UNIVERSIDAD SANTO TOMÁS

Formación Adaptativa De Sistemas De Enjambres De Drones Por Medio De La Técnica De Pastoreo

---

Realizado por

Edson Farid Macias Cifuentes

Proyecto de Investigación presentado como requisito para optar el título de:
Magíster en Ingeniería Electrónica

Grupo de Investigación GED (Grupo de Estudio y Desarrollo en Robótica)
Facultad de Ingeniería Electrónica
División de Ingenierías

Junio de 2024

Formación Adaptativa De Sistemas De Enjambres De Drones Por Medio De La Técnica De Pastoreo

Realizado por

Edson Farid Macias Cifuentes

Proyecto de Grado presentado en cumplimiento del requisito
para optar por el grado de Ingeniería Electrónica

Dirigido por

Ph.D David Alejandro Martínez Vásquez

Co-Dirigido por

Ph.D Juan Manuel Calderón Chávez

MSc. Sindy Paola Amaya

Grupo de Investigación GED (Grupo de Estudio y Desarrollo en Robótica)
Facultad de Ingeniería Electrónica
División de Ingenierías

Junio de 2024

Autoridades de la Universidad

RECTOR GENERAL
R.P. FRAY ÁLVARO JOSÉ ARANGO RESTREPO, O.P.

VICERRECTOR ADMINISTRATIVO Y FINANCIERO GENERAL
R.P. FRAY HERNÁN YESID RIVERA ROBERTO, O.P

VICERRECTOR ACADÉMICO GENERAL
R.P. FRAY MAURICIO ANTONIO CORTÉS GALLEGO, O.P.

SECRETARIO GENERAL
Dra. INGRID LORENA CAMPOS VARGAS

DECANO DIVISIÓN DE INGENIERÍAS
R.P. FRAY JAVIER ANTONIO HINCAPIÉ ARDILA, O.P.

SECRETARIA DE DIVISIÓN DE INGENIERÍAS
E.C. LUZ PATRICIA ROCHA CAICEDO

DECANO FACULTAD DE INGENIERÍA ELECTRÓNICA
Ph.D CARLOS ANDRÉS TORRES PINZÓN

Nota de Aceptación

_____
_____
_____
_____
_____

_____

Firma del autor

_____

Firma del jurado

_____

Firma del jurado

BOGOTÁ D.C. —— DE 2024

Advertencia

La Universidad Santo Tomás no se hace responsable de las opiniones y conceptos expresados en el trabajo de grado, solo velará por qué no se publique nada contrario al dogma ni a la moral católica y porque el trabajo no tenga ataques personales y únicamente se vea el anhelo de buscar la verdad científica.

Capítulo III –Art. 46 del Reglamento de la Universidad Santo Tomás.

Agradecimientos

Quiero expresar mi más sincero agradecimiento a todas las personas y organizaciones que han contribuido al desarrollo y finalización de este proyecto. En primer lugar, agradezco al Ejército Nacional por brindarme los recursos y el tiempo necesario para estudiar. Su apoyo ha sido fundamental para que pudiera dedicarme a este proyecto con la dedicación y el esfuerzo requeridos. Agradezco a mis tutores David, Juan Manuel Y Sindy por su orientación, paciencia y apoyo constante durante todo el proceso. Sus conocimientos y experiencia han sido fundamentales para la concepción y realización de este trabajo. Extiendo mi gratitud a mis compañeros y amigos por su colaboración y ánimo incondicional, que me han motivado a seguir adelante en momentos difíciles. A Sandra Por su comprensión y apoyo a lo largo de todo este camino. Sin su respaldo, este logro no habría sido posible. Finalmente, agradezco a la Universidad Santo Tomás y al Departamento de Ingeniería Electrónica por brindarme la oportunidad y los recursos necesarios para llevar a cabo este proyecto. Gracias a todos por su apoyo y contribuciones, que han hecho posible la culminación exitosa de este trabajo.

Edson Macias Cifuentes

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Glosario

- **Agentes:** Son los miembros que hacen parte de un enjambre y que tienen la capacidad de comunicarse entre si para lograr una tarea especifica. [1]
- **Aprendizaje por refuerzo:** Es una forma de entrenar agentes a través de la obtención de premio/recompensa con el fin de mejorar sus habilidades de acuerdo a la experiencia acumulada.
- **Aprendizaje por Refuerzo Multiagente (MARL):** Técnica de aprendizaje automático donde múltiples agentes autónomos aprenden a través de sus propias interacciones en un entorno compartido, con el objetivo de encontrar soluciones óptimas que permitan alcanzar objetivos compartidos o competitivos.
- **Boids:** Modelo computacional de comportamiento grupal de animales como bandadas de aves y bancos de peces, utilizado para simular movimientos colectivos de agentes autónomos siguiendo reglas simples de separación, alineación y cohesión.
- **Drones:** Vehículos aéreos no tripulados controlados de manera remota, utilizados en diversas aplicaciones tanto militares como civiles, incluyendo la detección de minas, patrullaje y formación de enjambres.
- **Enjambre:** Es un termino usado por la robótica de enjambres para referirse a un grupo de robots autónomos que cooperan entre si para un mismo fin, inspirado en el comportamiento social animal como el de las abejas, hormigas aves, peces, entre otros. A los enjambres se le atribuyen tareas de navegación, comunicación y toma de decisiones colectivas. [2]

- **Maquinas de estados Finitos Probabilísticas:** Un tipo especial de máquinas de estados donde las transiciones se realizan de acuerdo a la probabilidad de ocurrencia de un evento. [2]
- **Metaheuristicas:** es una estrategia usada para dar solución a problemas complejos basada en un conocimiento previo que permite hacer exploraciones de forma óptima y definir que acción tomar para dar una solución al problema. Una de estas técnicas metaheuristicas utilizan algoritmos de optimización bio-inspirados, es decir que emplean métodos usados por la naturaleza para resolver problemas. [3]
- **Morfogénesis:** Se refiere a la estructura que puede adoptar un enjambre de robots volviéndose una unidad resistente y segura como resultado de adaptación al medio con el que interactúan.[3]
- **Multiagente:** que adopta una inteligencia colectiva que le permite idear mecanismos de auto-organización y adaptación a un entorno. [3]
- **Navegación:** consiste en la ruta que debe explorar un enjambre para obtener información de un entorno, para esto los agentes del enjambre interactúan entre si y definen una estrategia que les permita mantener una distancia entre ellos.[4]
- **Optimización:** Proceso de encontrar la mejor solución posible para un problema dado, maximizando o minimizando una función objetivo sujeta a ciertas restricciones.
- **Pastoreo:** Es una técnica novedosa aplicada en la robótica de enjambres para evitar que estos colisionen con algún obstáculo, bio-inspirada en la labor que realizan los perros de pastoreo para mover las ovejas.[1]
- **Q-learning:** Algoritmo de aprendizaje por refuerzo que busca aprender la política óptima de un agente mediante la maximización de una función de recompensa acumulada a través de la interacción con el entorno.
- **Simulación:** Uso de entornos computacionales para modelar y evaluar el comportamiento de sistemas complejos, en este caso, para probar y validar algoritmos de enjambres de drones.
- **UxV :** Concepto que describe a sistemas remotamente tripulados, al refererirse al conjunto de sistemas aéreos, terrestres y navegantes.

Capítulo 2

Resumen

Este proyecto se basa en el diseño y la implementación de sistemas adaptativos de enjambres de drones utilizando técnicas de pastoreo. El objetivo principal es desarrollar métodos eficientes para controlar y coordinar múltiples drones, que puedan evadir obstáculos y que puedan ser guiados a un área de cobertura por otro grupo de agentes denominados pastores. Se utilizan algoritmos como Boids y técnicas de aprendizaje por refuerzo, incluyendo Q-learning y Aprendizaje por Refuerzo Multiagente (MARL). Del mismo modo se modificaron y se hicieron mejoras a los algoritmos con el fin de obtener mejores resultados. Las simulaciones se realizan en entornos como CoppeliaSIM, validando experimentalmente la eficacia de las técnicas propuestas. Los resultados demuestran mejoras significativas en la coordinación y eficiencia de los enjambres de drones comparados con metodologías tradicionales.

