tabla 8.

Tabla 8. Estados de la segunda solución propuesta con un enemigo

Estado	Ri	rr	rr Enemigo	Cuadrante ri	Cuadrante rr
1	0	0	0	X	X
2	1	0	0	0	X
3	1	0	0	1	X
4	1	0	0	2	X
5	1	0	0	3	X
6	X	1	0	X	0
7	X	1	0	X	1
8	X	1	0	X	2
9	X	1	0	X	3
10	X	X	1	X	X

Donde la x representa condiciones de no importa con el fin de simplificar la cantidad de estados. En la columna de ri y rr, se coloca 0 para representar que no hay ningún robot en el cuadrante para cada radio respectivamente, y 1 para indicar cuál es el cuadrante con mayor cantidad de robots para cada radio. En la columna $rr \; enemigo$, se coloca 0 para representar que no hay enemigo dentro del radio y 1 para representar que si hay un enemigo.

Al conjunto de acciones se agrega una nueva acción que será retirarse del enemigo. La política de recompensa permanece igual.

$$R_{t-1} = \begin{cases} 0 \; si \; s_t = 1 \\ 1 \; si \; a_t = s_t - 1 \\ -1 \; cualquier \; otro \; caso \end{cases}$$

10. PRUEBAS Y RESULTADOS

Las dos soluciones propuestas de aprendizaje por refuerzo explicadas previamente se implementaron en Matlab para los experimentos con múltiples agentes. Los sistemas propuestos se evalúan a través de varios experimentos variando el tamaño del enjambre. Inicialmente se simulan los algoritmo solo en Matlab probando las dos propuestas con dos variaciones, que consiste primero en dejar una sola matriz Q compartida para todos los agentes del enjambre de tal forma que cada agente puede acceder y modificar la misma matriz Q, y segundo, en asignar para cada agente una matriz Q de forma individual sin que puedan compartir sus experiencias. El enjambre se modela mediante grafos tomando cada agente del enjambre como un nodo o vértice dentro del grafo con el fin de medir la conectividad y el grado de cada vértice. Las gráficas de la secuencia de movimiento son obtenidas durante el aprendizaje. También se realizarán pruebas del algoritmo vinculando Matlab con el entorno de simulación de robótica V-Rep (Virtual Robot Experimentation Platform).

10.1. Configuración de parámetros para la simulación.

Para simular las dos soluciones propuestas se deben establecer los valores iniciales de los parámetros del algoritmo Q-Learning y los definidos en el planteamiento. Para el algoritmo de Q-Learning, el factor de la tasa de aprendizaje se estableció en $\alpha = 0.2$, el factor de descuento $\gamma = 0.5$. El valor del factor de exploración inicial para la política de exploración se estableció en $k_p = 100$. Para la primera solución se discretizaron las distancias RA y RB en tres niveles, siendo el primer nivel muy cerca, una distancia $r_{ij} < 0.8$ unidades de distancia, el segundo nivel un intervalo con $0.8 \leq r_{ij} \leq 1.2$ unidades de distancia, y el ultimo nivel, $r_{ij} > 1.5$ para representar que el robot está muy lejos. Para la segunda solución propuesta se define un radio de atracción o radio de interacción $ri = 5$ unidades de distancia, y un radio de repulsión $rr = 1$. Estos valores se establecieron de forma experimental.

10.2. Pruebas de la primera solución propuesta en Matlab

Para esta solución, basada en la búsqueda de los dos vecinos más cercanos, se realizaron 4 experimentos en total utilizando Matlab, donde los dos primeros experimentos muestran el comportamiento del enjambre compartiendo la misma matriz Q, con diferentes números de robots, y los dos últimos experimentos se realizaron asignando matrices Q para que cada agente individualmente tenga su propia matriz Q.

10.2.1. Experimento con matriz Q compartida

- **Experimento 1 primera solución.**

  Este experimento se realizó con 10 robots inicialmente posicionados de forma aleatoria en un espacio de 10X10 unidades de área, con todos los robots compartiendo la misma matriz Q. Para este experimento se realizaron 400 episodios.

  En la figura 10.1 se muestra la secuencia completa de como los robots se agrupan buscando a sus dos vecinos más cercanos. En la secuencia se muestra el rastro del movimiento de cada agente a través del proceso de aprendizaje. Al final de la secuencia, en la figura 10.1 se muestra la posición final de los robots formando dos clústeres. En la figura 10.2 se muestra como terminaron conectados los robots en la posición final mediante un grafo no dirigido, en el que los vértices representan la conexión entre

agentes. La conexión entre los agentes se representa de forma matricial con la matriz de adyacencia o conectividad, matriz A, donde se representa con 1 los agentes que están conectados y con 0 los que no tiene comunicación. La conexión se establece cuando los vecinos están a una distancia igual o menor al segundo nivel discreteado.

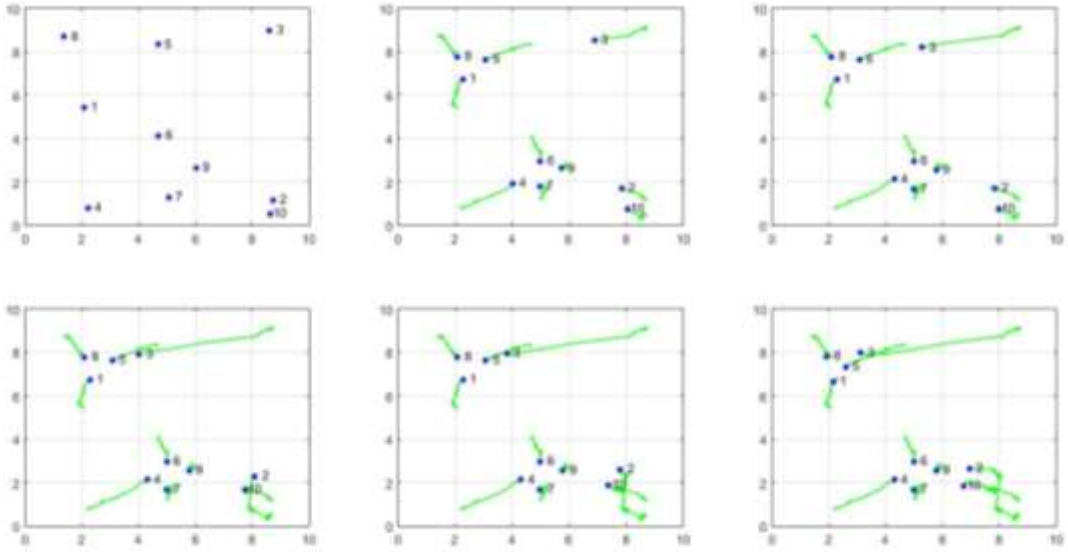


Fig. 10.1 Secuencia experimento 1 primera solución.

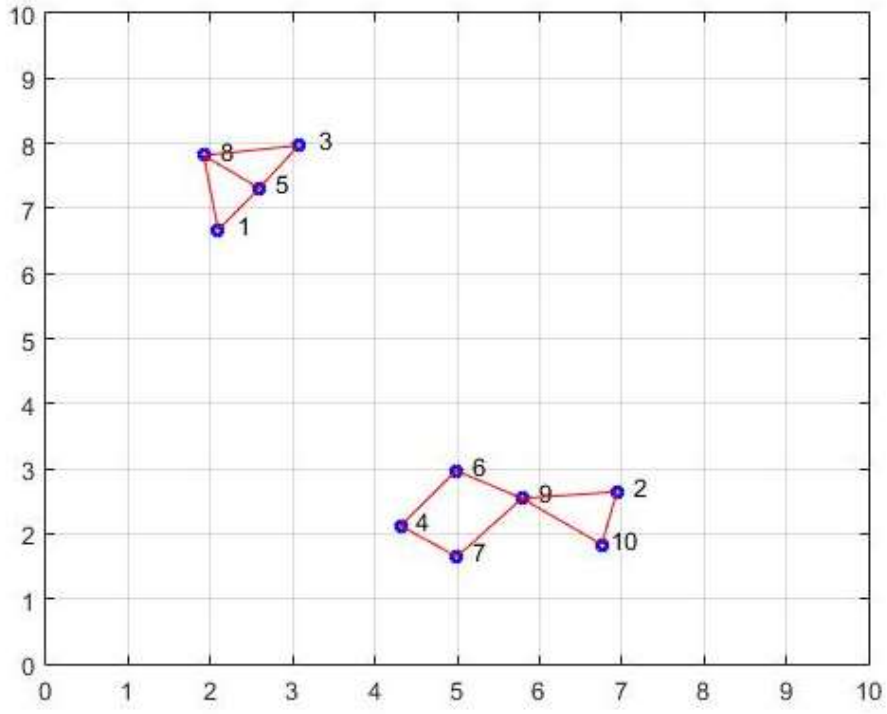


Fig. 10.2 Posición final de los robots.

La matriz A es de tamaño NxN, donde N es la cantidad de agentes. En (10.2.1.1) se muestra la matriz de adyacencia del grafo formado por la interconexión de los robots en la posición final que se muestra en la figura 10.2.

Matriz de adyacencia $(\mathbf{10.2.1.1})$

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

El grado de conectividad de cada vértice es la cantidad de conexiones que tiene con los otros nodos, y se representa de forma matricial con la matriz diagonal D, que contiene el grado de cada vértice en su diagonal. En (10.2.1.2) se muestra la matriz D del grafo formado en la figura 10.2.

Matriz de grado de conectividad $(\mathbf{10.2.1.2})$

$$D = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix}$$

La matriz laplaciana está definida como $L = D - A$. Ordenando los valores propios de L se tiene que $|\lambda 1| \leq |\lambda 2| < \cdots \leq |\lambda \mathrm{N}|$. Con el segundo valor propio $\lambda 2$ de la matriz laplaciana, se puede determinar si el grafo es o no conexo. El grafo es conexo si se cumple que $\lambda 2 > 0$, según [14, 32]. A continuación se muestran los valores propios de la matriz laplaciana L.

$$\lambda_1 = 0;\ \lambda_2 = 0;\ \lambda_3 = 0.76;\ \lambda_4 = 2;\ \lambda_5 = 2;$$
$$\lambda_6 = 3;\ \lambda_7 = 3;\ \lambda_8 = 4;\ \lambda_9 = 4;\ \lambda_{10} = 5.24$$

Puesto que $\lambda_2 = 0$, se puede decir que el grafo no es conexo. Esto también se puede ver en la figura 10.2, ya que se forman dos clústeres o sub grafos que no tienen conexión.

En la figura 10.3 se muestra cómo evoluciona el grado de conectividad a través del tiempo durante todo el experimento. Después del episodio 213, no se registró cambios en el grado de conectividad de cada agente. Esto se puede interpretar como que los robots aprendieron a agruparse con sus dos vecinos más cercanos, o que los agentes han llegado a un estado en el que conservan la distancia deseada con sus dos vecinos. De la figura 10.3 se puede apreciar que la mayor parte de los agentes se estabilizan en

un grado de conectividad igual a dos con un total de 7 robots, seguido por el grado de conectividad tres con dos robots y finalmente el grado cuatro con un robot.

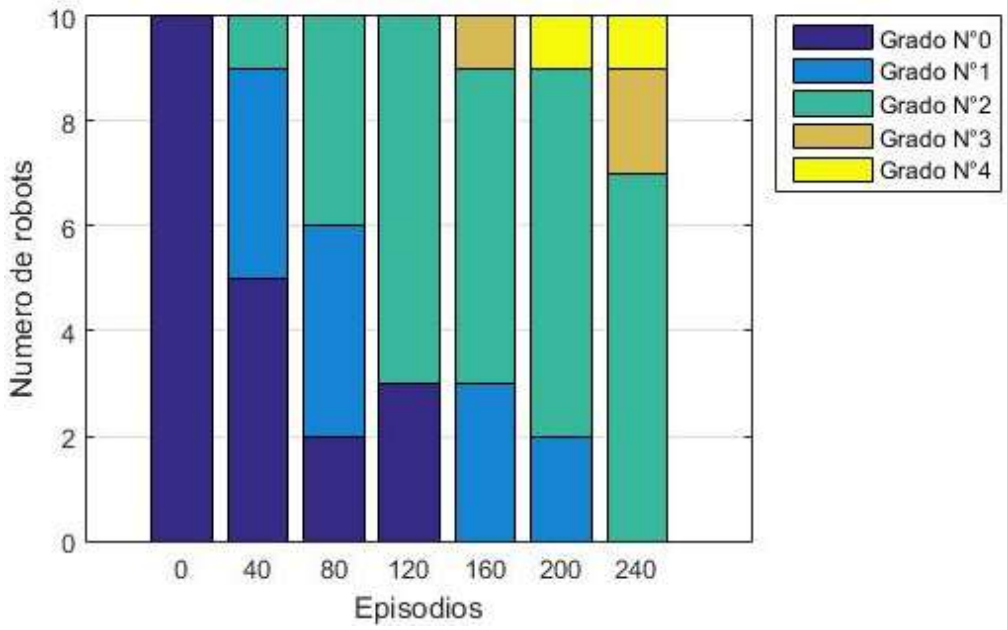


Fig. 10.3 Grado de conectividad a través del proceso de aprendizaje.

En la figura 10.4 se muestra como el promedio del grado de conectividad de todos los agentes evoluciona a través del proceso de aprendizaje para los 250 primeros episodios. El promedio del grado de conectividad se estabiliza después del episodio 213 en un valor de 2.4, como se observa en la figura 10.4.

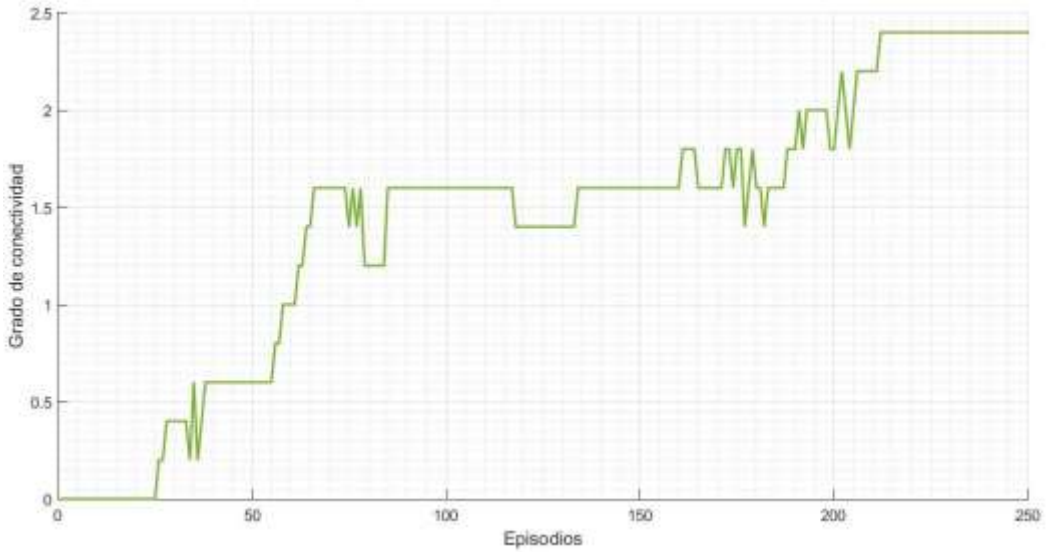


Fig. 10.4 Promedio del grado de conectividad a través del proceso de aprendizaje.

- **Experimento 2 primera solución.**

  Para este experimento se utilizan 15 robots ubicados aleatoriamente al principio de la simulación y se desarrolló en 400 episodios. En este experimento todos los robots comparten la misma matriz Q. En la figura 10.5 se muestra la secuencia del enjambre de robots navegando mientras aprenden a permanecer cerca de sus dos vecinos más cercanos.

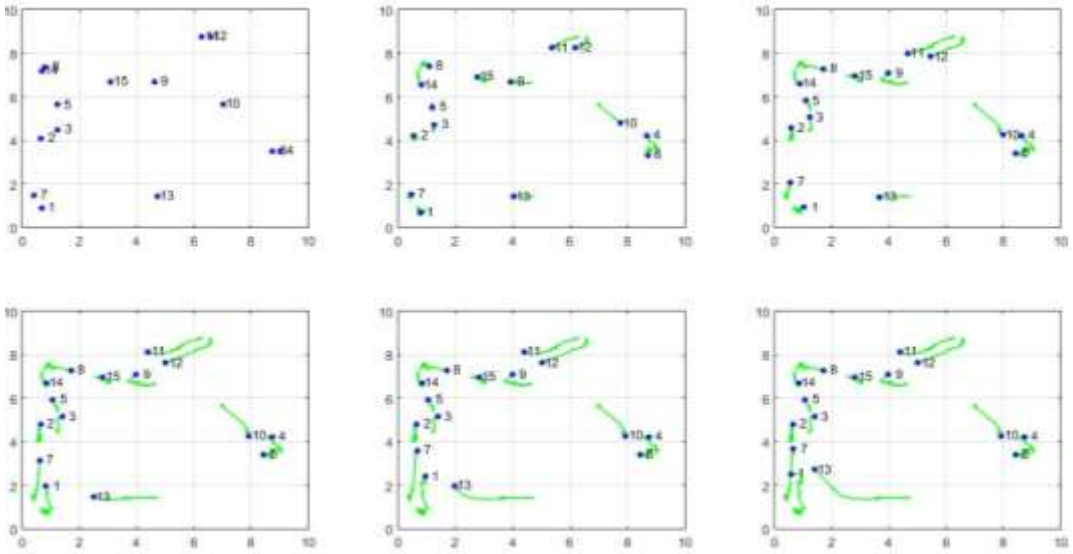


Fig. 10.5 Secuencia experimento 2 primera solución.