Palabras clave: enjambres de drones, pastoreo, Boids, Q-learning, MARL, CoppeliaSIM, coordinación, aprendizaje por refuerzo

Capítulo 3

Abstract

This project is based on the design and implementation of adaptive swarm systems of drones using herding techniques. The main objective is to develop efficient methods to control and coordinate multiple drones that can avoid obstacles and be guided to a coverage area by another group of agents called shepherds. Algorithms such as Boids and reinforcement learning techniques, including Q-learning and Multi-Agent Reinforcement Learning (MARL), are used. Similarly, modifications and improvements were made to the algorithms to achieve better results. Simulations are conducted in environments like CoppeliaSIM, experimentally validating the effectiveness of the proposed techniques. The results demonstrate significant improvements in the coordination and efficiency of drone swarms compared to traditional methodologies.

Keywords: drone swarms, herding, Boids, Q-learning, MARL, CoppeliaSIM, coordination, reinforcement learning

Capítulo 4

Introducción

4.1. Planteamiento del Problema

Los sistemas remotamente tripulados, entre estos los drones; son una tecnología que desde sus orígenes han sido empleados en usos militares, los cuales han iniciado su uso desde principios de 1900's [5]. A nivel mundial muchas de estas aplicaciones centran sus esfuerzos en el uso de estas tecnologías para el apoyo de unidades militares, dada su versatilidad y el riesgo que minimizan a las vidas humanas dentro de ambientes hostiles; un ejemplo de esto es el uso de sistemas remotamente tripulados para la detección de minas, detección y acción contra el terrorismo, patrullaje de unidades militares y uno de los más recientes ejemplos es el uso de modelos de formación y control de múltiples drones de manera simultánea [2], muy similar a las formaciones en enjambres de hormigas; de aquí deriva el concepto de lo que hoy se conoce como "enjambre de drones" lo cual hace referencia a la agrupación de dos o más drones para un mismo fin.

Cabe aclarar que, la tecnología actual de sistemas de enjambres de drones permite que estos mantengan una cohesión y estructura al dar cumplimiento a lo que se denomina, "Leyes de Reynolds" y la emulación de comportamientos de especies homogéneas, sin embargo en algunas ocasiones es necesario modificar la estructura del enjambre debido a necesidades del terreno, por ejemplo suponiendo que sea necesario explorar varias veces un mismo sitio lo cual sería repetitivo en la exploración del área de trabajo e ineficiente en términos energéticos.

Por otro lado, la estructura y formación de un enjambre de drones ayuda a mejorar la eficiencia en búsquedas dentro de zonas de desastres, y en este mismo orden de ideas, dentro del ámbito militar podría ayudar a mejorar la seguridad de sus bases. No obstante, si los terrenos o lugares de exploración cambian debido a factores naturales, o a factores propios de un desastre o emergencia; una formación de enjambre de drones podría dejar de ser eficiente, lo que le obligaría a adoptar una forma diferente, que de acuerdo a su número de agentes logre una mayor eficiencia, pero esto demandaría tiempo y gasto energético, por lo que lograr formación adaptativa al área donde se desempeña es un reto que aún se encuentra en estudio y que se ha convertido en una meta por alcanzar a través de la emulación del comportamiento de enjambres, del mismo modo como lo han hecho las especies en su teoría evolutiva.

Para tal efecto, se ha planteado la técnica de morfogénesis, la cuál consiste en utilizar un sistema de enjambres con un nivel jerárquico superior, que adapte su forma a las condiciones del terreno y dentro de este enjambre exista otro sub-enjambre que se encargue de la recolección y procesamiento de la información, lo que supone un reto de ingeniería y se trataría de un sistema heterogéneo bio-inspirado en el comportamiento que ejercen los perros de pastoreo con otros semovientes. En cuanto al pastoreo, consiste en una técnica donde los drones asumen el papel de los perros pastores sobre semovientes reales, sin embargo dentro de la comunidad científica la información que apunte a que un sistema de drones asuma el papel del semoviente y otro sistema asuma el papel del pastor es muy precaria, al ser un enfoque nuevo de investigación que recién emerge.

En razón a lo expuesto anteriormente, la presente investigación busca dar solución a la siguiente pregunta: ¿De que manera se puede simular un sistema enjambre que pueda modificar su estrategia de formación y se adapte al área de cobertura mediante la técnica de pastoreo?

4.2. Objetivos

4.2.1. Objetivo General

Implementar un ambiente simulado de un sistema enjambre que modifique la estrategia de formación y que se adapte al área de cobertura mediante la técnica de pastoreo.

4.2.2. Objetivos Específicos

- Modelar un sistema de enjambre mediante un entorno de simulación basado en el comportamiento de un rebaño que extraiga la información necesaria del terreno.
- Diseñar una estrategia basada en la técnica de pastoreo que oriente un enjambre de agentes para mejorar la cobertura y modifique su estructura.
- Comparar la técnica de pastoreo con un conjunto de modelos definidos desde el estado del arte a través de índices de desempeño.

4.3. Justificación

El comportamiento social de algunas especies de animales es un tema que ha fascinado a los científicos por ejemplo, el movimiento coreográfico sincronizado de las bandadas de aves, los cardúmenes de peces y los enjambres de abejas siempre han sido un objeto de modelamiento por parte de matemáticos e investigadores, dado que los múltiples beneficios aportados por el trabajo en equipo, son quienes ayudan a la propagación y supervivencia de la especie, sin embargo estos comportamientos sociales hacen parte de un ciclo evolutivo que se ha venido perfeccionando.

Es por esto que, con la creciente demanda de sistemas robóticos, se han venido imitando estos comportamientos, a través de sistemas robóticos autónomos, los cuales usan métodos colaborativos que ayudan a mejorar temas de movilidad, búsqueda y rescate en territorios de desastre, mejorar los tiempos de entrega para sistemas delivery (entrega de paquetes), entre otros. En razón a esto, se establece que los robots son considerados parte de nuestro entorno y pueden actuar para modificarlo, todo esto da lugar a lo que hoy se conoce como ingeniería de enjambre o swarm (IS) por su siglas en inglés, la cual establece los métodos y comportamientos que rigen la interacción de múltiples sistemas autónomos como son; los drones. Ante este panorama,

es indiscutible evidenciar como la naturaleza ha demostrado que por medio de una constante evolución, los sistemas colaborativos son funcionales y altamente efectivos.

En cuanto al uso de las nuevas tecnologías a nivel local, es momento de ir apostando al uso de la robótica colaborativa con sistemas robóticos autónomos dentro de las unidades militares Colombianas, ya que a nivel mundial es lo que se encuentra en vanguardia; adicional a esto los índices de inseguridad son muy elevados y minimizar los riesgos es competencia del gobierno; quien podría poner en marcha cualquier uso de estas tecnologías y financiarlo a gran escala, volviendo a las instituciones de defensa más competitivas a nivel internacional e ir preparando a la fuerza pública para protección frente a los nuevos escenarios que se vienen desarrollando en los conflictos futuros, los cuales contemplan el uso de sistemas colaborativos altamente eficientes como herramienta ante cualquier situación de emergencia. Es por esto que, el gobierno Nacional se encuentra incentivando tecnologías a través de sus lineamientos, que buscan ser líder en ciencia tecnología e innovación, con el fin de poder desarrollar tecnologías para luego exportarlas a los países vecinos y a la industria civil con el fin de volverse sostenible. Mientras tanto, el Ejército Nacional de Colombia ha demostrado su interés en ir generando una independencia tecnológica, la cual le permita desarrollar dispositivos que se adapten fácilmente a la diversidad que se puede encontrar en la geografía Colombiana, con una producción tecnológica propia que le lleve a no depender de tecnologías extranjeras, de este modo pretende iniciar investigaciones en robótica colaborativa y sistemas autónomos remotamente tripulados, que en un futuro sirva como apoyo a las unidades militares. Para ejemplificar lo anteriormente expuesto, este apoyo a unidades militares se refiere al beneficio potencial que puede significar: el control y vigilancia de unidades militares y de fronteras, oportuna acción ante desastres, operaciones de búsqueda y rescate aéreo o terrestre, apoyo a operaciones antiterroristas, apoyo a misiones de desminado humanitario, entre otros.

En consecuencia, la intención de la presente investigación es simular el comportamiento de los perros de pastoreo con las ovejas, a través de la robótica colaborativa con una alta flexibilidad que pueda adaptarse fácilmente a las situaciones que deben atender las fuerzas militares, ya sea en protección y/o detección de amenazas en unidades militares, y hasta situaciones de apoyo en búsqueda y rescate en zonas de desastres por posibles emergencias naturales o no.

4.4. Impacto Social

El proyecto tiene como propósito principal enriquecer las bases teóricas existentes sobre la técnica del pastoreo aplicada en un enjambre de drones. En el momento éstas son muy escasas

y aún se encuentran en fase de pruebas de simulación por los costos que implicaría a gran escala su implementación. Además del impacto en la comunidad científica que se encuentra abordando el tema por su amplio interés de emular el comportamiento de la naturaleza que después de tanta evolución lo ha logrado con éxito; en el ámbito militar tiene una gran acogida por las numerosas aplicaciones que podrían derivar de obtener un sistema de enjambre de drones con formación adaptativa al área de cobertura. Esto incentivaría el uso de la robótica colaborativa de enjambres en vigilancia y protección de unidades militares, apoyo en rescates y en operaciones de búsqueda, apoyo en emergencias por desastres, mejora de los tiempos en alertas tempranas, encontrar rutas óptimas, entre otros. En virtud a lo anterior todo esto beneficia a la sociedad que contaría con lo último en tecnología para la seguridad nacional, del mismo modo las formaciones adaptativas hacen del sistema de enjambres una tecnología flexible para aplicaciones en cualquier entorno y que puede atender fácilmente las necesidades que surgen de imprevistos para salvaguardar la vida de las personas, por ejemplo en rescates donde apremia el tiempo; y las acciones oportunas por parte de los organismos encargados brindan tranquilidad a la comunidad.

De acuerdo a las políticas y lineamientos de proyección social y extensión universitaria se pretende articular con los Objetivos de desarrollo sostenible (ODS) mediante el fomento de la innovación y aumento de la investigación científica en temas de fortalecimiento de la industria aeronáutica basada en sistemas robóticos multiagentes, a través de esto se puede ayudar a fortalecer el uso de la tecnología como mecanismo que favorezca la acción por el clima (ODS 13) y mediante este tipo de tecnologías colaborar en el fortalecimiento de la capacidad de desarrollo de sistemas de alerta temprana para la reducción y gestión de riesgos (ODS 3).

Capítulo 5

Antecendentes

Dado que la técnica que se desea aplicar en este proyecto conocida como pastoreo es relativamente nueva, son contadas las aplicaciones en la temática, en muchos de los casos sólo se limitan a probar sus hipótesis, pero aún no existe una aplicación física. En base a esto se pretende estudiar las diferentes metodologías que han sido empleadas como punto de partida inicial.

En el campo del pastoreo, de acuerdo a investigaciones previas realizadas por *Long et al* en [6], se analiza un compendio de metodologías y algoritmos, mediante el cuál se generan las premisas de control de enjambres de drones a través de una planificación y orientación de alto nivel ejercida por “pastores"hacia otro grupo de agentes con funciones básicas, realizando un análisis comportamental del sistema de pastoreo y definiendo las características que debe cumplir el sistema. Desde las perspectiva mostrada por [6] se anticipa que controlar múltiples agentes coordinados representa un problema un tanto complejo que requiere un alto nivel de planificación de trayectorias y dinámicas de agente de bajo nivel, proponiendo el uso de técnicas de *Machine Learning* para optimizar la planificación de los pastores. Vinculado al concepto de Pastoreo, *Elamvazhuthi et al* en [1] aplica el uso de de máquinas de estados finitos probabilísticas (PSO), basadas en cadenas de Markov, donde un agente líder conduce un grupo de 8 a 10 agentes, sin embargo el hecho de planificar las trayectorias del líder supone un reto de ingeniería para no afectar las probabilidades de movimientos de los seguidores, este proyecto propone sistemas heterogéneos entre UGV y UAV, donde el UAV realiza funciones de líder y los UGV se comportan como seguidores, estudiando la controlabilidad entre los dos subsistemas. Por otra parte, *Xie et al* en [7] propone que para la formación adaptativa de sistemas de enjambres combinada con la técnica similar del pastoreo se puede utilizar una topología de

múltiples líderes, bajo un protocolo que evalúa los estados relativos de los vecinos para encontrar los posibles líderes dentro de la estructura, esto permitiría que la estructura sea guiada por agentes de jerarquía superior.

Por su parte, el trabajo de *Li et al* en [8], propone una red de drones autónomos que con la ayuda de altavoces que "ladran" para suplir la función de los perros pastores, guiando animales de granja como las ovejas para apoyo a los granjeros. El objetivo es que los animales puedan agruparse después de haber estado en una posición dispersa y ser dirigidas a un sitio de resguardo, sin embargo, los efectos del zumbido de los drones no fueron tenidos en cuenta durante la interacción entre los robots y los semovientes. No obstante, se exploraron la bases teóricas que fundamentan la teoría del pastoreo para futuros usos donde se desee aplicar la técnica y se requiera modelar el comportamiento de los pastores para usos en otro tipos de sistemas. Dentro de las técnicas de aprendizaje para los sistemas de pastoreo de drones, *Zhi et al* en [9] demuestra que por medio del uso de Aprendizaje profundo por refuerzo *Deep Reinforcement learning* combinado con técnicas probabilísticas, se pueden entrenar varios pastores para que trabajen de manera colaborativa en ambientes poblados de obstáculos, permitiendo encontrar rutas cortas en menores tiempos. De igual manera *Housein et al* en [10] aplica el aprendizaje por refuerzo para realizar un pastoreo autónomo, pero no emplea ningún algoritmo basado en reglas para determinar los puntos de rutas guías para el pastor, sino que establece dos subtareas para el aprendizaje; una en el que se direcciona a un agente a una ruta de un punto inicial a la meta; y la segunda tarea consiste en elegir si agrupar agentes dispersos o conducir un grupo hacia el destino. Lo que indica que el pastor primero aprende a conducir a una oveja y después aprende a seleccionar si agrupa ovejas dispersas o conduce a un grupo hacia la meta. En este caso se evidencia que el aprendizajes no fue lo suficientemente escalable (que va desde un entorno simple a uno complejo), es decir se emplea una adaptabilidad al medio donde el comportamiento del perro pastor cambia para mantener rendimiento en diferentes entornos, pero no fue ágil en la tarea de aprender a elegir la mejor opción. Sin embargo, uno de los ítems recomendados en este trabajo es el diseño de un plan de estudios para agilizar el aprendizaje.