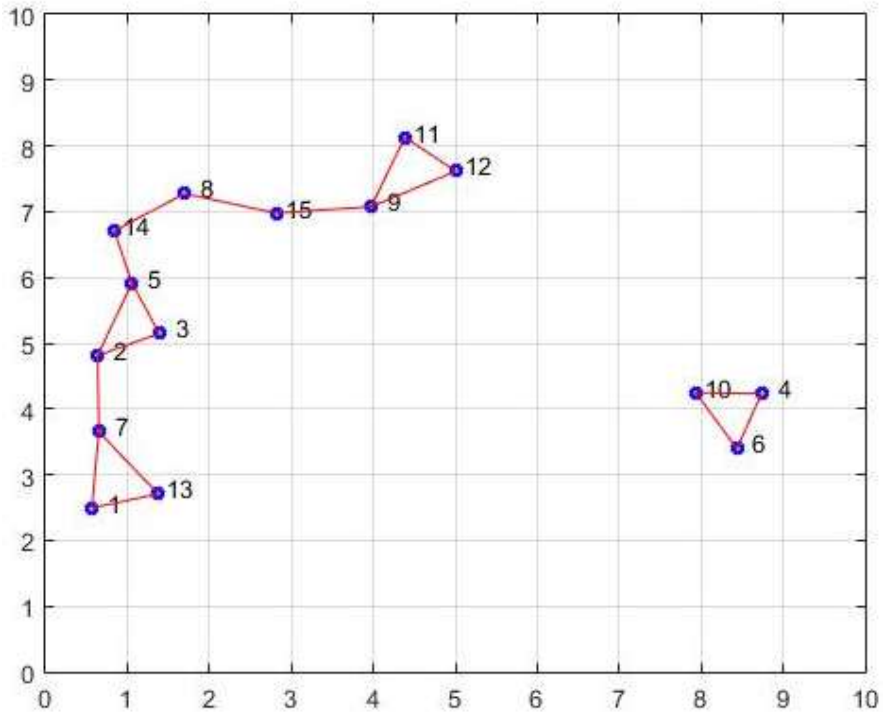


Fig. 10.6 Posición final de los robots.

En la figura 10.6 se muestra la posición final de los robots y como terminaron conectados los robots representándolos en forma de grafos no dirigidos. En la posición final se forman dos clústeres o sub grafos que no están conectados, por lo que es un grafo no conexo. La conexión se establece cuando los vecinos están a una distancia igual o menor al segundo nivel discreteado.

La matriz de adyacencia que se forma del grafo de la figura 10.6, se muestra en la matriz (10.2.1.3), que es la representación de la comunicación entre los agentes.

El grado de conectividad de cada agente de la figura 10.6 se representa de forma matricial en la matriz de grado mostrada en (10.2.1.4), cuya diagonal equivale al grado de conexión de cada agente de la figura 10.6.

Matriz de adyacencia $(\mathbf{10.2.1.3})$

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Matriz de grado de conectividad $(\mathbf{10.2.1.4})$

$$D = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix}$$

Se calcula la matriz laplaciana del grafo de la figura 10.6, haciendo $L = D - A$, y se calculan los valores propios de la matriz laplaciana para comprobar matemáticamente

que el grafo que se forma en la figura 10.6 es un grafo no conexo. Los valores propios se muestran a continuación.

$$\lambda_1 = 0;\ \lambda_2 = 0;\ \lambda_3 = 0.09;\ \lambda_4 = 0.39;\ \lambda_5 = 0.80;$$
$$\lambda_6 = 1.77;\ \lambda_7 = 3;\ \lambda_8 = 3;\ \lambda_9 = 3;\ \lambda_{10} = 3;$$
$$\lambda_{11} = 3;\ \lambda_{12} = 3;\ \lambda_{13} = 4;\ \lambda_{14} = 4.24;\ \lambda_{15} = 4.69$$

Como $\lambda_2 = 0$, se comprueba matemáticamente que el grafo de la Figuera 10.6 no es conexo, como se mencionó anteriormente.

En la figura 10.7 se muestra la evolución del grado de conectividad durante el experimento.

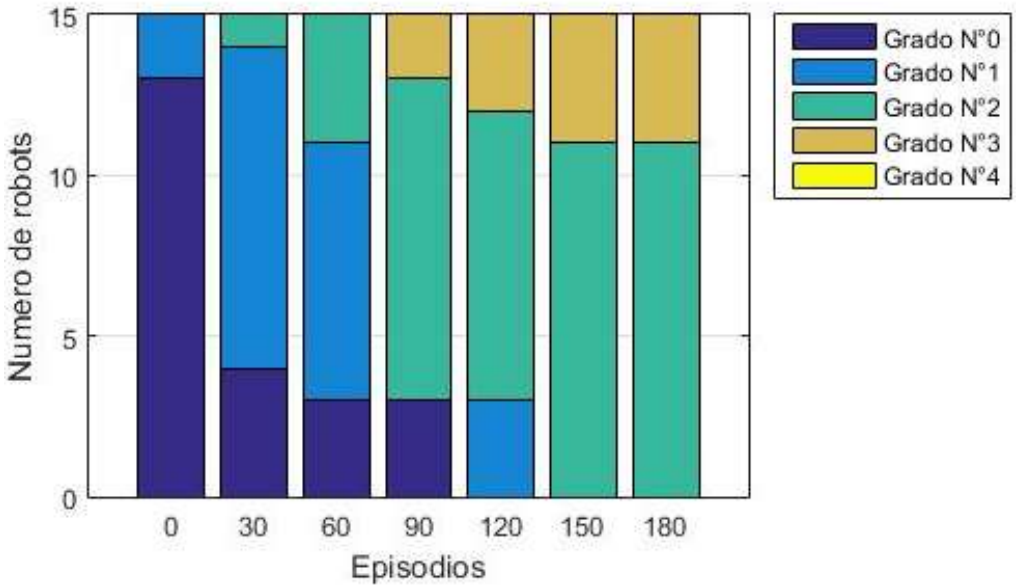


Fig. 10.7 Grado de cada robot a través del proceso de aprendizaje.

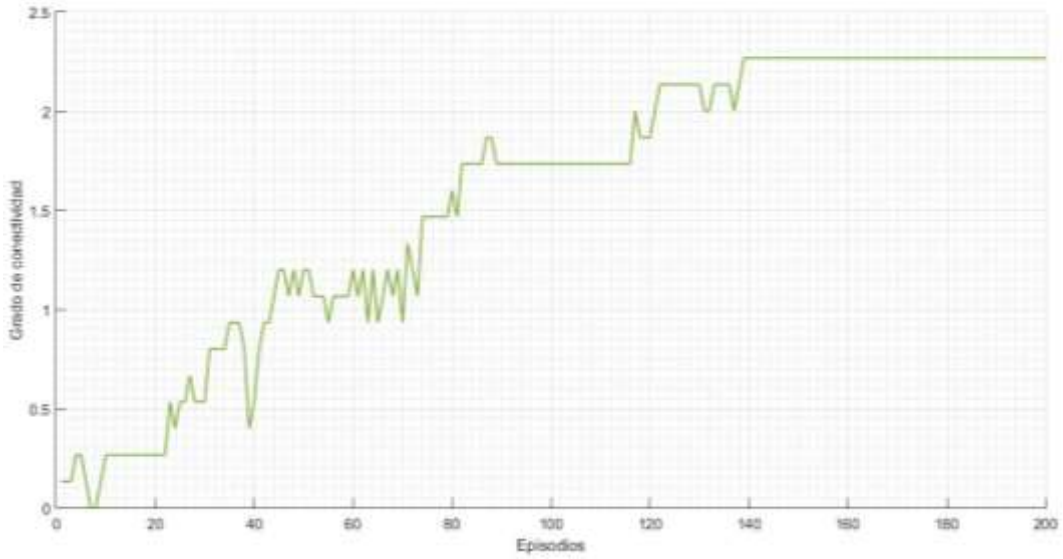


Fig. 10.8 Promedio del grado de conectividad a través del proceso de aprendizaje.

Se puede apreciar en la figura 10.7 que la mayoría de robots tienden a estabilizar su grado de conectividad en dos, y otros pocos se estabilizan con un grado de tres. El grado de conectividad para todos los robots se estabiliza después del episodio 141, con 11 robots que convergen al grado de conectividad 2, y 4 robots que convergen al grado de conectividad 3.

En la figura 10.8 se muestra el promedio del grado de conectividad en los primeros 200 episodios de este experimento. Después del episodio 141 el grado promedio de los robots se tiende a estabilizar en el valor de 2.267, como se muestra en la figura 10.8.

10.2.2. Experimento con matriz Q individual

- **Experimento 3 primera solución.**

  En este experimento se asignó una matriz Q independiente para cada robot de forma individual. En el experimento se utilizaron 20 robots ubicados al azar en un espacio de 15x15 unidades de área y se realizó durante 1200 episodios. La serie de movimientos de los robots durante el proceso de aprendizaje se muestra en la figura 10.9.

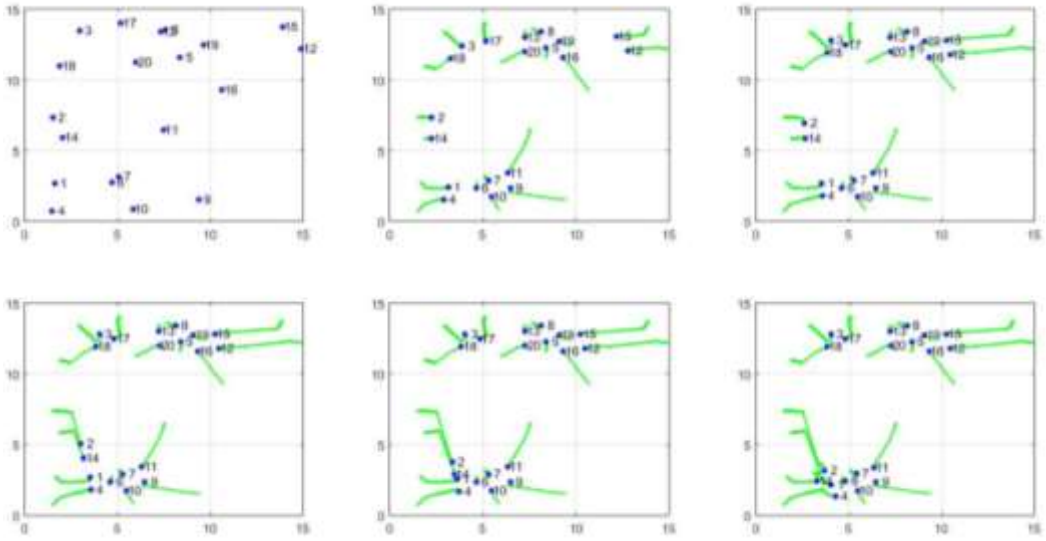


Fig. 10.9 Secuencia experimento 3 primera solución.

La posición de los robots al final del experimento que se muestra en la figura 10.10, es el resultado final del este experimento al hacer que los robots aprendan a permanecer cerca a sus dos vecinos más cercanos. En la figura 10.10 también se muestra como es la conexión que se formó al final entre los agentes, representando la conexión de los agentes con una línea que los une. Los robots se agrupan formando 3 sub grafos o clústeres como se muestra en la figura 10.10. La conexión se establece cuando los vecinos están a una distancia igual o menor al segundo nivel discreteado.

La conexión final también se representa de forma matricial con la matriz de adyacencia que se muestra en la matriz (10.2.2.1), y el grado de conectividad de cada agente se representa de forma matricial con la matiz (10.2.2.2).

Con la matriz de adyacencia y la matriz de grado de conectividad se calcula la matriz laplaciana haciendo $L = D - A$. Con el segundo valor propio de la matriz laplaciana se

puede conocer si el grafo es un grafo conexo o no, si cumple que $\lambda_2 > 0$. Para este caso se tiene que $\lambda_2 = 0$, por lo que el grafo es un grafo no conexo.

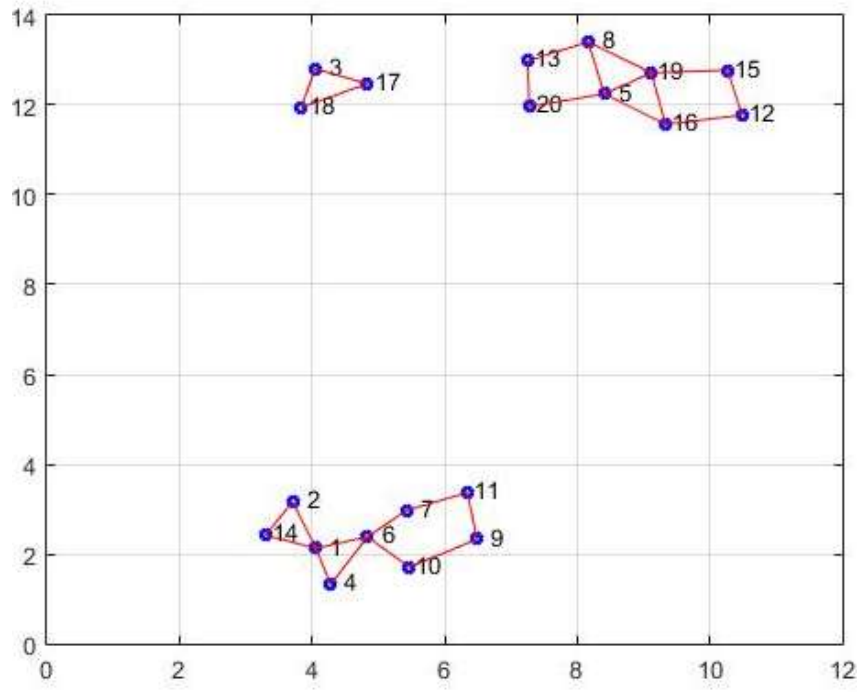


Fig. 10.10 Posición final de los robots.

Matriz de adyacencia $(\mathbf{10.2.2.1})$

$$A = \begin{pmatrix}
0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

$$D = \begin{pmatrix}
4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0
\end{pmatrix}$$

Matriz de grado de conectividad $(\mathbf{10.2.2.2})$

En la figura 10.11 se muestra el grado de conectividad durante el proceso de aprendizaje y también se puede apreciar que al inicio todos los agentes empiezan con grado de conectividad en cero, que es la zona azul en la gráfica, y conforme pasan los episodios se forman las conexiones. Al final del experimento 14 robots terminan con un grado de dos, cuatro robots con grado cuatro y dos robots con grado tres, como se muestra en la figura 10.11 El grado de conectividad permanece constante después del episodio 890 y no se registró cambios.

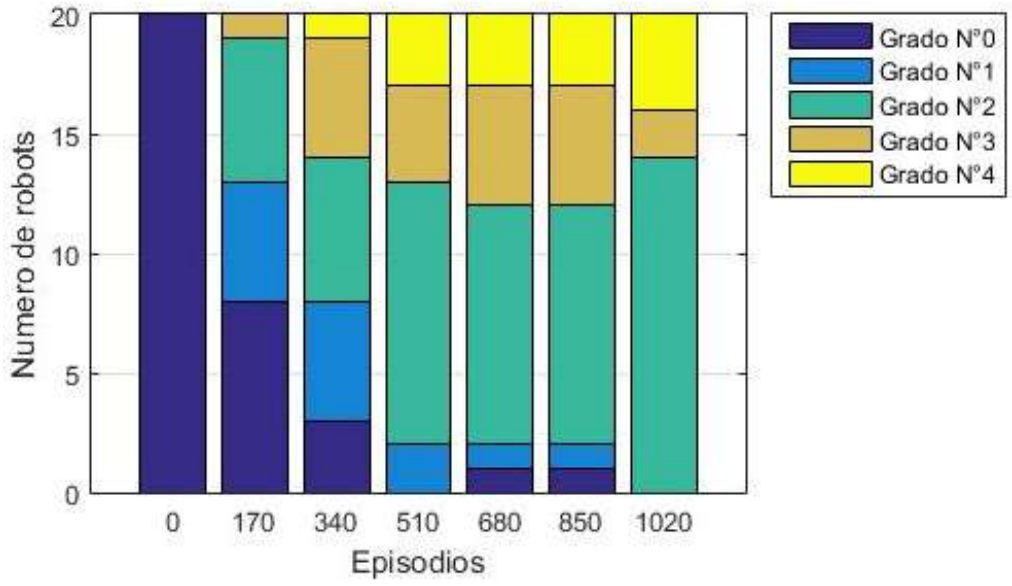


Fig. 10.11 Grado de conectividad a través del proceso de aprendizaje.

En la figura 10.12 se muestra el promedio del grado de conectividad de todos los robots durante el proceso de aprendizaje. En este experimento los robots en promedio tienden a estabilizar su grado de conectividad en 2.5, como se muestra en la figura 10.12

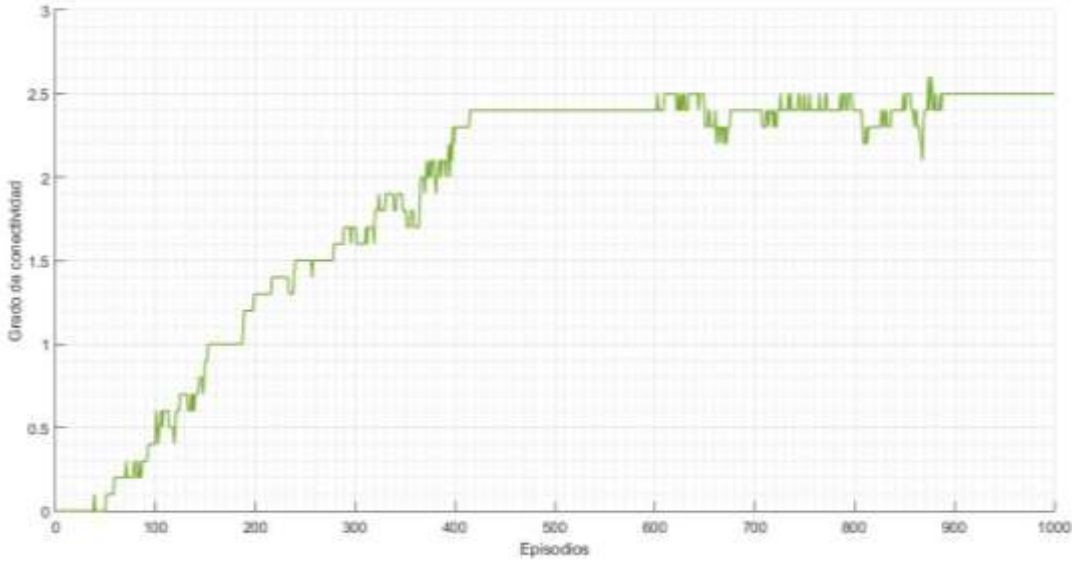


Fig. 10.12 Promedio del grado de conectividad a través del proceso de aprendizaje.

- **Experimento 4 primera solución.**
  Para este experimento se utilizaron 40 robots ubicados en un espacio de 20x20 unidades de área. Se utilizó una matriz Q para cada robot de forma individual. La secuencia completa de los movimientos se muestra en la figura 10.13.

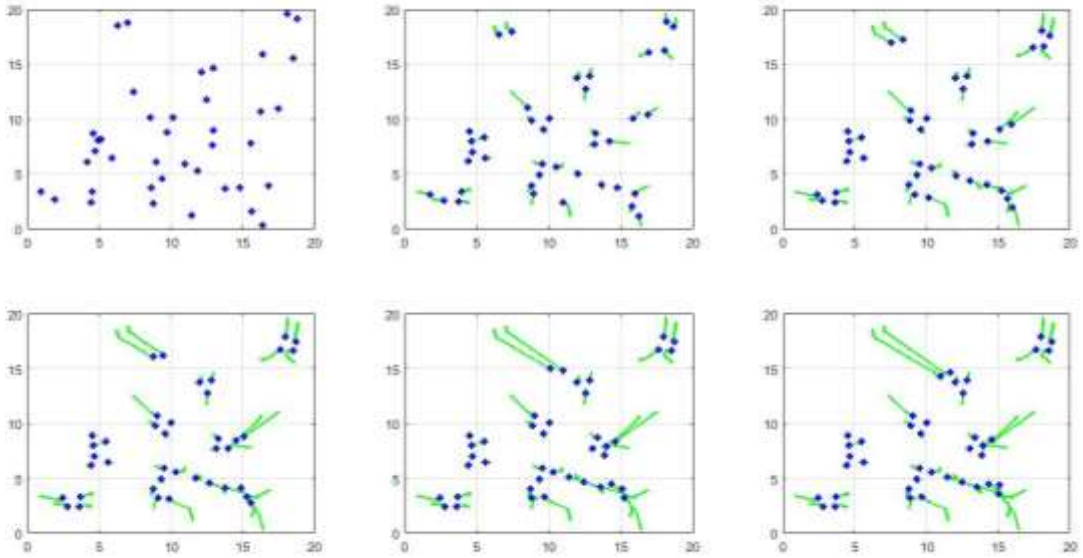


Fig. 10.13 Secuencia experimento 4 primera solución.

En la figura 10.14 se muestra la topología del grafo de conexión que se formó cuando los robots alcanzaron la posición final. La conexión se establece cuando los vecinos están a una distancia igual o menor al segundo nivel discreteado. Los robots se organizan formando 7 subgrupos o clústeres, tal como se muestra en la figura 10.14.

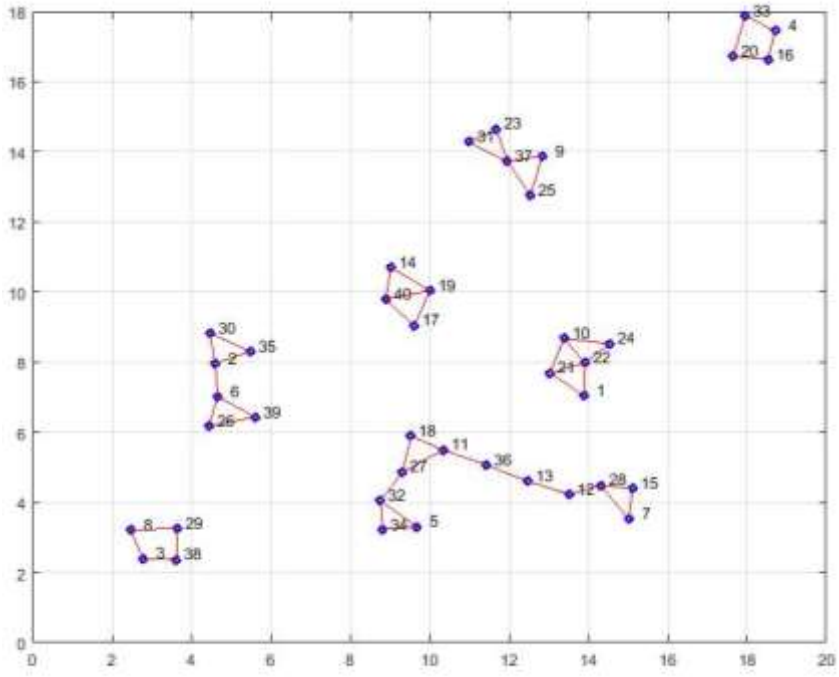


Fig. 10.14 Posición final de los robots.

Para este caso la matriz de adyacencia A y la matriz del grado de conectividad D que se forman, tienen dimensiones de 40x40, lo que supera el tamaño para incluirlas en este documento. Los valores propios de la matriz laplaciana del grafo de la figura 10.14 se muestran a continuación.

$$\lambda_1 = 0;\ \lambda_2 = 0;\ \lambda_3 = 0;\ \lambda_4 = 0;\ \lambda_5 = 0;\ \lambda_6 = 0;\ \lambda_7 = 0;\ \lambda_8 = 0.09;$$
$$\lambda_9 = 0.39;\ \lambda_{10} = 0.44;\ \lambda_{11} = 0.80;\ \lambda_{12} = 1;\lambda_{13} = 1.59;\ \lambda_{14} = 1.77;\ \lambda_{15} = 2$$
$$\lambda_{16} = 2;\ \lambda_{17} = 2;\ \lambda_{18} = 2;\ \lambda_{19} = 2;\ \lambda_{20} = 3;\ \lambda_{21} = 3;\ \lambda_{22} = 3;\ \lambda_{23} = 3;\ \lambda_{24} = 3;$$
$$\lambda_{25} = 3;\ \lambda_{26} = 3;\ \lambda_{27} = 3;\ \lambda_{28} = 3;\ \lambda_{29} = 3;\ \lambda_{30} = 4;\ \lambda_{31} = 4;\ \lambda_{32} = 4;\ \lambda_{33} = 4;$$
$$\lambda_{34} = 4;\lambda_{35} = 4.24;\ \lambda_{36} = 4.41;\ \lambda_{37} = 4.56;\ \lambda_{38} = 4.69;\ \lambda_{39} = 5;\ \lambda_{40} = 5$$

Con el segundo valor propio de la matriz laplaciana se puede comprobar matemáticamente que el grafo de la figura 10.14 es un grafo no conexo, puesto que se cumple que $\lambda_2 = 0$.

En la figura 10.15 se muestra el grado de conectividad de todos los agentes durante el experimento. El experimento se realizó en un total de 900 episodios. Después del episodio 517 el grado de conectividad de los todos los agentes se estabilizo, teniendo 28 agentes con grado de conectividad 2, 10 agentes con grado 3 y 2 agentes con grado de conectividad en 4.

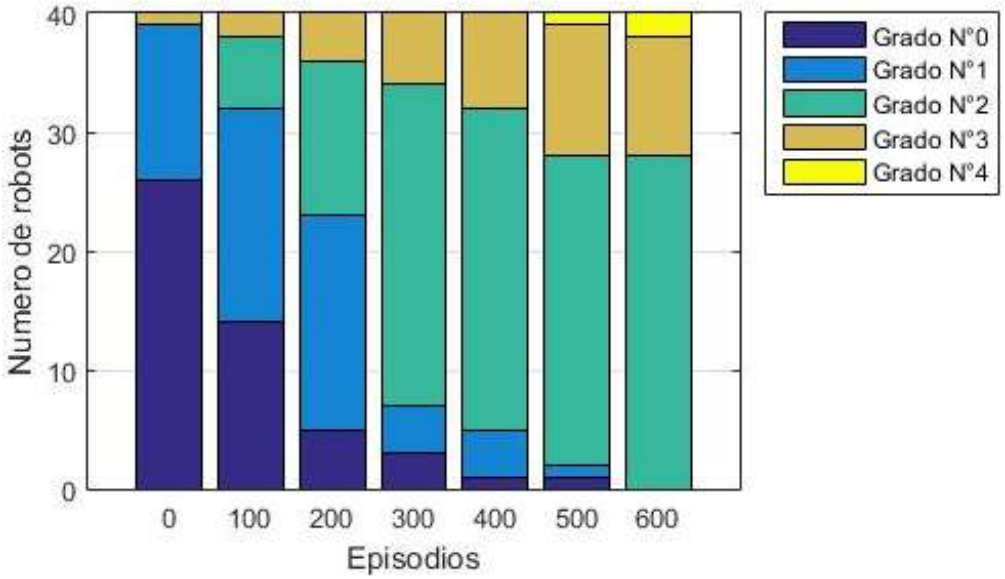


Fig. 10.15 Grado de cada robot a través del proceso de aprendizaje.

En la figura 10.16 se muestra el promedio del grado de conectividad de todos los agentes. Después del episodio 517 el promedio del grado de conectividad converge al valor de 2.35, como se muestra en 10.16.

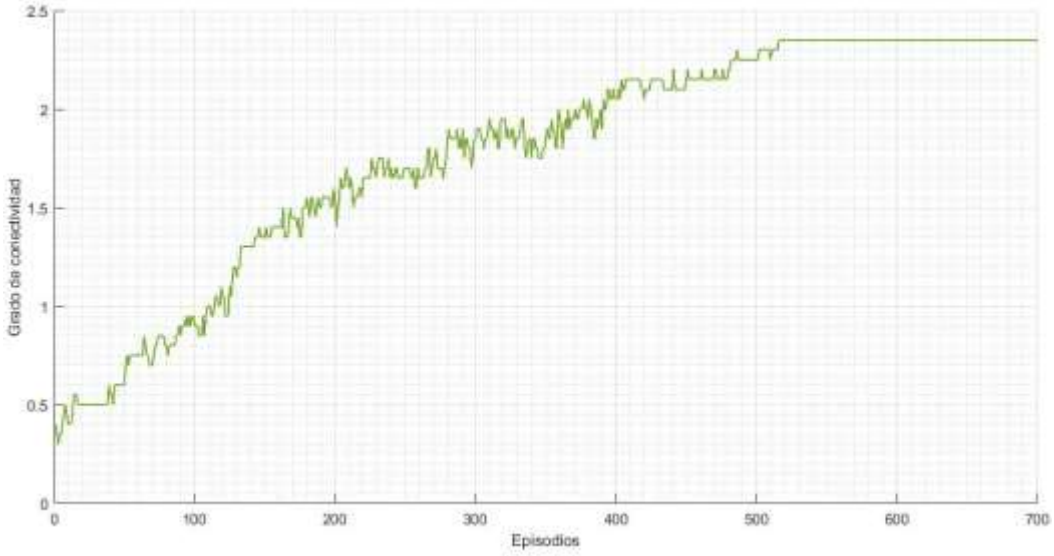


Fig. 10.16 Promedio del grado de conectividad a través del proceso de aprendizaje.

10.2.3. Experimento 5: Primera solución con un enemigo

Este es el experimento de la primera solución modificada para incluir un enemigo del enjambre, al cual los agentes deberán aprender a evitar. Para este experimento se utilizan 20 robots ubicados de forma aleatoria y un enemigo que navega aleatoriamente en el espacio del enjambre. Este experimento se desarrolló en 1000

episodios. En la figura 10.17 se muestra la secuencia de los movimientos de los robots. Los robots del enjambre se representan con puntos azules, y el robot enemigo se representa con un punto de color rojo.

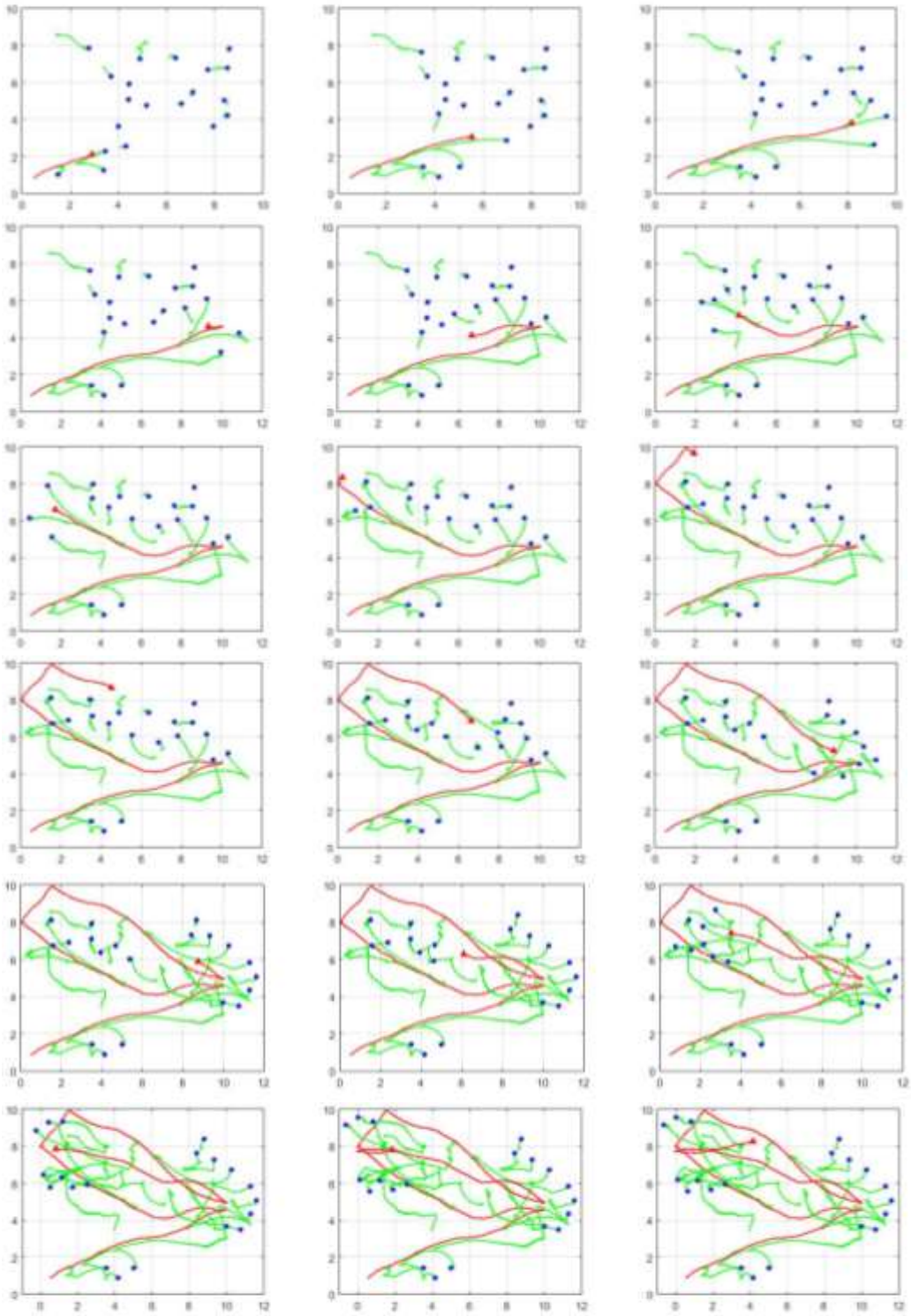


Fig. 10.17 Secuencia experimento primera solución con enemigo.

En la figura 10.18 se muestra el grado de conectividad de todos los robots por cada epoca durante todo el experimento. El promedio de conectividad de los robot durante las 1000 epocas se muestra en la figura 10.19.