Por otro lado, *ELsayed et al* en [11] propone dos técnicas de pastoreo usando el concepto de evolución diferencial para entornos cargados de obstáculos, la primera estrategia contempla que el sistema de pastoreo encuentre las rutas óptimas para quedar siempre detrás de las ovejas y la segunda contempla que el pastor calcule y optimice las rutas para las "Ovejas" dejando puntos de referencia sobre el camino óptimo para que éstas sigan los puntos. El concepto de técnicas de evolución diferencial combinadas con técnicas metaheurísticas como la optimización de enjambre de partículas también es utilizado por *Mohamed et al* en [12] donde se implementa la

técnica del pastoreo teniendo en cuenta condiciones realistas de limitaciones en los sensores de los robots, permitiendo hasta en un 50 % del tiempo de finalización de las tareas, sin embargo recomienda tener en cuenta que para asegurar la efectividad de la técnica se debe seguir investigando en su uso en ambientes con un gran número de obstáculos y se realice mediante técnicas de optimización 3D, dado que normalmente el problema se ataca desde una teoría de gráficos con perspectiva 2D. Otra forma de contribuir al uso de pastoreo es la propuesta por *Zhang et al* en [13], teniendo en cuenta las limitaciones de cobertura de los sensores por lo que se propone un pastoreo con múltiples líderes el cual crea subgrupos de rebaños hasta cercarlos en un único grupo, esta técnica no deja la responsabilidad a un único líder y logra incrementar las capacidades de los sensores siendo de gran interés para el uso dentro de este proyecto donde lo que se desea es lograr formaciones adaptativas mediante el uso de múltiples líderes que adapten la formación del rebaño.

Dentro del ámbito militar se muestra que *Wang et al* en [14], propone un sistema basado en modelos conocidos como Vicsek para generar sistemas robustos de formación adaptativa giratoria en torno a un objetivo de interés, logrando el atrapamiento de múltiples objetivos mediante un efecto giratorio en torno a su posición. Sin embargo, también se ha limitado a pruebas de simulación utilizando el simulador Coppeliasim y Matlab. Otra aplicación similar que podría orientarse al campo militar es la de *Chipade et al* [15], ya que realizan pruebas de simulaciones basados en el concepto de defensa y ataque entre grupos de agentes. La metodología empleada consiste en conducir un enjambre de "adversarios" hacia un área segura atravesando diversos obstáculos en un entorno 2D. Para esto, el enjambre "defensor" bloquea el camino de sus adversores o atacantes, utilizando una formación cerrada llamada: *"Stringnet"* cubriendo a sus atacantes. Esta estrategia utilizada por el defensor esta inspirada en el concepto de pastoreo, al guiar un grupo de adversarios a una zona segura alejada del área protegida. Este método de pastoreo denominado *"Stringnet Herding"* usa una combinación de controladores de ciclo abierto para la planificación de la formación de los defensores y controladores de tiempo finito con retroalimentación de estado para rastrear la formación deseada y dentro de este marco, [6] realiza un análisis en cuanto a la técnica de pastoreo para su uso en operaciones de seguridad y defensa en el guiado e intuye que su uso se puede dar para maniobras de combate de de sistemas no tripulados "UxV", búsqueda de intrusos dentro de unidades y recolección y detección de campos minados.

Sobre el asunto de formación de agentes, cabe mencionar la importancia de utilizar formaciones adaptativas al medio, que es uno de los objetivos del presente proyecto, por ejemplo en *Liu*

et al en [16], el efecto aerodinámico de retención en la formación del vuelo de un enjambre de drones fue contemplado para variar la formación en función a las condiciones del viento, es decir, adaptabilidad a las condiciones climáticas. Inspirado inicialmente en el comportamiento de aves e insectos, se logra un ahorro de energía debido a las fuerzas de arrastre y a la sustentación de traslación. Debido a esto también se logra aumentar el rango del vuelo, lo que beneficia a la aplicación de entrega de paquetes por parte del enjambre de drones y a su vez pueden hacerse entregas simultáneas, múltiples y de paquetes más pesados a un mismo destino. En este trabajo también se utilizan algoritmos de optimización para el cálculo de rutas y así mejorar los tiempos de entrega. Una estrategia interesante para la formación adaptativa del sistema de enjambre, es la propuesta por *Guzel et al* en [17], en donde se demuestra que por medio de un algoritmo adaptativo para la formación en formas circulares ayuda a que el enjambre pueda navegar eludiendo obstáculos de manera descentralizada previniendo colisiones dentro de ambientes abarrotados, esta formación adaptativa combinada con algoritmos de seguimientos de líderes podría ayudar a mejorar la estrategia del herding de interés en la presente propuesta.

Del tema igualmente, llama la atención la aplicación la robótica colaborativa de enjambres adaptativos en la micro y nano robótica en condiciones simuladas y reales, lo ha demostrado *Feola et al* en [18], llevando a cabo una coordinación en tiempos de ejecución y de espacio. Para esto empleó algoritmos probabilísticos para el cumplimiento de tareas en el menor tiempo posible, las cuales fueron divididas en dos tipos; fáciles y difíciles. Cada una de las cuales requiere un número diferente de agentes para lograr cierta tarea. Las estrategias adaptativas mejoraron en primera instancia la distribución de los robots oportunamente hacia donde se encuentran las tareas difíciles; y segundo los agentes aprendieron a formarse en equipos especializados para una tarea u otra. Es decir, los robots se agregaron estratégicamente en diferentes espacios para realizar tareas al mismo tiempo de forma paralela o coordinada.

A la hora de iniciar un proyecto que involucre un enjambre robótico hay que estudiar los entornos de simulación para realizar las pruebas iniciales y emular las condiciones de ambiente. En referencia a esto, se halló que en la universidad de Manchester se desarrolló el software BeeGround [19], el cuál consiste en una plataforma de simulación de código abierto para simular aplicaciones de robótica multiagente y de enjambre, la cuál permite población de robots de hasta 1000. Esta plataforma permite simular bajo el motor gráfico *unity3d engine* climas, obstáculos, humedad y la aplicación de técnicas de inteligencia artificial para el control de enjambres. Sin embargo de acuerdo a los estudios realizados por [20] se ha demostrado que el entorno de simulación *Coppeliasim* representa una gran alternativa para la integración con MatLab[14], además

su apartado gráfico puede representar condiciones cercanas a la realidad.

En conclusión y para efectos de desarrollo del presente proyecto se destacan las investigaciones como las mostradas por [6],[9],[10], y [15] las cuales pueden ser un gran punto de partida para iniciar a comprender la fundamentación del comportamiento de sistemas de drones para realizar la técnica de Pastoreo sobre otros subsistemas de jerarquía inferior. En relación a la formación adaptativa de sistemas de enjambres el trabajo de [17] expone los principios de como mediante la técnica del pastoreo se puede adaptar la formación de "ovejas" en ambientes abarrotados muy similar al modelo de [11] y por último, en cuanto a la selección de motores de simulación, se hace énfasis en el uso de Coppeliasim dada su versatilidad y los destacables resultados mostrados por [14] y [4].

Capítulo 6

Marco Teórico

6.0.1. Introducción a la Ingeniería de Enjambres

Los sistemas de enjambre están basados en simular y modelar el comportamiento de especies sociales de animales, normalmente entre los comportamientos mostrados por enjambres de abejas, cardúmenes de peces y algunas especies migratorias las cuales colaboran entre ellas para sortear las condiciones medioambientales mediante un comportamiento de manada. De acuerdo a lo mostrado por *Septfons et al* en [3], los sistemas de enjambres robotizados se definen como *"altamente bio-inspirado cuyo objetivo es modelar, mediante sistemas multiagente, los mecanismos de auto organización y adaptación observados en los organismos vivos. Estos modelos eventualmente dan lugar a algoritmos que simulan fenómenos naturales en software (en una computadora) o hardware (con robots), o sirven como metaheurísticas para problemas de inteligencia artificial"*

Los sistemas basados en inteligencia de enjambres deben obedecer a tres principios básicos mostrados por [2]:

- Flexibilidad: mediante la capacidad de resolver diferentes problemas en diferentes ambientes.
- Robustez: mediante la capacidad de afrontar la pérdida de agentes.
- Escalabilidad: mediante la capacidad de resolver problemas a diferentes escalas independientemente del tamaño del enjambre.

La ingeniería de enjambres se fundamenta bajo las reglas de Reynolds como muestra *Lewis et al* en [25], la cual está basada en tres principios básicos para la navegación colaborativa; contempla que para que un sistema pueda navegar de manera conjunta, debe flanquear los obstáculos que se le presenten y evitar la colisión con los vecinos, los agentes que intervienen en el sistema deben igualar sus rumbos y velocidades y debe existir una cohesión hacia el centro del rebaño que permita estar cerca de los vecinos. Normalmente los sistemas enjambre son vistos desde un punto de vista macroscópico como un conjunto en sí y elementos individuales en ese sistema no tienen cabida para el análisis. Adicionalmente las reglas de Reynolds se encuentran modeladas mediante grafos de comunicación que permiten el análisis de los sistemas desde un punto de vista matemático.

Normalmente el comportamiento colectivo de los sistemas enjambre organiza espacialmente a sus integrantes por medio de formaciones y/o patrones que permiten aprovechar el área de interés donde se mueven los agentes integrantes, pero normalmente lo hacen imitando los comportamientos de especies naturales no domesticadas como aves, peces y abejas, tratando de cumplir las reglas de Reynolds.

6.0.2. Metodologías de Diseño de Sistemas de Enjambres

Tal como describe *Brambilla et al* [2], para el diseño de sistemas de enjambres no existe un manual o metodología de diseño, es la experiencia e intuición del diseñador el ingrediente principal para el desarrollo de sistema, sin embargo los sistemas de enjambres se pueden dividir en metodologías de diseño basado en comportamientos, metodologías de diseño automático y otros métodos de diseño. Las metodologías de diseño basadas en comportamiento, contemplan una manera iterativa de comprobar el comportamiento de los enjambres entre estas se encuentran las máquinas de estados finitos probabilísticas *PSFM*, los movimientos del enjambre se dan de acuerdo a la probabilidad de que un hecho ocurra o no, esta probabilidad se da, de acuerdo a una función limites y como muestran *Li y Tan* en [23], las PSFM han demostrado ser una estrategia con gran estabilidad y con gran balance entre exploración y explotación de la información obtenida por los sensores de los agentes.Otra metodología basada en el comportamiento del enjambre es la enfocada en física virtual, donde los integrantes del enjambre son vistos como puntos de carga dentro de una malla de potenciales eléctricos donde existen fuerzas de atracción y repulsión.

Las metodologías basadas en diseño automático contemplan el uso de técnicas de inteligencia artificial y robótica evolutiva, para esto se utilizan técnicas como el Aprendizaje por refuerzo *Reinforcement Learning* donde a través de recompensas el enjambre va optimizando su comportamiento. Por otro lado, la robótica evolutiva donde los sistemas evolucionan mediante la evolución de una función de fitness.

Por último, se pueden encontrar otras técnicas de diseño de sistemas de enjambres, las cuales contemplan técnicas heurísticas como algoritmos *PST: Particle Swarm Theory* con el fin de optimizar el comportamiento del enjambre.

6.0.3. Comportamientos Básicos de los Sistemas de Enjambres

Los comportamientos de los enjambres están divididos en tres grupos básicos que contemplan la agregación de individuos al enjambre, la dispersión de agentes para la cobertura de áreas y el movimiento coordinado de un enjambre.

6.0.3.1. Comportamiento de Agregación

El comportamiento de agregación consiste en el agrupamiento de los agentes del enjambre en un punto seguro. Para esto *león et al*[20] muestra como mediante gradientes de atracción se genera el consenso de múltiples agentes para la detección de posibles víctimas en zonas de desastre y mediante entornos de simulación poder establecer la dinámica del enjambre, este comportamiento va orientado a la toma de decisiones colectivas del enjambre.
De la misma manera *Cardona et al*[4] muestra como a través del algoritmo *Rendezvous concesnsus* un grupo de agentes se separa para poder cubrir algún área de desastre mediante subenjambres y luego volver a agregarse en un punto de encuentro consensuado generando una formación en este nuevo punto de encuentro.

6.0.3.2. Comportamiento de Dispersión

Es el comportamiento que adquieren los agentes para poder cubrir un área de interés separándose entre ellos *Sempere*[27], muestra que mediante este comportamiento dispersivo se logra una gran cobertura de las áreas de interés. Este comportamiento de dispersión suele ser de gran utilidad cuando el sitio de exploración posee una gran cantidad de obstáculos y se dificulta que el sistema navegue de manera coordinada entrando los agentes en un estado llamado *Paseo aleatorio*.

6.0.3.3. Comportamiento de Movimiento Coordinado

El comportamiento de movimiento coordinado o *Flocking* consiste en la navegación conjunta de un conjunto de dos o más agentes, este movimiento normalmente inspirado en las bandadas de aves, donde estas igualan sus velocidades y realizan vuelos en formación manteniendo patrones de vuelo, *Newaz et al*[21], muestran como a través de algoritmos de exploración 3D para la navegación coordinada mantienen una estructura triangular e introducen el concepto de seguimiento de líder. Según *Sempere* [27] en el comportamiento de movimiento coordinado se aprecian las reglas de Reynolds donde se ve la separación alineación y cohesión.

6.0.4. Otros Comportamientos de Sistemas de Enjambres

Aunque normalmente los sistemas de enjambres se modelan mediante el comportamiento de sistemas basados en una sola especie como aves, abejas o peces, existen otros comportamientos que asemejan la interacción entre especies, estas interacciones pueden ser las mostradas por los perros pastoreo con las ovejas por lo que es objeto de estudio durante este proyecto y consiste en poder controlar un subsistema de enjambre por medio de otro sistema de enjambre maestro superior que sirva como guía al sistema todo el enjambre, como muestra *Dutta et al* en [24].

Después de una extensa revisión bibliográfica se realizo un enfoque basado en los siguientes conceptos representados numéricamente a la hora de simular un modelo de enjambre:

6.1. Algoritmo Boids

Para la creación de enjambres se requiere dar cumplimiento inicialmente a las leyes de Reynolds [32], mediante las cuales se regulan 3 principales movimientos del enjambre: Cohesión, Alineación y Separación, este algoritmo fue desarrollado por Craig Reynolds en 1986 [33], el algoritmo Boids da las pautas necesarias para poder simular el comportamiento de bandadas de pájaros o cardúmenes de peces. Estas leyes se describen de la siguiente forma:

- Cohesión: Es uno de los comportamientos que debe adquirir el enjambre, para ello cada uno de los agentes debe mantener una cohesión y moverse hacia el centro de masa de sus vecinos para mantener la unidad del grupo. Para lograr esto, primero se debe calcular el centro de masa del enjambre (CM), que es la posición media de todos los agentes que pertenecen al enjambre. Luego, se determina el vector que va desde cada uno de

los agentes hacia ese centro de masa, multiplicado por una constante denominada fuerza de cohesión f_c. Al seguir esta ley, los drones en el enjambre se moverán hacia el centro de masa, lo que fomentará la formación de un enjambre cohesionado. La ecuación que muestra finalmente una velocidad de cohesión V_c está dada por:

$$CM = \frac{1}{N}\sum_{n=1}^{N} P_n(x, y)$$

$$V_c = (CM - P_n(x, y)) * f_c$$

Donde P_n Representa el pastor n y N representa el número de pastores totales.

- Alineación: La ley de alineación de Reynolds permite que los agentes que se encuentren dentro de un radio de alineación se orienten hacia la dirección promedio de todos los vecinos cercanos. De esta manera, dicho comportamiento asegura que las direcciones de movimiento de los agentes sigan una trayectoria conjunta, fomentando la cohesión del grupo y la sincronización en su movimiento. La velocidad V_a que deben adquirir para mantener una alineación se logra obteniendo la velocidad promedio de cada uno de los agentes (V_n) y multiplicándola por un factor de alineación (f_a) de la siguiente forma:

$$V_a = \frac{f_a}{N}\sum_{n=1}^{N} V_n$$

- Separación: La ley de separación de Craig Reynolds, es un comportamiento de enjambre que se centra en cómo los drones evitan acercarse demasiado entre sí para evitar colisiones y mantener una posición segura; esto se hace manteniendo una distancia determinada entre los agentes vecinos conocida como radio de separación (R_s). Es así como la velocidad de separación V_s se encuentra dada por:

$$V_s = \frac{f_s}{N_c}\sum_{j=1}^{N_c} \frac{p_i - p_j}{(||p_i - p_j||)^2}$$

donde (N_c) son los vecinos cercanos dentro del radio de separación (R_s) ver *Figura 1*, (p_i) es la posición actual del pastor y (p_j) es la posición del vecino cercano, esto se puede profundizar con la teoría de *Lewis et al* [25].