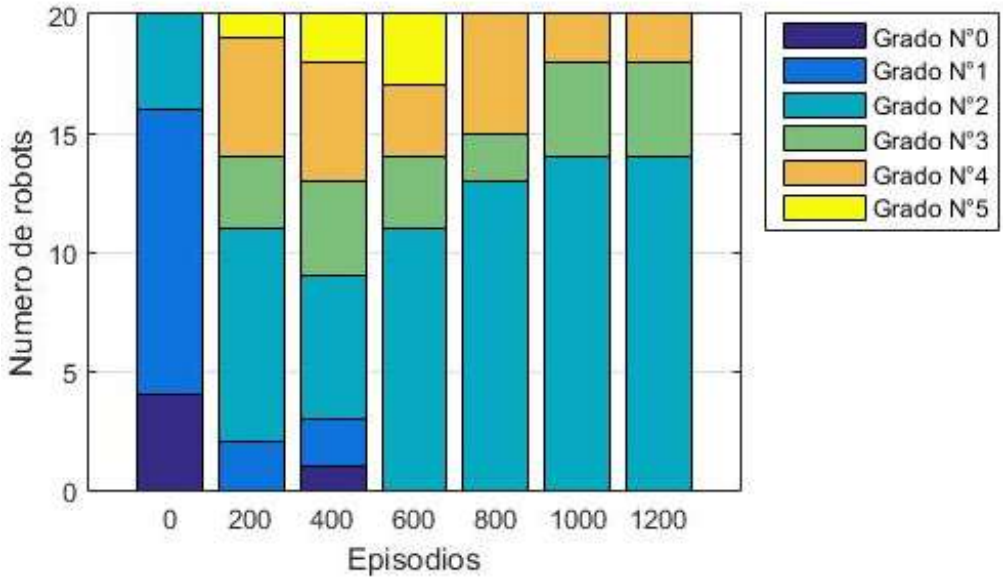


Fig. 10.18 Grado de conectividad a través del proceso de aprendizaje.

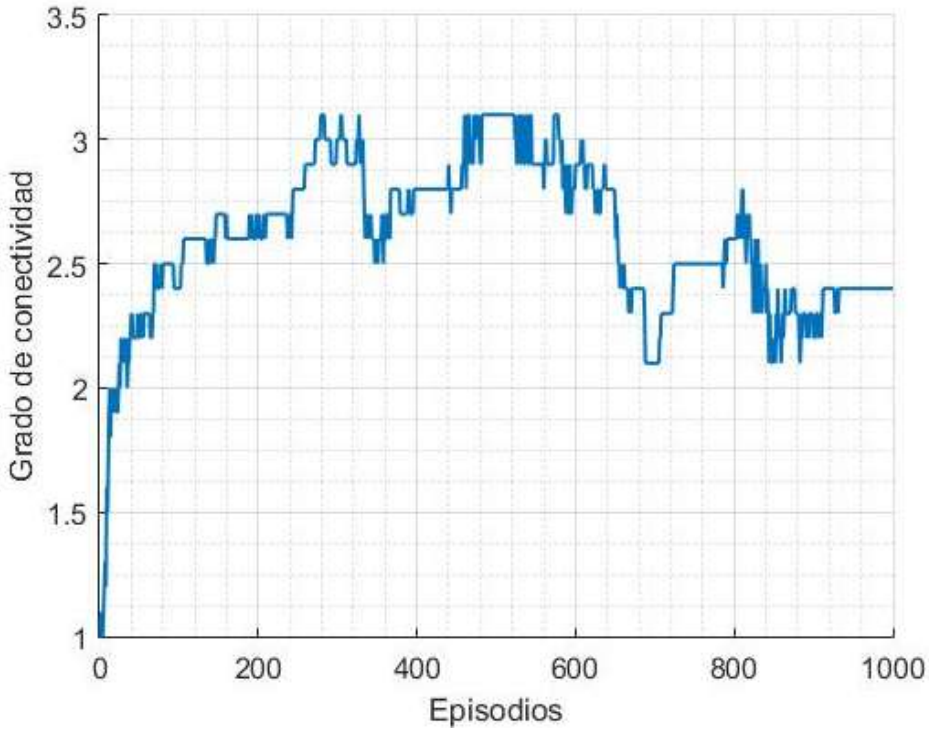


Fig. 10.19 Promedio del grado de conectividad a través del proceso de aprendizaje.

10.3. Pruebas de la segunda solución propuesta en Matlab

En la segunda solución propuesta se definen un radio de atracción o interacción y otro radio de repulsión, como se explicó en el capítulo anterior. Se realizaron 4 experimentos en Matlab, al igual que en las pruebas de la primera solución, donde en los dos primeros experimentos se muestra el comportamiento de los robots variando el número de agentes del enjambre y compartiendo la misma matriz Q, y los dos últimos experimentos se realizaron variando el tamaño del enjambre y asignando de forma individual una matriz Q para cada agente del enjambre.

10.3.1. Experimento con matriz Q compartida

- **Experimento 1 segunda solución.**

  En este experimento se utilizan 10 robots ubicados aleatoriamente. Todos los robots pueden acceder y modificar la misma matriz Q, compartiendo sus propias experiencias con el resto de robots del enjambre. El experimento se desarrolla en 1000 episodios. En la figura 10.20 se muestra la secuencia de movimientos de los robots del enjambre.

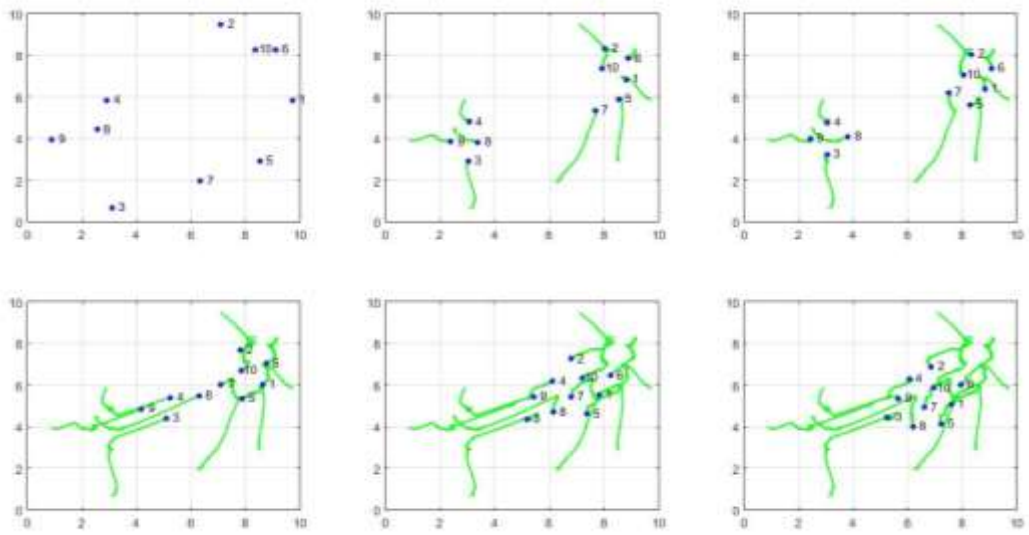


Fig. 10.20 Secuencia experimento 1 segunda solución.

Inicialmente el enjambre se agrupa formando dos clústeres o subgrupos, y conforme pasa el tiempo los dos clústeres formados se unen formando un solo clúster como se puede ver en la secuencia mostrada en la figura 10.20.

En la figura 10.21 se muestra la posición de los robots al final del experimento y la conexión entre los agentes que se formó. La conexión se establece con los vecinos que están a una distancia igual o menor a $1.1 * rr$, que igual o menor al radio de repulsión más el 10%. La interconexión de los agentes se modela mediante un grafo no dirigido, representando la comunicación entre los agentes con una línea roja que une a los robots que tienen comunicación. La topología del grafo que se forma se muestra en la figura 10.21.

En la figura 10.21 se puede ver que al final los robots se agrupan formando un solo clúster, por lo tanto, el grafo que se forma al final es un grafo conexo.

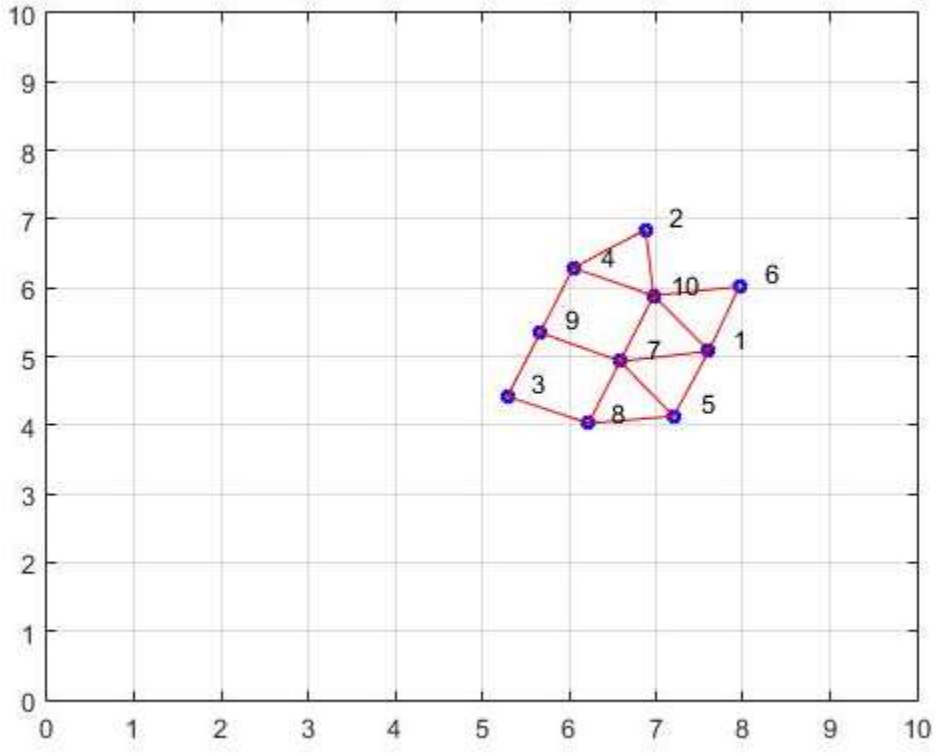


Fig. 10.21 Posición final de los robots.

El grafo que se forma en la figura 10.21 también se puede representar de forma matricial por medio de la matriz de adyacencia A, en la que se representa con 1 los agentes que tiene conexión y 0 los que no tienen conexión. En (10.3.1.1) se muestra la matriz de adyacencia del grafo de la figura 10.21.

Matriz de adyacencia $(\mathbf{10.3.1.1})$

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

La cantidad de conexiones que tiene cada agente se conoce como el grado de conectividad o simplemente grado del vértice y se representa matricialmente con la matriz de grado D, que contiene en su diagonal el grado de conectividad de cada robot. En (10.3.1.2) se muestra la matriz de grado del grafo de la figura 10.21.

Matriz de grado de conectividad $(\mathbf{10.3.1.2})$

$$D = \begin{pmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 \end{pmatrix}$$

Se calcula la matriz laplaciana definida como $L = D - A$, y se calculan los valores propios de L para comprobar matemáticamente que el grafo de la figura 10.21 es un grafo conexo. Los valores propios de L se muestran a continuación:

$$\lambda_1 = 0;\ \lambda_2 = 0.91;\ \lambda_3 = 1.21;\ \lambda_4 = 2.03;\ \lambda_5 = 2.71;$$
$$\lambda_6 = 3.85;\ \lambda_7 = 4.25;\ \lambda_8 = 4.48;\ \lambda_9 = 5.79;\ \lambda_{10} = 6.76$$

Si se cumple que $\lambda_2 > 0$, se dice que el grafo es conexo. Para este caso se tiene que $\lambda_2 = 0.91$; por consiguiente se puede decir que el grafo es conexo.
En la figura 10.22 se muestra la evolución del grado de conectividad a través de los 1000 episodios del experimento. La mayoría de los robots tiende a permanecer en con un grado de tres conexiones durante el experimento.

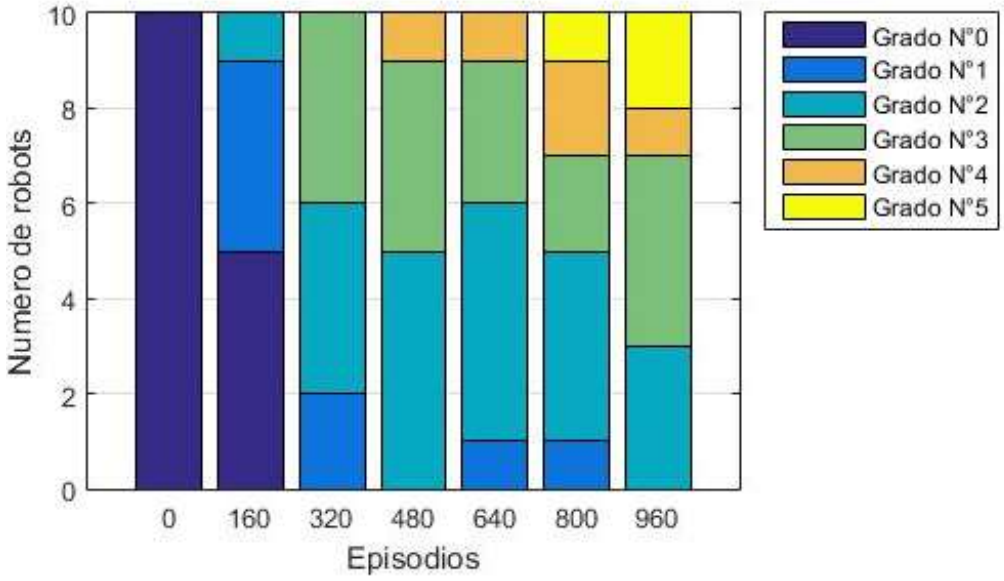


Fig. 10.22 Grado de cada robot a través del proceso de aprendizaje.

Al final del experimento cuatro de los diez robots alcanzaron tres conexiones, tres robots alcanzaron un grado de conectividad de dos, un robot término con cuatro conexiones y dos robots terminaron con cinco conexiones.
En la figura 10.23 se muestra el promedio del grado de conectividad de los todos robots.

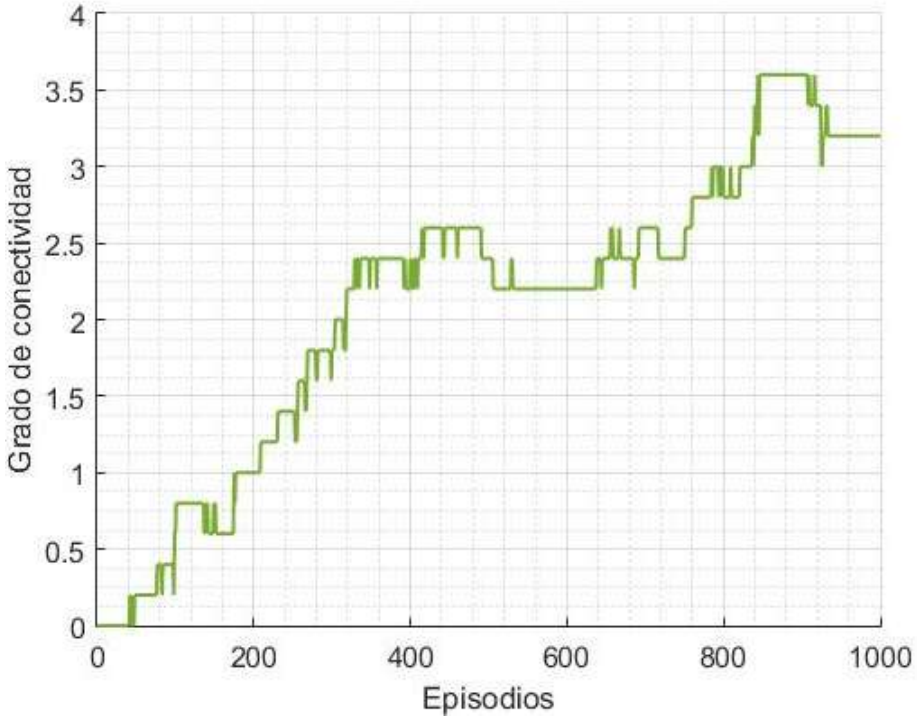


Fig. 10.23 Promedio del grado de conectividad a través del proceso de aprendizaje.

- **Experimento 2 segunda solución.**
  En este experimento se ejecutaron 1000 episodios utilizando 15 robots con la matriz Q compartida. En la figura 10.24 se muestra la secuencia de movimientos durante el experimento.