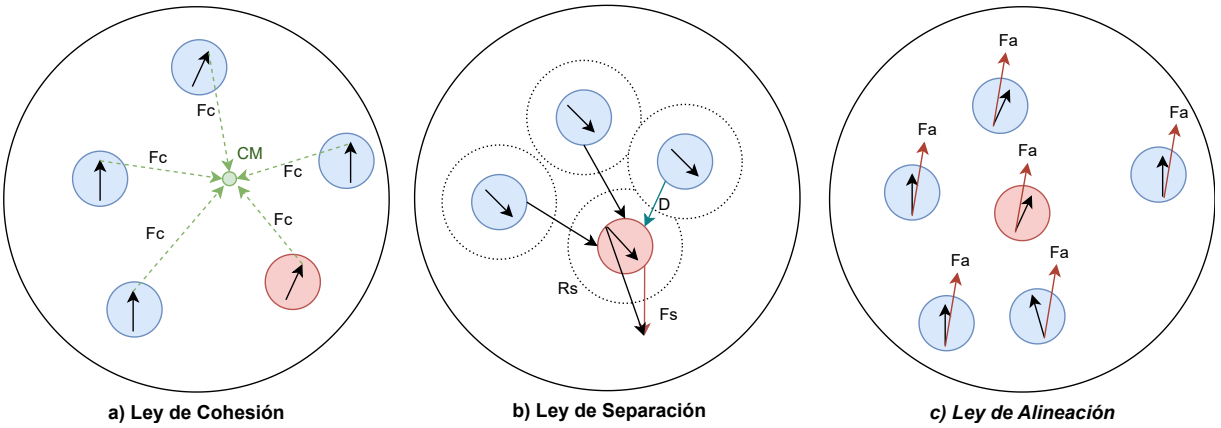

FIGURA 1: Leyes de Reynolds [Autor]

Finalmente, según las leyes de reynolds la posición final de cada agente estaría dada por la siguiente ecuación:

$$p(x(t+1), y(t+1)) = P(x(t), y(t)) + V_c + V_a + V_s$$

Donde:

- $P(x(t), y(t))$ es la posición actual del agente
- V_c es la velocidad de cohesión
- V_a es la velocidad de alineación
- V_s es la velocidad de separación

Estas reglas simples, aplicadas localmente por cada agente, dan lugar a un comportamiento colectivo emergente que imita el movimiento de bandadas o cardúmenes, *Sempere et al* [27] y *Brambilla et al* [2] han abordado el algoritmo Boids en el contexto de sistemas de enjambres de drones, y tiene su base solida en [31], [32] y [33] destacando su relevancia para modelar el comportamiento colectivo y la coordinación.

6.2. Pastoreo de Drones

Antes de abordar un algoritmo para el pastoreo de drones hay que tener claro a que hace referencia dicho termino. El pastoreo de drones se refiere aplicación de técnicas inspiradas en el pastoreo de animales, como ovejas, para controlar y guiar de manera autónoma a un grupo de

drones o robots hacia una posición objetivo. En este contexto, se busca que un dron o un grupo pequeño de drones "pastores" sean capaces de guiar el movimiento de un grupo más grande de drones rebaño, de manera similar a cómo los perros pastores conducen a las ovejas(*Strombom et al*)[28]. Por ende se propone un modelo basado en agentes para resolver el problema del pastoreo, es decir, cómo un agente pastor (perro) puede conducir a un rebaño de agentes (ovejas) hacia un objetivo. En este modelo los agentes (ovejas) se mueven de forma autónoma, siendo atraídos hacia el centro de masa de sus N vecinos más cercanos, repelidos por otros agentes muy próximos, y repelidos por el perro pastor si está dentro de cierto radio de detección. El perro pastor se alterna adaptativamente entre dos comportamientos: recolectar a las ovejas cuando están muy dispersas, y conducirlas hacia el objetivo una vez que están agregadas. Para recolectar, el pastor se mueve hacia la oveja más lejana del centro de masa del rebaño. Para conducir, se posiciona detrás del rebaño con respecto al objetivo. De estos comportamientos emerge un movimiento de zigzag del pastor detrás del rebaño, sin estar explícitamente programado para ello.

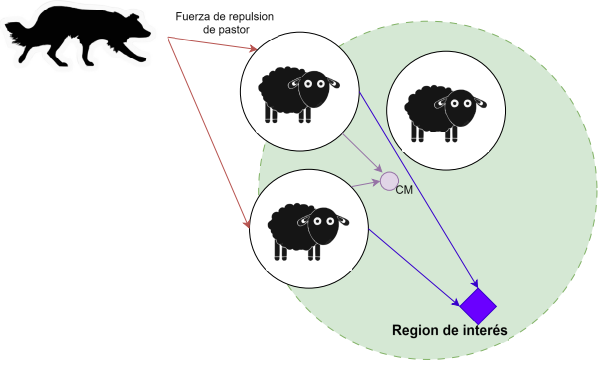

FIGURA 2: Algoritmo Herding Strombom [Autor]

De acuerdo a lo planteado por *Miki et al* [30], el pastoreo ya sea de robots o drones debe respetar unas normas especificas:

- Guiado: Guiar el rebaño hacia una dirección objetivo dada por un supervisor.
- Formar el rebaño: Empujar de vuelta a los individuos que se estén alejando hacia el rebaño.

- Mantener distancia: Mantener una cierta distancia con respecto a los miembros del rebaño para evitar perturbar el orden del rebaño.
- Cooperación : Evitar que los pastores se superpongan entre sí.

Para lograr lo planteado por [30], inicialmente se debe agregar una fuerza de repulsión V_r del pastor S hacia el enjambre, la cual esta definida por una variación de la ley de separación de Reynolds:

$$V_r = \frac{f_r}{N_c} \sum_{j=1}^{N_c} \frac{p_i - S_j}{(||p_i - S_j||)^2}$$

Adicional a esto, se requiere de una distancia mínima D_a de acercamiento entre el pastor S_c determinada por la distancia del vector entre el pastor y el agente más alejado del rebaño O_a. Tal y como se expresa a continuación:

$$V_p = O_a - S_c$$
$$U_p = \frac{V_p}{\|V_p\|}$$
$$S_c(x, y) = O_a(x, y) + U_p * D_a$$

En definitiva para que las ovejas detecten al pastor, este se debe posicionar en el punto $S_c(x, y)$ cercano a las ovejas, esto se logra con el vector unitario U_p que va desde la oveja más alejada del rebaño hasta el pastor y multiplicando este vector unitario por la distancia de alejamiento deseada D_a para ampliar la magnitud de ese vector, sumado a la posición de la oveja más alejada.

6.3. Técnicas de Optimización

La optimización es el proceso de encontrar la mejor solución (generalmente la máxima o mínima) de un problema dentro de un conjunto definido de posibilidades. En términos matemáticos, se busca encontrar el valor de una función objetivo que es mínima o máxima, sujeta a ciertas restricciones.

6.3.1. Optimización Border Collie (BCO)

El algoritmo de Border Collie Optimization (BCO) [24] es un algoritmo de optimización inspirado en el comportamiento de los perros pastores (Border Collies) al guiar y agrupar ovejas. Este algoritmo se basa en principios de inteligencia de enjambre y comportamiento social, donde los perros (agentes) interactúan entre sí, y con las ovejas (objetivo) para lograr una solución óptima. Las características principales de BCO son:

- Interacción Social: Los perros pastores cooperan para guiar las ovejas hacia una meta.
- Fuerzas de Repulsión y Atracción: Se utilizan fuerzas de repulsión para evitar colisiones entre perros y ovejas, y fuerzas de atracción para dirigir a las ovejas hacia el objetivo.
- Adaptabilidad: Los perros ajustan sus posiciones iterativamente basándose en la posición actual de las ovejas y otros perros.

6.3.2. Optimización Particle Swarm (PSO)

El Particle Swarm Optimization (PSO)[22] es un algoritmo de optimización inspirado en el comportamiento de enjambres de aves y peces. En PSO, cada solución potencial (partícula) se mueve en el espacio de búsqueda influenciada por su mejor posición conocida y la mejor posición conocida por el enjambre completo. Las características principales de PSO son:

- Enjambre de Partículas: Un conjunto de partículas (soluciones candidatas) explora el espacio de búsqueda.
- Velocidad y Posición: Cada partícula ajusta su velocidad y posición basándose en su experiencia personal y la del enjambre.
- Mejores Posiciones: Las partículas recuerdan sus mejores posiciones y son atraídas hacia la mejor posición encontrada por cualquier partícula en el enjambre.

6.3.3. Optimización con Fmincon

El algoritmo fmincon es una función en MATLAB [34] utilizada para resolver problemas de optimización no lineal con restricciones. Es adecuado para encontrar mínimos de funciones no lineales con restricciones lineales y no lineales, así como con límites en las variables. Las características principales de fmincon son:

- Función Objetivo No Lineal: fmincon minimiza una función objetivo que puede ser no lineal.
- Restricciones: Maneja restricciones tanto lineales como no lineales, además de límites en las variables.

6.4. Aprendizaje por Refuerzo

El aprendizaje por refuerzo *Reinforcement Learning, (RL)* es un enfoque del aprendizaje automático donde un agente aprende a tomar decisiones interactuando con un entorno, recibiendo recompensas o castigos basado en sus acciones. El objetivo del agente es aprender una ruta óptima, es decir, logra una estrategia que maximice la recompensa acumulada a largo plazo, a medida que aprende de acuerdo al número de iteraciones o acciones realizadas.[29]

Los elementos claves del aprendizaje por refuerzo son:

- Agente: La entidad responsable de aprender y tomar decisiones dentro de un entorno.
- Entorno: El mundo con el que interactúa el agente.
- Estado: La condición presente del entorno.
- Acción: Lo que el agente puede hacer en cada estado.
- Recompensa: La respuesta que el agente obtiene del entorno tras realizar una acción.
- Política: La estrategia agente utiliza para tomar decisiones y lograr un entrenamiento efectivo.

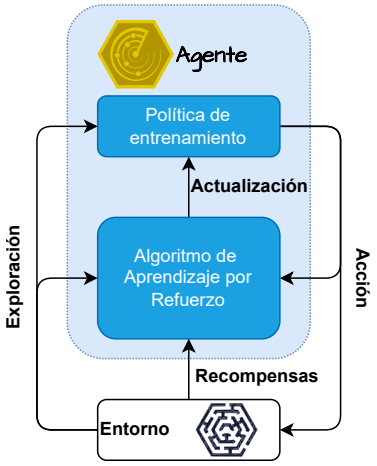

FIGURA 3: Aprendizaje por Refuerzo [Autor]

El Agente explora el entorno tomando acciones y observando las recompensas resultantes. Con el tiempo, aprende a asociar estados y acciones que conducen a mayores recompensas, ajustando su política en consecuencia.

6.4.1. Q-Learning

Q-Learning [35] es un algoritmo de aprendizaje por refuerzo que aprende una función Q denominada "Quality", la cual estima la recompensa futura esperada para cada par (estado-acción). La función Q se actualiza en cada iteración basándose en la experiencia del agente, siguiendo la *Ecuación de Bellman*:

$$Q(s,a) = Q(s,a) + \alpha * \left[R(s,a) + \gamma * max(Q(s',a')) - Q(s,a)\right]$$

- $Q(s,a)$ es el valor Q actual para el de acuerdo al estado y acción realizada.
- α es la tasa de aprendizaje "*learning rate*", que controla cuánto se actualiza Q en cada iteración.
- $R(s,a)$ es la recompensa inmediata recibida por tomar la acción a en el estado s.
- γ es el factor de descuento "*discount Factor*", que pondera la importancia de las recompensas futuras.
- $max(Q(s',a'))$ es la estimación del valor Q óptimo para el siguiente estado s', considerando todas las posibles acciones a'.

Durante el proceso de entrenamiento, el agente alterna entre la explotación de recompensas (seleccionando la acción con el mayor valor Q) y la exploración (seleccionando acciones al azar) para descubrir nuevas estrategias potencialmente mejores. Con suficientes iteraciones, la función Q converge hacia los valores óptimos, y el agente aprende la política óptima.

Una política comúnmente utilizada en el aprendizaje por refuerzo es la política Epsilon-Greedy $(\epsilon - Greedy)$ [29]. La política $(\epsilon - Greedy)$ combina la exploración y la explotación de manera equilibrada. Con una probabilidad ϵ, el agente selecciona una acción aleatoria (exploración), y con una probabilidad $1 - \epsilon$, el agente selecciona la acción con el valor Q máximo estimado (explotación).

La exploración permite al agente descubrir nuevas estrategias potencialmente mejores, mientras que la explotación le permite aprovechar el conocimiento adquirido hasta el momento. El valor de ϵ, generalmente se establece alto al inicio del entrenamiento para fomentar la exploración y se reduce gradualmente a medida que avanza el entrenamiento, lo que permite una mayor explotación a medida que el agente aprende.

El uso de la política $(\epsilon - Greedy)$ permite al agente encontrar un equilibrio entre la exploración de nuevas estrategias y la explotación del conocimiento adquirido, lo que es crucial para un aprendizaje efectivo en entornos complejos y dinámicos.

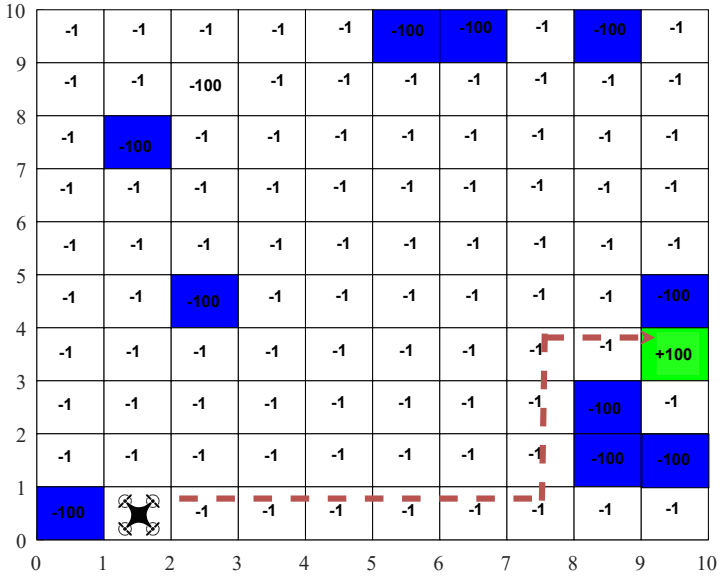

FIGURA 4: Algortitmo *Q-Learning* [Autor]

6.4.2. Aprendizaje por Refuerzo Multiagente

En el aprendizaje por refuerzo multi-agente (Multi-Agent Reinforcement Learning, MARL) [36], donde se utilizan datos de experiencias individuales para cada agente, el objetivo es que múltiples agentes autónomos aprendan a tomar decisiones óptimas en un entorno compartido, basándose en sus propias interacciones y recompensas.

A diferencia del aprendizaje por refuerzo de un solo agente, en MARL cada agente tiene su propio conjunto de datos de experiencias, que consiste en las transiciones (estado, acción, recompensa, siguiente estado) que ha encontrado al interactuar con el entorno. Estos datos son específicos para cada agente y reflejan su perspectiva individual del entorno y las consecuencias de sus acciones [37].