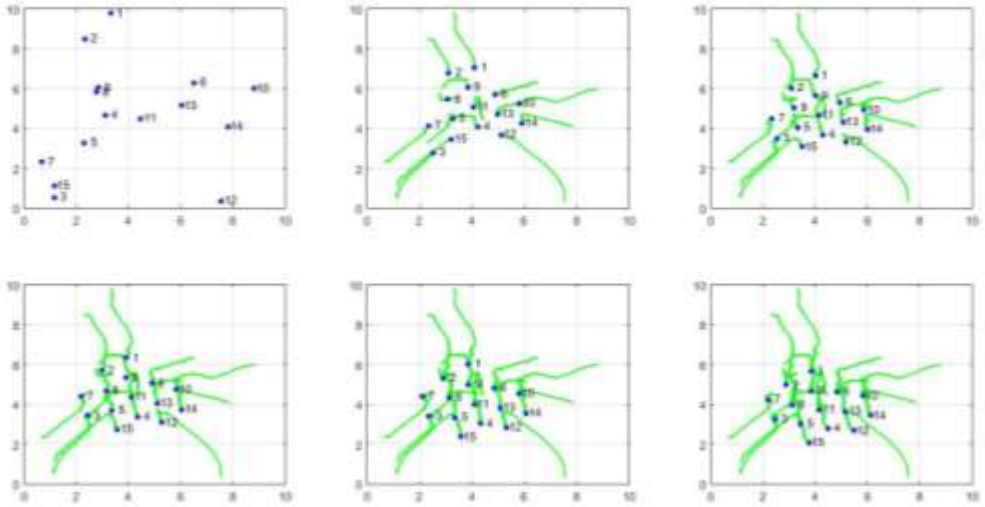


Fig. 10.24 Secuencia experimento 2 segunda solución.

El tiempo transcurrido durante los 1000 episodios le permitió al enjambre aprender a agruparse como se muestra en la figura 10.24. En la figura 10.25 se muestra la topología del grafo que se forma al final del experimento. Al final los robots del enjambre se agrupan formando un solo clúster. La conexión se establece con los vecinos que están a una distancia igual o menor a $1.1 * rr$.

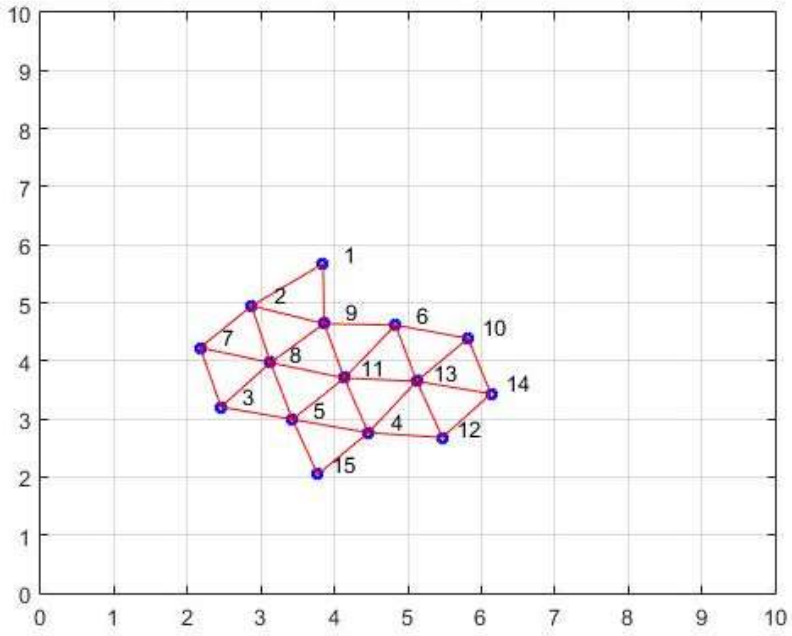


Fig. 10.25 Posición final de los robots.

La matriz de adyacencia del grafo de la figura 10.25 se muestra en la matriz (10.3.1.3).

Matriz de adyacencia $(\mathbf{10.3.1.3})$

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

La matriz de grado de conectividad del grafo que se forma en la figura 10.25 se muestra en la matriz (10.3.3.4)

Matriz de grado de conectividad $(\mathbf{10.3.1.4})$

$$D = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix}$$

La matriz laplaciona L se calcula haciendo la diferencia entre la matriz de grado D, y la matriz de adyacencia A, $L = D - A$. Al calcular los eigenvalores de la matriz laplaciana del grafo de la figura 10.25 tenemos:

$$\lambda_1 = 0;\ \lambda_2 = 0.65;\ \lambda_3 = 1.17;\ \lambda_4 = 1.69;\ \lambda_5 = 2.40;$$
$$\lambda_6 = 2.96;\ \lambda_7 = 3.40;\ \lambda_8 = 4.18;\ \lambda_9 = 4.68;\ \lambda_{10} = 4.83;$$
$$\lambda_{11} = 5.76;\ \lambda_{12} = 6.16;\ \lambda_{13} = 6.82;\ \lambda_{14} = 7.49;\ \lambda_{15} = 7.79$$

Puesto que $\lambda_2 > 0$, se comprueba matemáticamente que el grafo de la figura 10.25 es un grafo conexo. En la figura 10.26 se muestra el grado de conectividad de todos los agentes durante el proceso de aprendizaje.

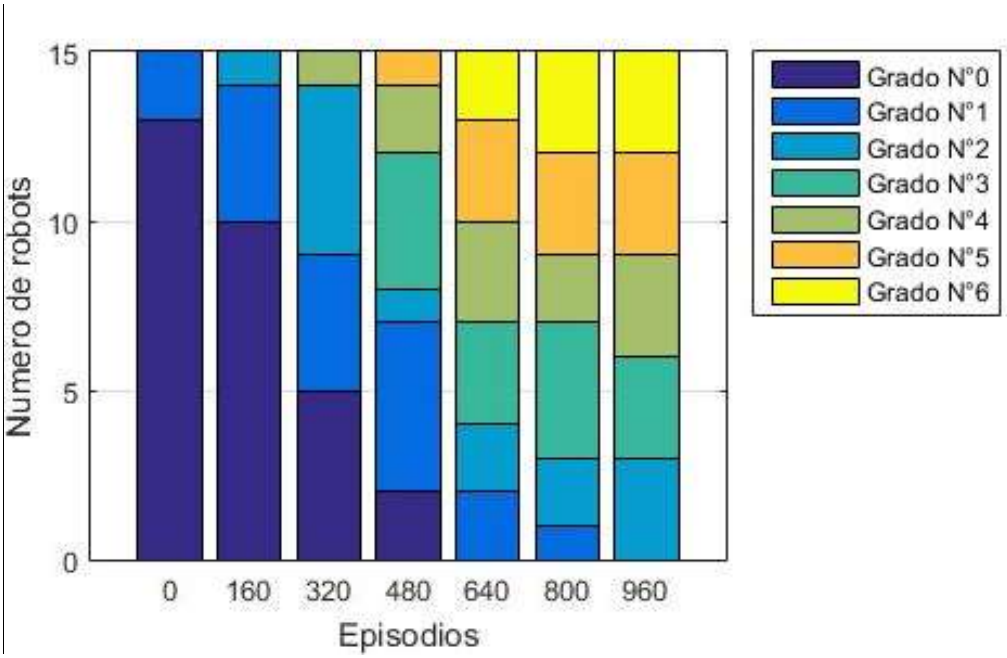


Fig. 10.26 Grado de cada robot a través del proceso de aprendizaje.

En la figura 10.27 se muestra el promedio de conexiones del enjambre durante los 1000 episodios transcurridos del experimento.

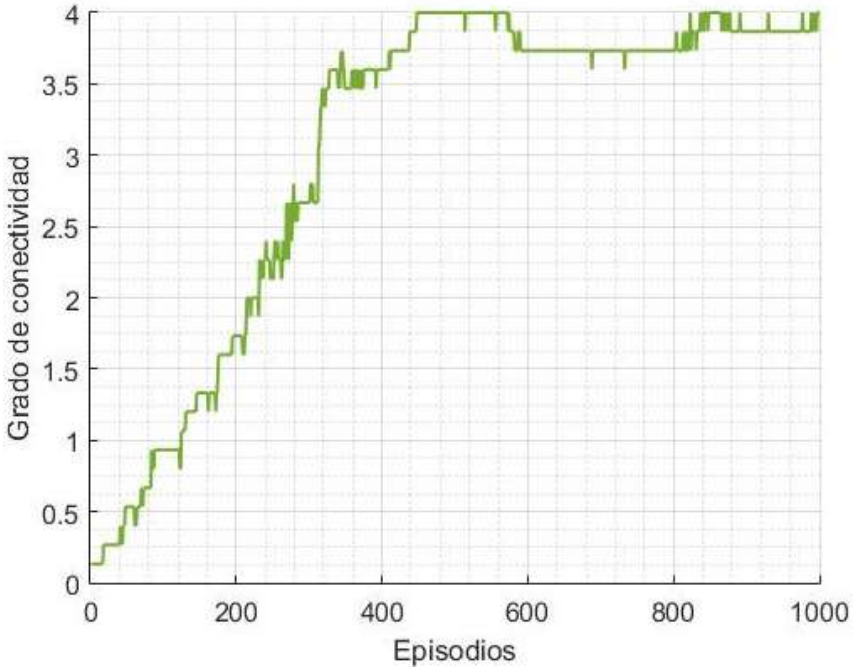


Fig. 10.27 Promedio del grado de conectividad a través del proceso de aprendizaje.

10.3.2. Experimento con matriz Q individual

- **Experimento 3 segunda solución.**
  Para este experimento se utilizaron 20 robots ejecutando la simulando durante 2000 episodios. A cada robot se asignada una matriz Q individual, de tal forma que ningún robot pueda acceder o modificar la matriz Q asignada a otro robot. En la figura 10.28 se muestra la secuencia de movimientos durante este experimento.

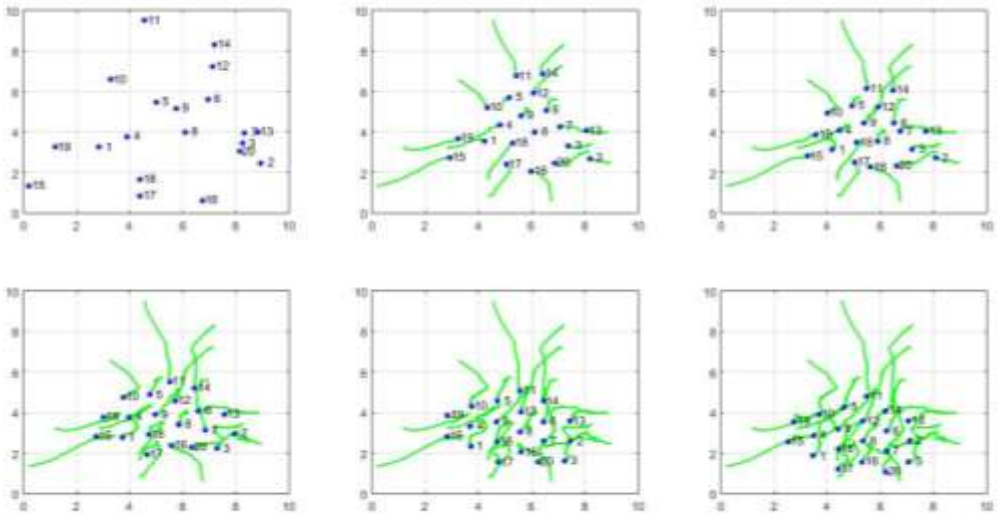


Fig. 10.28 Secuencia experimento 3 segunda solución.

En la figura 10.29 se muestra la posición de todos los robots y la topología del grafo que se formó al final del experimento. La conexión se establece con los vecinos que están a una distancia igual o menor a $1.1 * rr$. Los robots terminaron agrupados formando un único clúster como se puede ver en la figura 10.29.

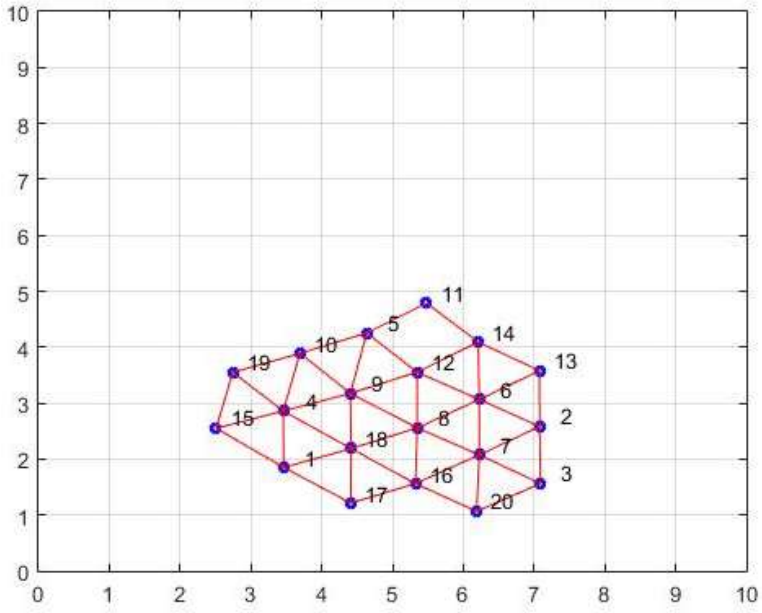


Fig. 10.29 Posición final de los robots.

La matriz de adyacencia del grafo de la figura 10.29 se muestra en se muestra en la matriz (10.3.2.1).

Matriz de adyacencia $(\mathbf{10.3.2.1})$

$$A = \begin{pmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{pmatrix}$$

$$D = \begin{pmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \end{pmatrix}$$

Matriz de grado de conectividad $(\mathbf{10.3.2.2})$

Se calcula la matriz laplaciana haciendo $L = D - A$, con la matriz de adyacencia y la matriz de grado de conectividad. Se calcula los valores propios de L y con el segundo valor propio de la matriz laplaciana se puede comprobar si el grafo es un grafo conexo o no. Para este caso se tiene que $\lambda_2 = 0.59 > 0$, por lo que el grafo es un grafo conexo. En la figura 10.30 se muestra el grado de conectividad de todos los robots durante el proceso de aprendizaje. El grado de conectividad de cada agente tiende a estabilizarse a partir del episodio 1800.

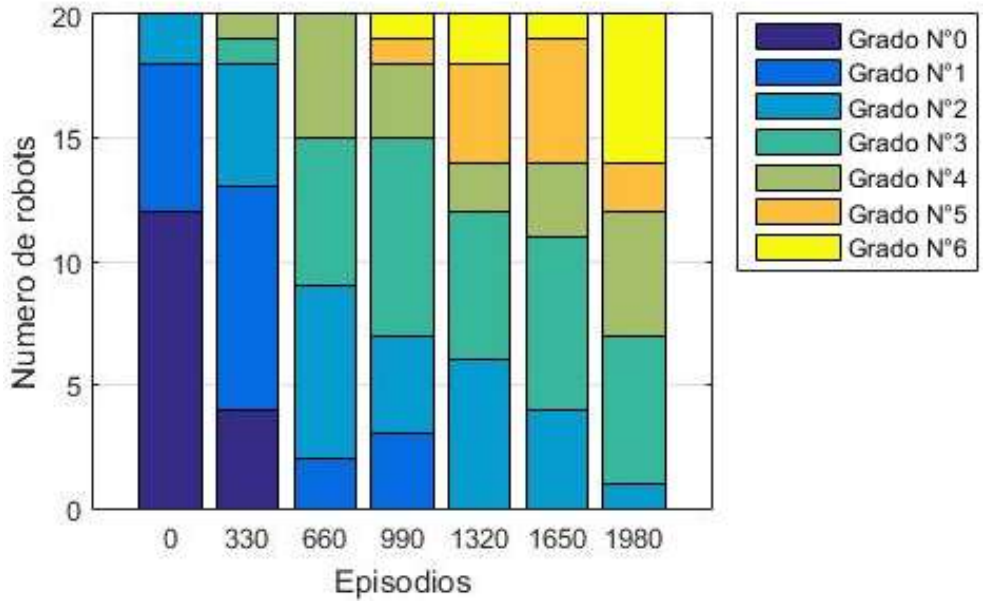


Fig. 10.30 Grado de cada robot a través del proceso de aprendizaje.

En la figura 10.31 se muestra el promedio del grado de conectividad de todos los agentes. Después del episodio 1800, los cambios registrados del promedio del grado de conectividad tienden convergen al valor de 4.4, como se muestra en 10.31.

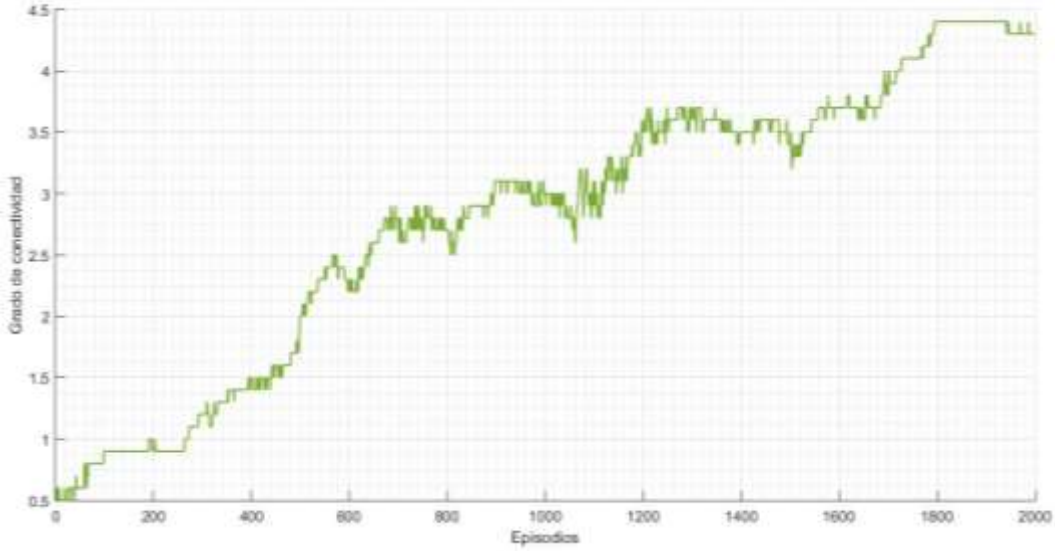


Fig. 10.31 Promedio del grado de conectividad a través del proceso de aprendizaje.

- **Experimento 4 segunda solución.**

  Este experimento se ejecuta en un total de 5000 episodios utilizando 40 robots con matriz Q individual. Para este experimento no se muestra la matriz A de adyacencia ni la matriz de grado D, puesto que las dimensiones de las matrices son de 40x40 y superan el tamaño para incluirlas en este documento. En la figura 10.32 se muestra la secuencia completa de movimientos durante el experimento.

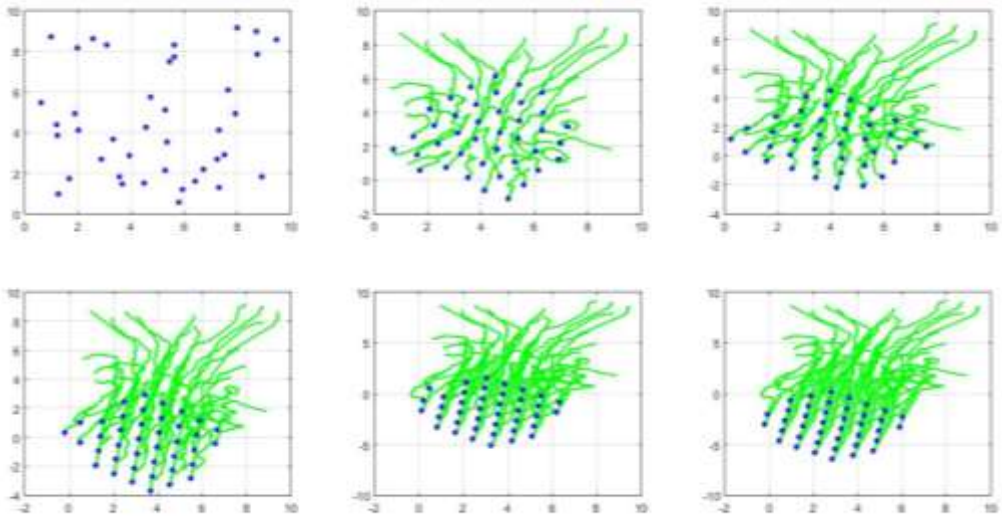


Fig. 10.32 Secuencia experimento 3 segunda solución.

En la figura 10.33 se muestra la posición de los robots al final de la simulación modelando la comunicación como un grafo no dirigido. La conexión se establece con los vecinos que están a una distancia igual o menor a $1.1 * rr$.

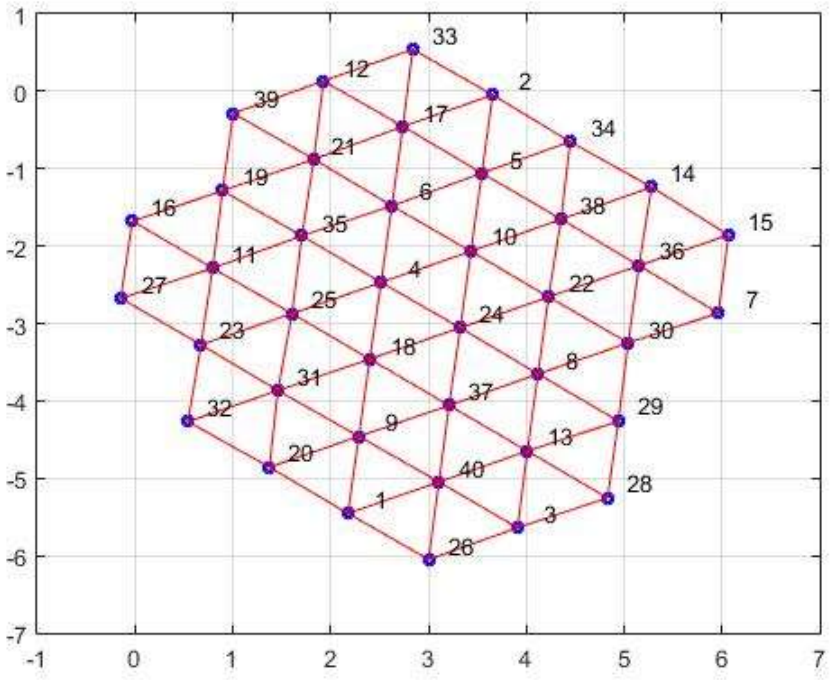


Fig. 10.33 Posición final de los robots.

Los valores propios de la matriz lapaciana del grafo de la figura 10.33 se muestran a continuación:

$$\lambda_1 = 0;\ \lambda_2 = 0.38;\ \lambda_3 = 0.44;\ \lambda_4 = 0.86;\ \lambda_5 = 1.18;\ \lambda_6 = 1.60;\ \lambda_7 = 1.75;$$
$$\lambda_8 = 1.94;\ \lambda_9 = 2.42;\ \lambda_{10} = 2.76;\ \lambda_{11} = 3.08;\ \lambda_{12} = 3.26;\ \lambda_{13} = 3.61;$$
$$\lambda_{14} = 3.7;\ \lambda_{15} = 3.81;\ \lambda_{16} = 4.09;\ \lambda_{17} = 4.41;\ \lambda_{18} = 4.60;\ \lambda_{19} = 4.91;$$
$$\lambda_{20} = 5.03;\ \lambda_{21} = 5.2;\ \lambda_{22} = 5.29;\ \lambda_{23} = 5.47;\ \lambda_{24} = 5.75;\ \lambda_{25} = 5.99;$$
$$\lambda_{26} = 6.15;\ \lambda_{27} = 6.57;\ \lambda_{28} = 6.64;\ \lambda_{29} = 6.92;\ \lambda_{30} = 6.98;\ \lambda_{31} = 7.26;$$
$$\lambda_{32} = 7.45;\ \lambda_{33} = 7.61;\ \lambda_{34} = 7.71;\ \lambda_{35} = 7.94;\ \lambda_{36} = 7.98;\ \lambda_{37} = 8.06;$$
$$\lambda_{38} = 8.08;\ \lambda_{39} = 8.55;\ \lambda_{40} = 8.56$$

Como $\lambda_2 > 0$, se demuestra que el grafo de la figura 10.33 es un grafo conexo.
En la figura 10.34 se muestra la evolución del grado de conectividad de todos los robots durante el proceso de aprendizaje. El grado de conectividad de cada agente tiende a estabilizarse a partir del episodio 4332. Al final solo nueve robots terminaron con tres conexiones, ocho robots alcanzaron un grado de conectividad de cuatro, tres robots finalizaron con cinco conexiones y veinte robots alcanzaron un grado de conectividad de 6 conexiones.
En la figura 10.35 se muestra el promedio del grado de conectividad de todos los agentes. Después del episodio 4332, no se registraron cambios del promedio del grado

de conectividad y el valor final tiende a converger al valor de 4.55, como se muestra en 10.35.

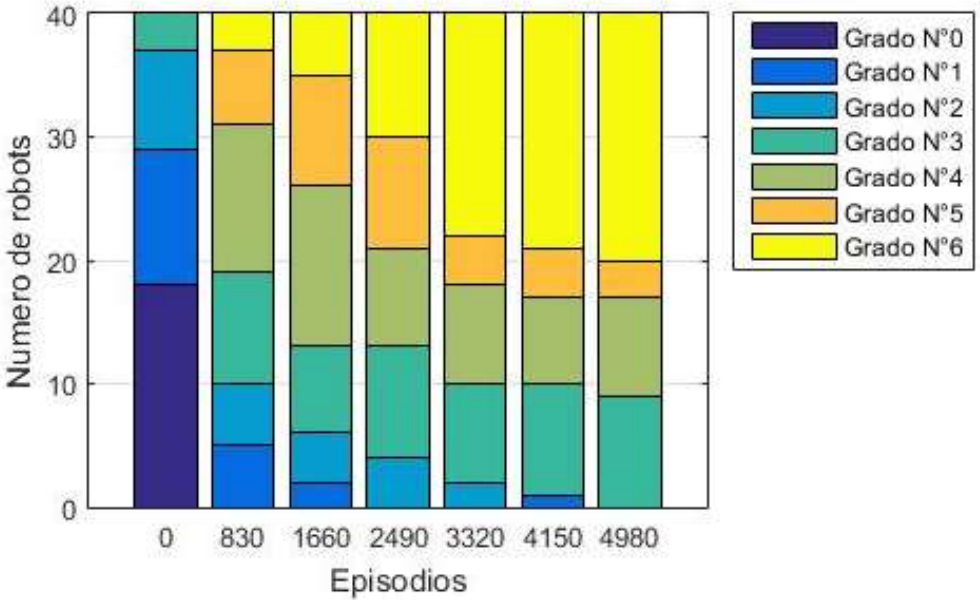


Fig. 10.34 Grado de cada robot a través del proceso de aprendizaje.

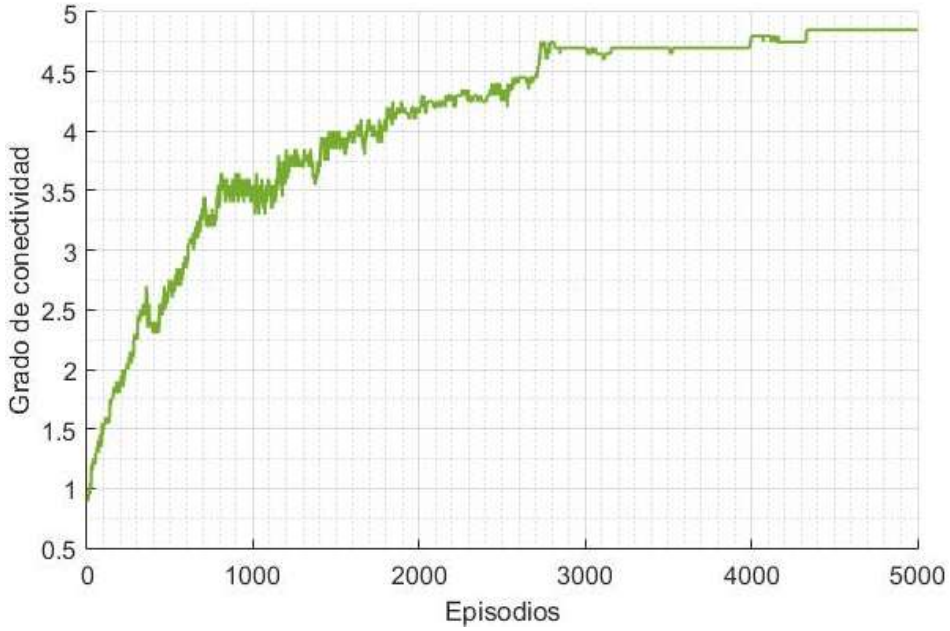


Fig. 10.35 Promedio del grado de conectividad a través del proceso de aprendizaje.

10.3.3. Experimento 5: Segunda solución con un enemigo

En esta parte se mostrara el experimento con la modificación de la segunda solución para incluir un enemigo del enjambre. Los agentes deberán aprender a agruparse mientras evitan al enemigo. Para este experimento se utilizan 20 robots ubicados de forma aleatoria y un enemigo que navega aleatoriamente en el espacio del enjambre.

Este experimento se desarrolló durante 2000 episodios. En la figura 10.36 se muestra la secuencia de los movimientos de los robots. Los robots del enjambre se representan con puntos azules, y el robot enemigo se representa con un punto de color rojo.

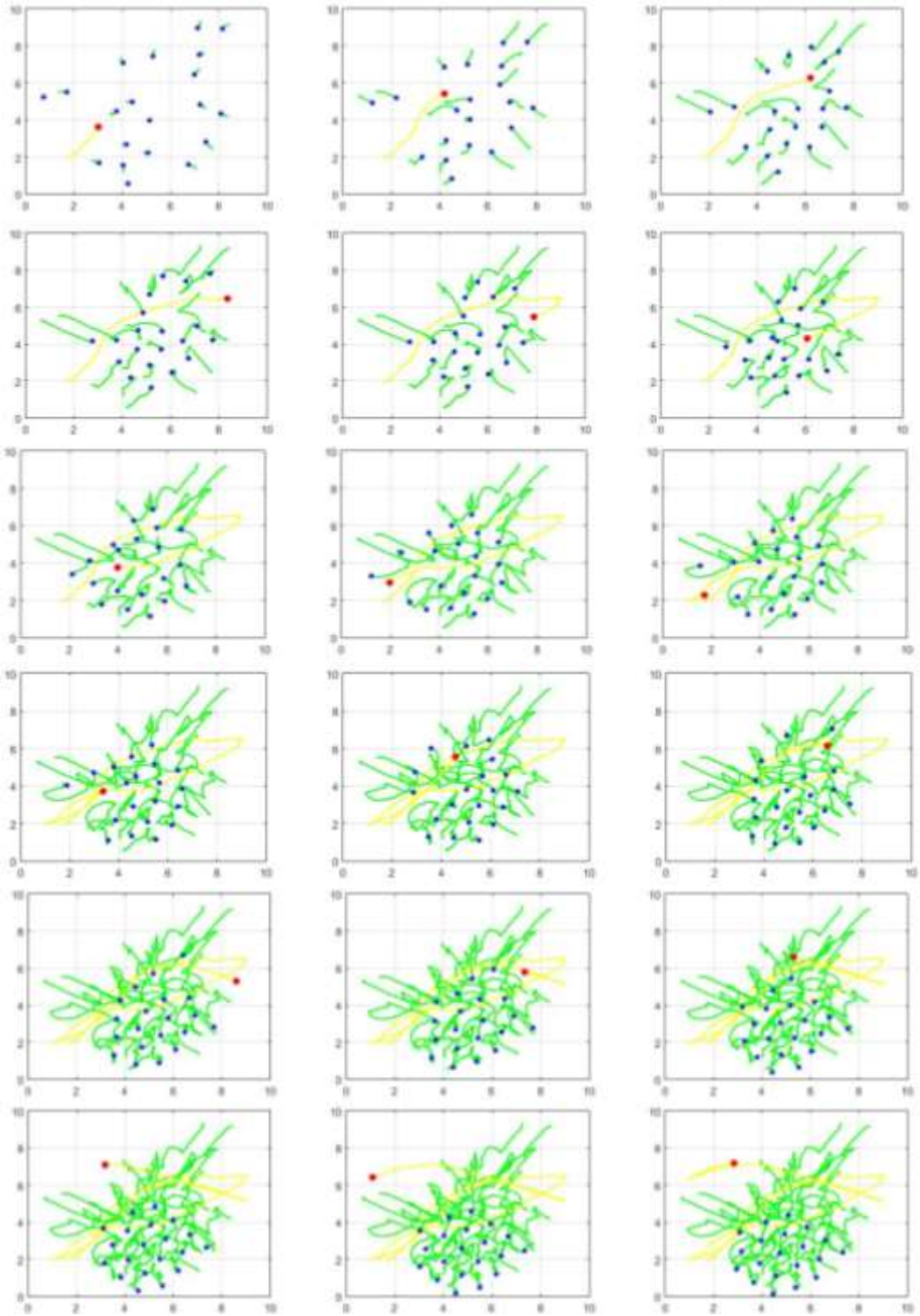


Fig. 10.36 Secuencia experimento segunda solución con enemigo.

En la figura 10.37 se muestra el grado de conectividad de todos los robots por cada epoca durante todo el experimento. El promedio de conectividad de los robot durante las 2000 epocas del experimento se muestra en la figura 10.38.

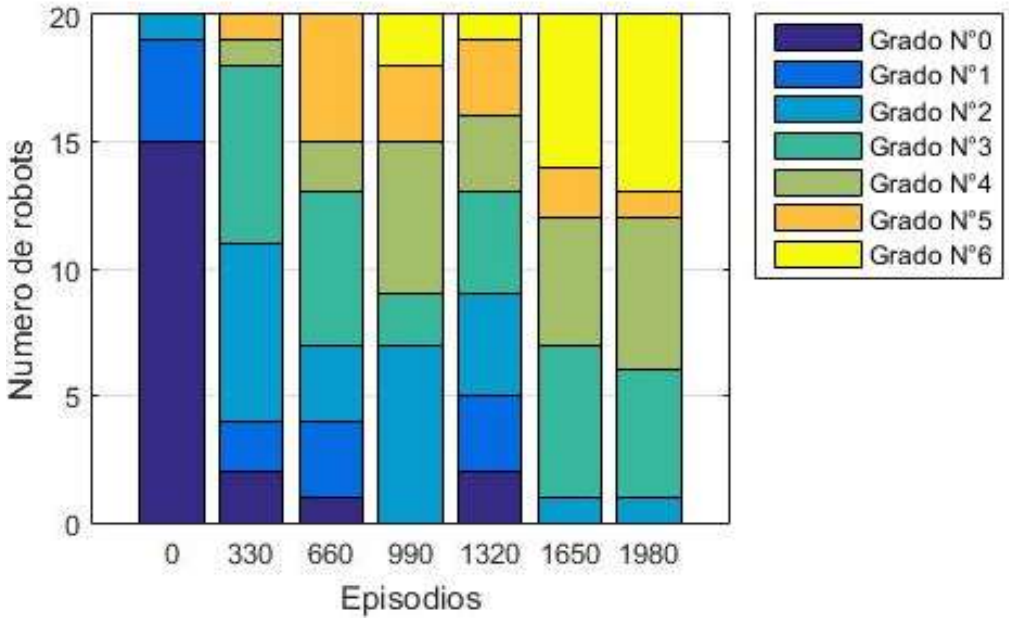


Fig. 10.37 Grado de cada robot a través del proceso de aprendizaje.

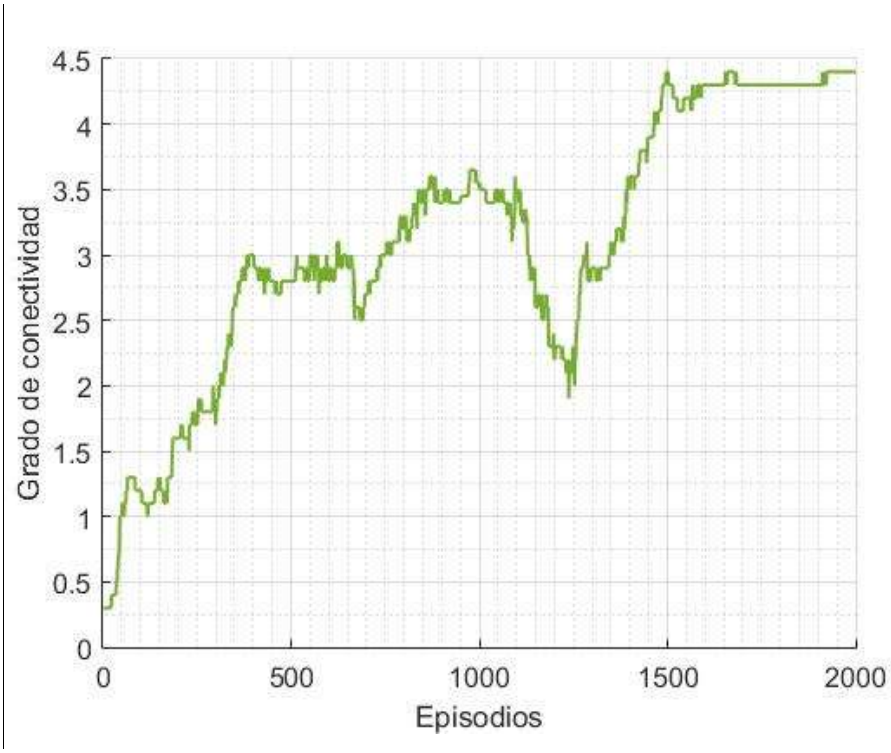


Fig. 10.38 Promedio del grado de conectividad a través del proceso de aprendizaje.

10.4. Prueba en V-Rep

El segundo sistema de aprendizaje por refuerzo propuesto se evalúa utilizando Matlab, para realizar los cálculos matemáticos, sincronizado con V-Rep edición educativa, para la simulación del enjambre con robots virtuales. En este experimento se desarrolla en un total de 1000 episodios y se utilizan 18 robots iguales ubicados aleatoriamente. Los robots utilizados en este experimento son el modelo virtual de un cuadricóptero. El modelo del cuadricóptero se muestra en la figura 10.39. Cada robot virtual simulado en V-Rep se controla desde Matlab, cambiando la posición del target.

Fig. 10.39 Robot cuadricóptero virtual

En la figura 10.40 se muestra una secuencia de imágenes con la posición de los robots a lo largo del experimento. Los robots navegan mientras aprenden a agruparse y permanecer cerca de sus vecinos sin chocar o colisionar con otros robots del enjambre.

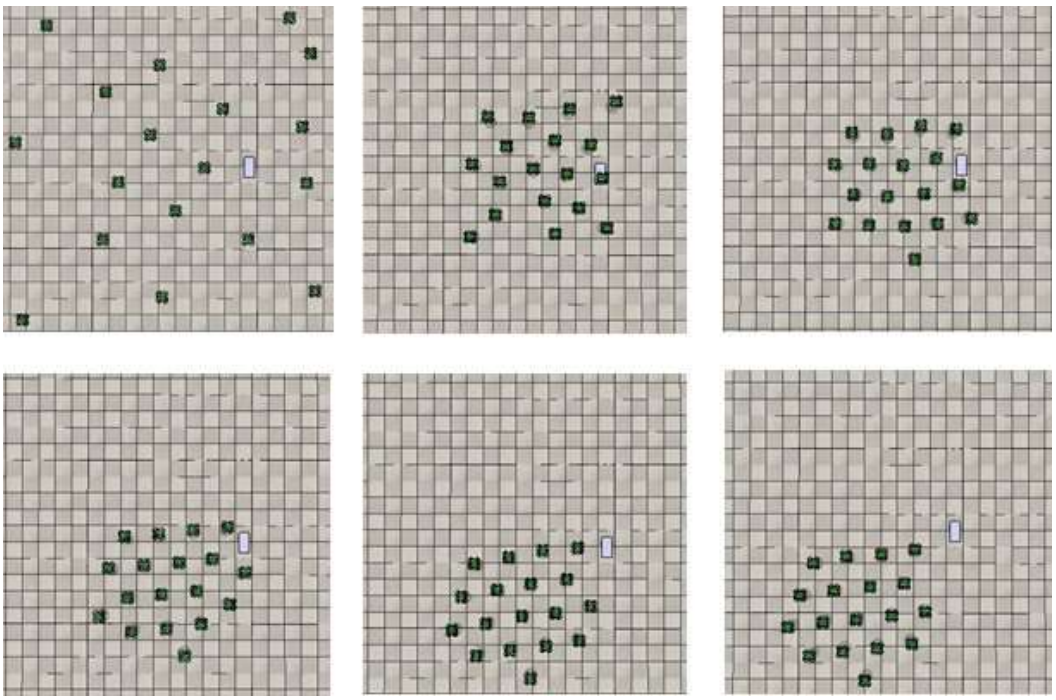


Fig. 10.40 Secuencia experimento V-Rep.

En la figura 10.41 se muestra la posición final de los robots enumerados y con sus respectivas conexiones. La figura 10.41 se construyó en Matlab tomando la posición de los robots desde V-Rep.

A partir del grafo de la figura 10.41 se construye la matriz de adyacencia o conectividad, representando con 1 a los robots que tiene conexión y con 0 a los robots que no tienen conexión. La matriz de adyacencia se muestra en la matriz (10.4.1).
La cantidad de conexiones o grado de conectividad de cada robot también se representa de forma matricial con la matriz de grado D. La matriz de grado se muestra en la matriz (10.4.2)

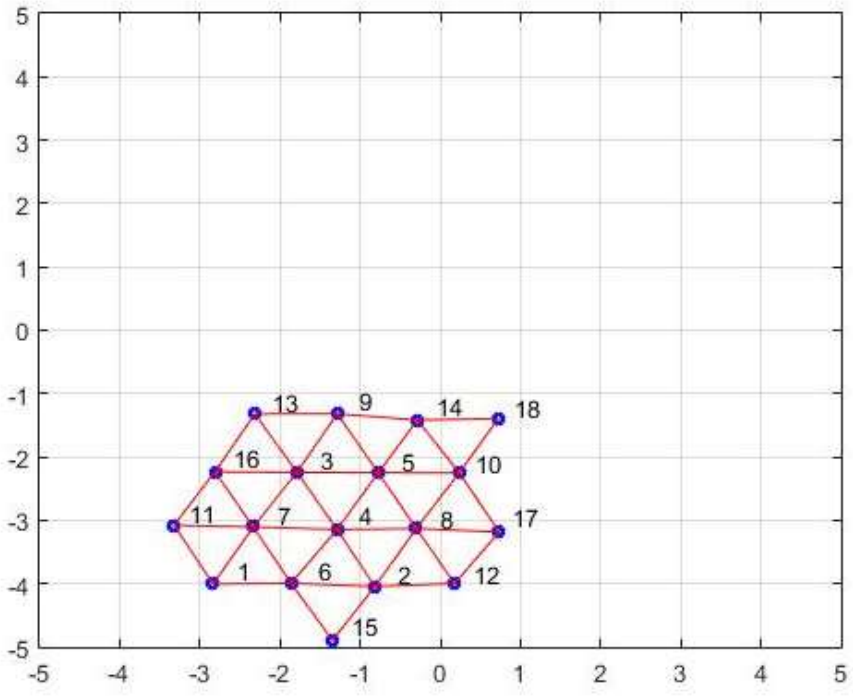


Fig. 10.41 Posición final de los robots.

$$A = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Matriz de adyacencia $(\mathbf{10.4.1})$

Se calcula la matriz laplaciana, $L = D - A$, del grafo de la figura 10.41, y se calculan los valores propios de la matriz laplaciana para comprobar matemáticamente que el grafo que se forma en la figura 10.41 es un grafo conexo.

$$D = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix}$$

Matriz de grado de conectividad $(\mathbf{10.4.2})$

Como se cumple que $\lambda_2 > 0$, se comprueba que el grafo de la figura 10.41 es un grafo conexo.

$$\lambda_1 = 0;\ \lambda_2 = 0.7;\ \lambda_3 = 0.95;\ \lambda_4 = 1.59;\ \lambda_5 = 1.72;\ \lambda_6 = 2.88;\ \lambda_7 = 3.37;$$
$$\lambda_8 = 3.58;\ \lambda_9 = 3.76;\ \lambda_{10} = 4.86;\ \lambda_{11} = 4.93;\ \lambda_{12} = 5.33;\ \lambda_{13} = 5.91;$$
$$\lambda_{14} = 6.25;\ \lambda_{15} = 6.95;\ \lambda_{16} = 7.26;\ \lambda_{17} = 7.93;\ \lambda_{18} = 8.01$$

En la figura 10.42 se muestra el grado de conectividad de los robots durante todo el experimento.

En la figura 10.43 se muestra el promedio del grado de conectividad de todos los robots durante el proceso de aprendizaje. En la figura 10.43 se puede ver que después del episodio 400 el promedio del grado tiene a permanecer constante en un valor de 4.22 con pequeñas variaciones. Esto puede interpretarse como que los robots aprendieron a agruparse y permanecer juntos como un enjambre.

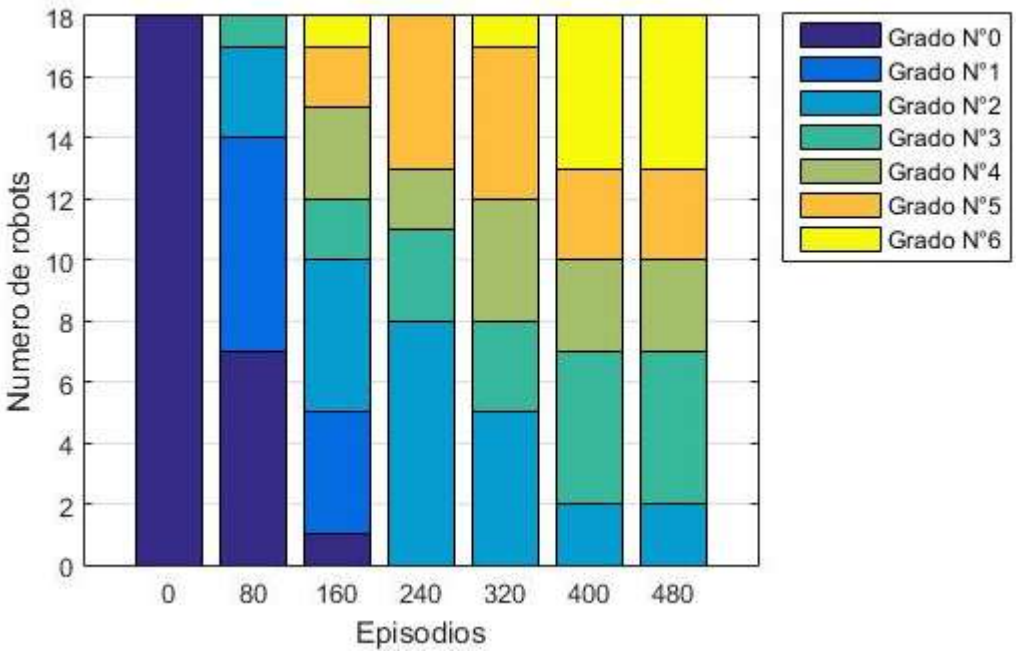


Fig. 10.42 Grado de cada robot a través del proceso de aprendizaje.

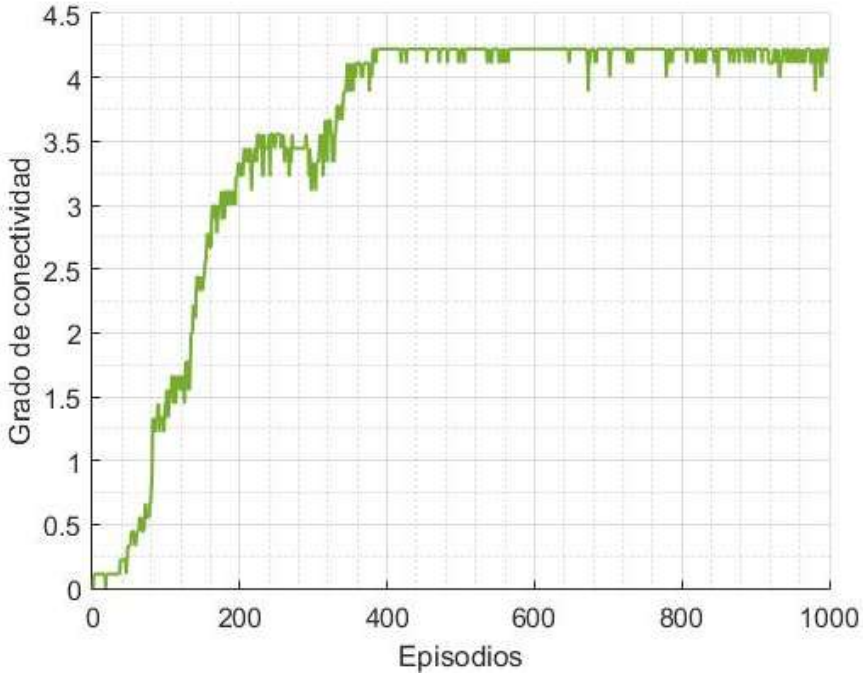


Fig. 10.43 Promedio del grado de conectividad a través del proceso de aprendizaje.

11. CONCLUSIONES Y TRABAJOS FUTUROS

Se presentaron e implementaron en Matlab dos posibles soluciones basadas en técnicas de aprendizaje por refuerzo con diferentes enfoques para cumplir con los objetivos del proyecto y se validaron mediante diferentes pruebas cada una de las propuestas presentadas. Cada solución presentada en este trabajo se modificó para incluir un enemigo del enjambre al que los robots aprendieron a que evitar. En las dos propuestas planteadas los robots no poseen información sobre la ubicación espacial. Para cada propuesta se asumió resuelta la comunicación entre los agentes del grupo. Cada agente se conectó con sus vecinos inmediatos creando la red de comunicación que se formó al final de los experimentos. Esta red de comunicación se modelo como un grafo no dirigido con los enlaces correspondientes a las conexiones formadas entre los agentes. Se ha medido el grado de conectividad de los agentes a través del proceso de aprendizaje y se ha utilizado teoría algebraica de grafos para determinar si la topología del grafo formado es conexo o no. Finalmente se evaluó y probó la segunda solución propuesta con robots virtuales en V-Rep, usando el modelo virtual de un cuadricóptero.

En las pruebas que se realizaron con la matriz Q compartida por todos los robots del enjambre, se evidencio que: el proceso de aprendizaje se desarrolló más rápido, los robots se agruparon y alcanzaron la posición final en un menor tiempo, convergiendo a los valores finales en menos épocas que los experimentos realizados con la matriz Q individual; ya que el compartir la matriz Q le permitió a cada robots compartir sus propias experiencias, vivencias pasadas y la base del conocimiento con los otros agentes del enjambre.

Como se pudo observar en los resultados de la primera solución propuesta, al final de los experimentos los robots se agruparon formando varios sub grupos o sub clústeres creando grafos no conexos. En todos los experimentos realizados de la primera solución, a pesar de incrementar el número de robots en cada experimento, al final el grado de conectividad de los robots tendió a permanecer en un valor de 2 y el promedio del grado de conectividad se estabilizo en un valor entre 2 y 3, cerca de 2.5. Esto se debe a la forma como se definieron los estados, porque para la primera solución los estados de cada robot se forman en función de las distancias de los dos vecinos más cercanos, por lo que los robots aprenden a permanecer cerca solo de dos de sus vecinos, sin buscar a los otros robots del enjambre. También por esta razón al final se forman sub grupos o clústeres separados y se crea un grafo no conexo.

Por otro lado, se observó que en todos los experimentos mostrados de la segunda solución propuesta, los robots se agruparon formando un único clúster, generando así un grafo conexo al final de la simulación. También se pudo observar en el primer experimento de la segunda solución, mostrado en la secuencia de la figura 10.20, que aunque al comienzo del experimento los robots se agruparon formando dos clústeres, cuando los agentes de uno de los clústeres detectan dentro de su radio de interacción a los agentes del otro clúster, terminan convergiendo y formando un único clúster. No obstante, el grupo que tiende a moverse al otro clúster es el que tiene menos agentes, es decir, el grupo con mayor cantidad de robots tiende a arrastrar al clúster con menos agentes. De igual forma se pudo observar de las pruebas de la segunda solución, que a medida que se aumentaron el número de robots del enjambre, al final la mayor parte de agentes tendían a alcanzar un grado de conectividad de 6, y el promedio del grado de conectividad tendía a incrementar

conforme se incrementó el tamaño del enjambre; sin embargo, como el máximo grado de conectividad que puede alcanzar un agente del grupo es de 6, el promedio del grado de conectividad normalmente será menor que 6.

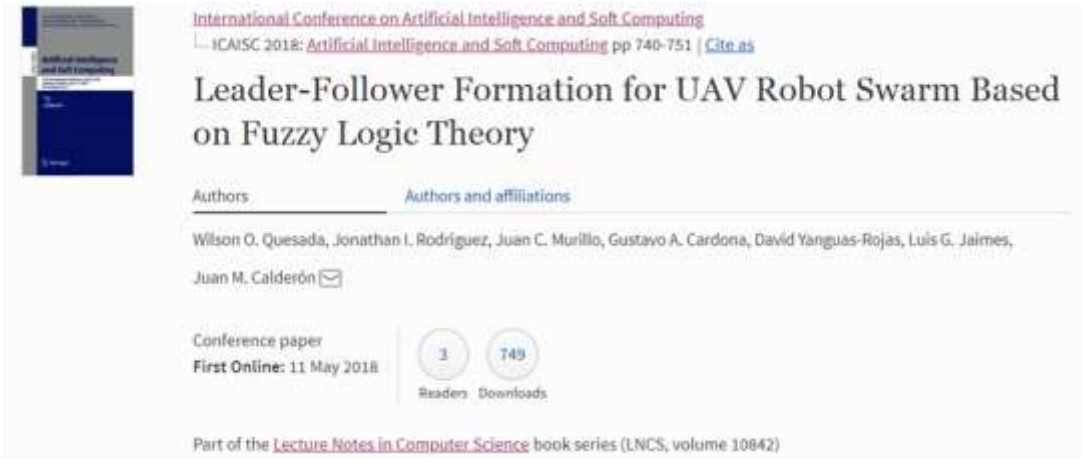
Durante la ejecución de las pruebas de la segunda solución se pudo observar que algunos robots tardaban más que otros en unirse al grupo, y en algunas ocasiones algunos agentes parecían permanecer quietos por un tiempo. Esto es debido a la forma en que se definieron los estados y como se ejecutan las acciones de la segunda solución. Ya que para cada robot se establecen los estados dividiendo su espacio en cuatro cuadrantes de forma local, y el robot calcula el vector de movimiento con la cantidad de agentes vecinos dentro del radio de atracción o repulsión, respectivamente, en la dirección del cuadrante de la acción seleccionada, por lo que si en el cuadrante de la acción seleccionada no hay vecinos, a pesar de ejecutar la acción, el movimiento resultante será igual a cero.

Durante la etapa del desarrollo práctico en unas de las pruebas de la simulación del algoritmo de la segunda solución que no fueron registrados en este documento, se pudo observar en algunas ocasiones que un robot iniciaba en una posición demasiado retirada del resto del grupo, y sin tener otros robots dentro de su radio de interacción. En estos casos los otros robots del enjambre se agruparon pero el robot que inicio demasiado aislado de sus compañeros permaneció quieto durante todo el experimento, ya que este robot no podía ver a ninguno de sus compañeros.

Para trabajos futuros se planea incluir un algoritmo de navegación con evasión de obstáculos, como árboles o paredes, e incluir tareas más complejas para el enjambre como la búsqueda y detección de victimas en lugares de desastres.

12. PUBLICACIONES Y PRODUCTOS RELACIONADOS

Este trabajo de grado genero un artículo titulado “Leader-Follower Formation for UAV Robot Swarm Based on Fuzzy Logic Theory”.



International Conference on Artificial Intelligence and Soft Computing

ICAISC 2018: Artificial Intelligence and Soft Computing pp 740-751 | Cite as

Leader-Follower Formation for UAV Robot Swarm Based on Fuzzy Logic Theory

Authors | Authors and affiliations

Wilson O. Quesada, Jonathan I. Rodriguez, Juan C. Murillo, Gustavo A. Cardona, David Yanguas-Rojas, Luis G. Jaimes, Juan M. Calderón

Conference paper
First Online: 11 May 2018

3 Readers | 749 Downloads

Part of the Lecture Notes in Computer Science book series (LNCS, volume 10842)

Fig. 12.1 Artículo presentado en ICAISC [33].

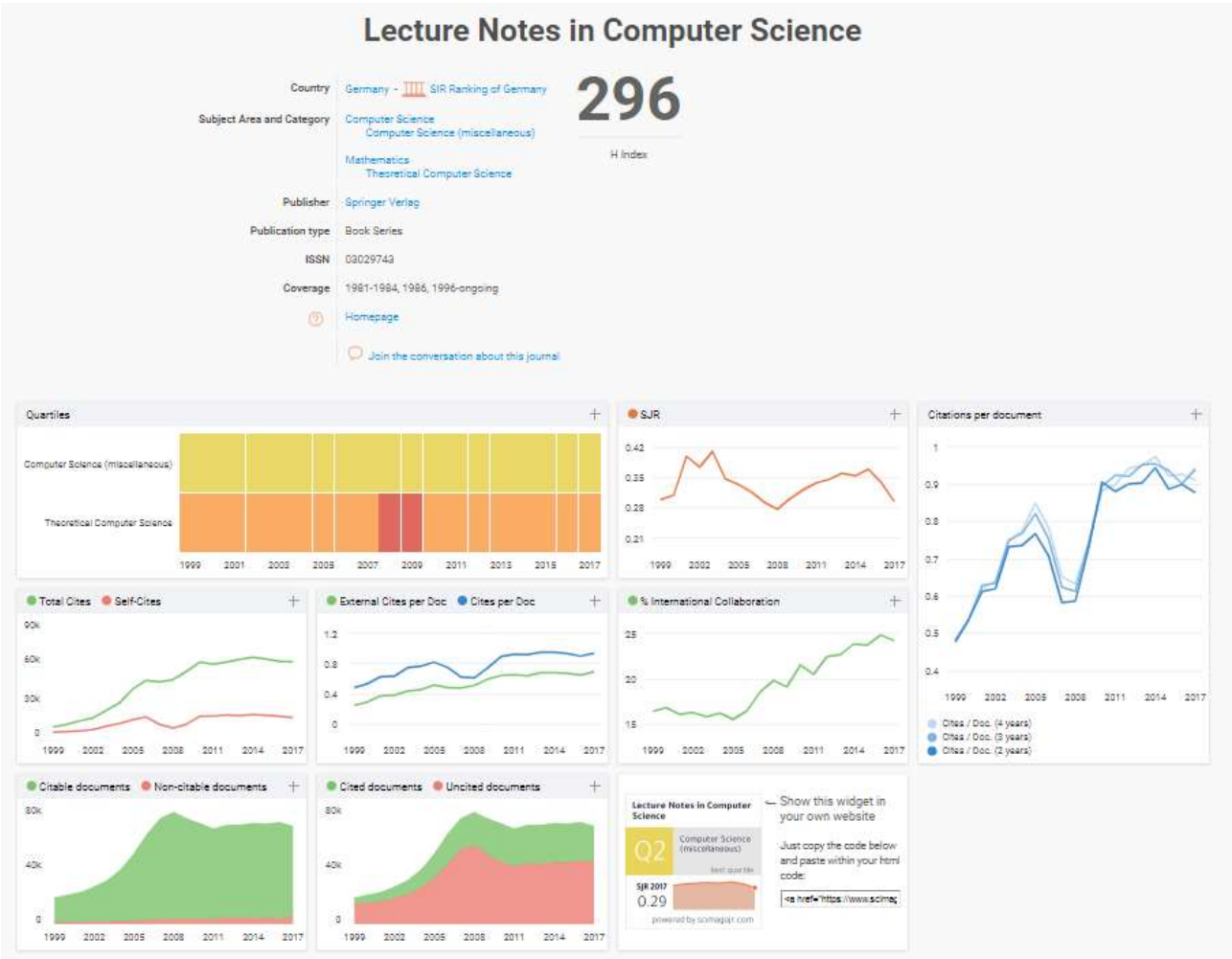


Fig. 12.2 Clasificación según portal SCIMAGO JOURNAL [34].

Este artículo se presentó en la conferencia “International Conference on Artificial Intelligence and Soft Computing ICAISC 2018” el día 11 de Mayo de 2018. Tal como se observa en la figura 12.1 [33].

Las memorias de este evento fueron publicadas en “Lecture Notes in Computer Science” de la editorial Springer Nature. Este tipo de publicación seriada se encuentra clasificada como Q2 por el portal “SCIMAGO JOURNAL”, además posee un SJR = 0.29 y un H Index = 296 tal como se observa en la figura 12.2 [34].

13. BIBLIOGRAFÍA

[1] Centre for research on the epidemiology of disasters CRED, “The human cost of natural disasters: A global perspective”, 2015.

[2] Yoon, H., Shiftehfar, R., Cho, S., Spencer, B. F., Nelson, M. E., & Agha, G. A. (2016). “Victim Localization and Assessment System for Emergency Responders”. Journal of Computing in Civil Engineering, 30(2), [04015011].

[3] Rodrıguez, Saith, Eyberth Rojas, Katherın Pérez, Carlos Quintero, Oswaldo Pena, Andrés Reyes, and Juan Calderón. "STOx’s 2013 Team Description Paper." (2013).

[4] Quintero, Carlos, Saith Rodríguez, Katherín Pérez, Jorge López, Eyberth Rojas, and Juan Calderón. "Learning soccer drills for the small size league of robocup." In Robot Soccer World Cup, pp. 395-406. Springer, Cham, 2014.

[5] Rodríguez, Saith, Eyberth Rojas, Katherín Pérez, Jorge López, Carlos Quintero, and Juan Calderón. "Fast path planning algorithm for the robocup small size league." In Robot Soccer World Cup, pp. 407-418. Springer, Cham, 2014.

[6] Rodrıguez, Saith, Eyberth Rojas, Katherın Pérez, Jorge López, Carlos Quintero, Juan Manuel Calderón, and Oswaldo Pena. "STOx’s 2015 Extended Team Description Paper." Joao Pessoa, Brazil (2014).

[7] Cardona, Gustavo A., Wilfrido Moreno, Alfredo Weitzenfeld, and Juan M. Calderon. "Reduction of impact force in falling robots using variable stiffness." In SoutheastCon, 2016, pp. 1-6. IEEE, 2016.

[8] Elibol, Ercan, Juan Calderon, Martin Llofriu, Carlos Quintero, Wilfrido Moreno, and Alfredo Weitzenfeld. "Power usage reduction of humanoid standing process using q-learning." In Robot Soccer World Cup, pp. 251-263. Springer, Cham, 2015.

[9] Calderón, Juan M., Wilfrido Moreno, and Alfredo Weitzenfeld. "Fuzzy variable stiffness in landing phase for jumping robot." In Innovations in bio-inspired computing and applications, pp. 511-522. Springer, Cham, 2016.

[10] Elibol, Ercan, Juan Calderon, Martin Llofriu, Wilfrido Moreno, and Alfredo Weitzenfeld. "Analyzing and Reducing Energy Usage in a Humanoid Robot During Standing Up and Sitting Down Tasks." International Journal of Humanoid Robotics 13, no. 04 (2016): 1650014.

[11] Calderon, Juan, Gustavo A. Cardona, Martin Llofriu, Muhaimen Shamsi, Fallon Williams, Wilfrido Moreno, and Alfredo Weitzenfeld. "Impact Force Reduction Using Variable Stiffness with an Optimal Approach for Falling Robots." In Robot World Cup, pp. 404-415. Springer, Cham, 2016.

[12] Calderon, Juan M., Eyberth R. Rojas, Saith Rodriguez, Heyson R. Baez, and Jorge A. Lopez. "A Robot soccer team as a strategy to develop educational iniciatives." In Latin American and Caribbean Conference for Engineering and Technology, Panama City, Panama. 2012.

[13] Baez, Heyson, Katherin Perez, Eyberth Rojas, Saith Rodriguez, Jorge Lopez, Carlos Quintero, and Juan Manuel Calderon. "Application of an educational strategy based on a soccer robotic platform." In Advanced Robotics (ICAR), 2013 16th International Conference on, pp. 1-6. IEEE, 2013.

[14] M.J. Mataric, “Reinforcement Learning in the Multi-Robot Domain”, Autonomous Robots 4, 73–83 (1997)

[15] A Roadmap for US Robotics: From Internet to Robotics. 2016 Edition. Disponible en internet. URL: http://jacobsschool.ucsd.edu/contextualrobotics/docs/rm3-final-rs.pdf

[16] Y. Zennir, “Apprentissage par renforcement et système distribués: application a l'apprentissage de la marche d'un robot hexapode”, Ph.D Thesis, Institut National Des Sciences Appliquées De Lyon, 2004. Disponible en internet. URL: http://theses.insa-lyon.fr/publication/2004ISAL0034/these.pdf

[17] J. León, “Simulación De Enjambres De Robots En Labores De Exploración Para Detección De Posibles Víctimas”, Tesis de maestría en ingeniería electrónica, Universidad Santo Tomás Bogotá DC, 2017.

[18] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, “Swarm robotics: a review from the swarm engineering perspective”, Swarm Intelligence, vol. 7, no. 1, pp. 1–41, 2013.

[19] León, Jose, Gustavo A. Cardona, Andres Botello, and Juan M. Calderón. "Robot swarms theory applicable to seek and rescue operation." In International Conference on Intelligent Systems Design and Applications, pp. 1061-1070. Springer, Cham, 2016.

[20] León, José, Gustavo A. Cardona, Luis G. Jaimes, Juan M. Calderón, and Pablo Ospina Rodriguez. "Rendezvous Consensus Algorithm Applied to the Location of Possible Victims in Disaster Zones." In International Conference on Artificial Intelligence and Soft Computing, pp. 700-710. Springer, Cham, 2018.

[21] Yanguas-Rojas, David, Gustavo A. Cardona, Juan Ramirez-Rugeles, and Eduardo Mojica-Nava. "Victims search, identification, and evacuation with heterogeneous robot networks for search and rescue." In Automatic Control (CCAC), 2017 IEEE 3rd Colombian Conference on, pp. 1-6. IEEE, 2017.

[22] S. Zhiguo, T. Jun, Z. Qiao, Z. Xiaomeng, W. Junming, "The Improved Q-Learning Algorithm based on Pheromone Mechanism for Swarm Robot System", IEEE 32nd Chinese Control Conference (CCC), pp. 6033-6038, 2013.

[23] A. Šošić, A.M. Zoubir, H. Koeppl, “Reinforcement learning in a continuum of agents”, Swarm Intelligence, Vol. 12. no. 1, pp 23–51, 2018.

[24] W.O. Quesada, J.I. Rodríguez, J.C. Murillo, G.A. Cardona, D.Y. Rojas, L.G. Jaimes, J.M. Calderón, “Leader-Follower Formation for UAV Robot Swarm Based on Fuzzy Logic Theory”, Artificial Intelligence and Soft Computing. ICAISC 2018. Lecture Notes in Computer Science, vol 10842. Springer, Cham

[25] P.J. Denning, “Computer Science: The Discipline”, 1999. Disponible en internet. URL: http://denninginstitute.com/pjd/PUBS/ENC/cs99.pdf

[26] F.S. Caparrini, “Introducción al aprendizaje automático”, Articulo, Dpto. de Ciencias de la Computación e Inteligencia Artificial, Universidad de Sevilla. Tomado de medio virtual en Mayo del 2018 desde http://www.cs.us.es/~fsancho/?e=75

[27] R.S. Sutton, A.G. Barto, “Reinforcement Learning: An Introduction”, Near-final draft, May 27, 2018. Disponible en internet. URL: http://incompleteideas.net/book/the-book-2nd.html

[28] DL4J, A Beginner’s Guide to Deep Reinforcement Learning, Tomado de medio virtual en Mayo del 2018 desde: https://deeplearning4j.org/deepreinforcementlearning

[29] Analytics Vidhya, Simple Beginner’s guide to Reinforcement Learning & its implementation, Tomado de medio virtual en Mayo del 2018 desde: https://www.analyticsvidhya.com/blog/2017/01/introduction-to-reinforcement-learning-implementation/

[30] Intel AI, Guest Post (Part I): Demystifying Deep Reinforcement Learning, Tomado de medio virtual en Mayo del 2018 desde: https://ai.intel.com/demystifying-deep-reinforcement-learning/

[31] G. Beni, “From Swarm Intelligence to Swarm Robotics”, International Workshop on Swarm Robotics, SR 2004: Swarm Robotics pp 1-9.

[32] F.L. Lewis, “Cooperative Control of Multi-Agent Systems - Optimal and Adaptive Design Approaches”, Communications and Control Engineering.

[33] https://link.springer.com/chapter/10.1007/978-3-319-91262-2_65

[34] https://www.scimagojr.com/journalsearch.php?q=25674&tip=sid&clean=0