Sin embargo, dado que los agentes comparten el mismo entorno, sus acciones pueden influir en los estados y recompensas experimentados por otros agentes. Esto introduce desafíos adicionales, como la coordinación y la aparición de dinámicas emergentes complejas. Para abordar esto, los algoritmos de MARL a menudo incorporan mecanismos para promover la cooperación o tratar con las acciones introducidas por los comportamientos de otros agentes en aprendizaje.

Algunos enfoques comunes en MARL con datos de experiencias individuales incluyen:

- Aprendizaje Independiente: Cada agente aprende de forma independiente utilizando solo sus propios datos, tratando a los otros agentes como parte del entorno, en este entorno descentralizado es común el uso de la técnica Q-Learning.
- Aprendizaje Centralizado con Ejecución Descentralizada: Los agentes aprenden en una configuración centralizada con acceso a información global, pero ejecutan sus políticas de forma descentralizada basándose solo en sus observaciones locales.
- Aprendizaje con Comunicación: Los agentes pueden compartir información a través de mecanismos de comunicación para mejorar la coordinación y el aprendizaje grupal, pero sin acceso directo a datos de experiencias de otros.
- Heterogeneidad de los Agentes: Los agentes pueden ser homogéneos (con el mismo espacio de estados y acciones) o heterogéneos. La homogeneidad permite compartir parámetros y reducir el tiempo de entrenamiento, mientras que la heterogeneidad puede abordar escenarios más diversos y realistas.

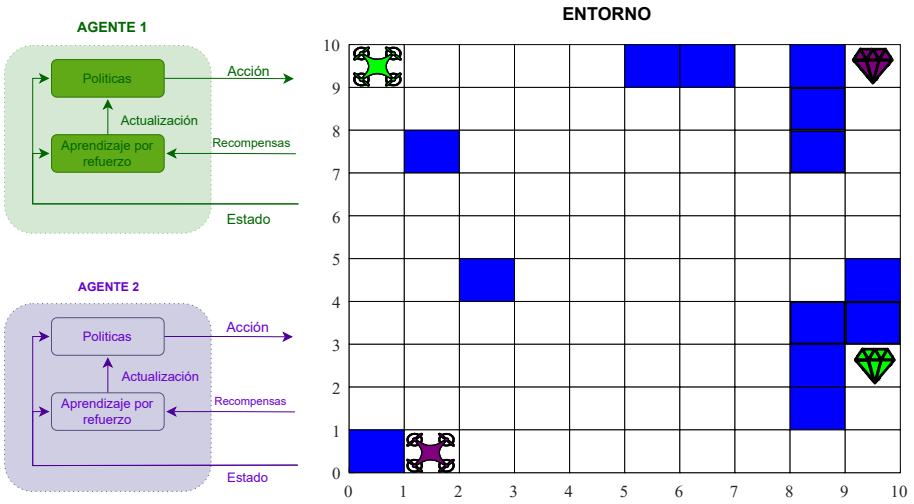

FIGURA 5: Aprendizaje por Refuerzo Multiagente [Autor]

En resumen, el aprendizaje por refuerzo multiagente (MARL) permite a múltiples agentes autónomos aprender simultáneamente a partir de sus propias interacciones y experiencias individuales. Este enfoque se centra en encontrar soluciones óptimas para agentes que comparten un entorno común, permitiendo el descubrimiento de estrategias útiles de coordinación y comportamiento emergente para alcanzar objetivos tanto compartidos como competitivos. Esto es especialmente relevante en escenarios de movilidad autónoma y gestión del tráfico, donde la cooperación y la comunicación entre vehículos son esenciales para alcanzar objetivos comunes y evitar colisiones [38].

Capítulo 7

Diseño y Metodología

El diseño y desarrollo de sistemas basados en enjambres de drones requiere de experimentación e investigación con el fin de validar el funcionamiento de las teorías emergentes. Inicialmente se realiza un análisis conceptual y teórico donde se logre la comprensión de las diferentes teorías, métodos y fundamentos matemáticos, y aunque como lo planteaba [2] para el diseño de sistemas de enjambres no existe una metodología exacta, en la cual el principal ingrediente es la pericia y experiencia del diseñador, para sistemas bioinspirados basados en comportamientos sociales de seguimiento de lideres, se propone el estudio de temáticas relacionadas al desarrollo de algoritmos de optimización basados en técnicas metaheurísticas, sistemas de simulación, y generación de modelos cinemáticos de robots móviles entre otros. Teniendo en cuenta la complejidad del sistema el diseño metodológico se enfoca de manera mixta en la cual de forma exploratoria se intentará validar la estrategia de formación adaptativa de sistemas de drones, por medio de la técnica del pastoreo y cuantificar su eficiencia de acuerdo a otros métodos de formación de enjambres, para esto la investigación se realizará en 4 pasos descritos a continuación:

7.1. Revisión Bibliográfica

Al realizar la revisión del estado del arte buscando el fundamento necesario y conocer diferentes técnicas, conceptos, soluciones y tendencias referentes al pastoreo de enjambres y su implementación, se dio con una solución capaz de cumplir los aspectos necesarios para la formación adaptativa de sistemas de enjambres con la implementación de la técnica del pastoreo. Dado

que la técnica del pastoreo puede ser implementada de diferentes formas, se evaluaron diferentes de ellas analizando los índices de desempeño, facilidad de uso y complejidad con el fin de adoptar una solución que sea capaz de cumplir con las necesidades del proyecto. De toda esta revisión básicamente el algoritmo boids, sus variaciones, entre otras estrategias; permitieron abordar el proyecto, pero en resumen, se hizo énfasis en:

- Algoritmo boids
- Algoritmo Leader Follower
- Algoritmos basados en la técnica de Strombrom
- Q-learning y MARL
- Técnicas de optimización
    - Optimización Border collie (BCO)
    - Optimización Particle Swarm (PSO)
    - Optimización Fmincon

Una vez que se comprendieron los principios fundamentales y las teorías que sustentan cada uno de estos algoritmos y técnicas, se procedió a la etapa de modelado y diseño. En esta fase, se desarrollaron modelos e integraciones que aprovecharon las fortalezas de cada enfoque, buscando soluciones eficientes.

7.2. Modelo y Diseño

Al tratarse de un sistema jerárquico entre dos subconjuntos de drones (Pastores Vs. Ovejas), se realizaron pruebas mediante simulación numérica, las cuales incluyeron el modelado e implementación de bajo nivel en el software Matlab. Estas simulaciones numéricas implicaron la simulación de algoritmos de seguimiento de líderes, evaluando la técnica del pastoreo y el desempeño según las condiciones iniciales. El diseño contempló el uso y elaboración de modelos de interacción entre enjambres para la aplicación de la técnica de pastoreo, con el fin de lograr una cohesión y un seguimiento óptimo entre los conjuntos de drones durante el pastoreo.

7.3. Simulación

En esta etapa se contempla la simulación y aplicación de los algoritmos de pastoreo basado en entornos de simulación gráfica (CoppeliaSIM), esto permite evaluar comportamientos de una manera visual y evaluar las diferentes restricciones, ya sea por las necesidades del entorno o por las necesidades de la misión asignada, para esto fue de gran apoyo las simulaciones gráficas que se adapten semejantemente a un entorno real. Sin embargo, las simulaciones numéricas realizadas inicialmente en Matlab demandaron un protocolo de comunicación para integrar estos dos tipos de simulaciones; fue así que con la comunicación UDP bajo una API proporcionada por CoppeliaSIM y con los comandos necesarios permitieron la interacción entre los dos tipos de software.

7.3.1. Simulación en Entorno CoppeliaSIM:

Tras las simulaciones preliminares(numéricas), se utilizaron entornos de simulación más avanzados, como CoppeliaSIM, para validar los resultados obtenidos en las pruebas iniciales. CoppeliaSIM proporcionó un entorno más realista y detallado para evaluar el comportamiento de los algoritmos en condiciones más cercanas a las del mundo real, esto permitió hacer ajustes a los algoritmos para hacer que se cumplan las condiciones cinemáticas y de entorno.

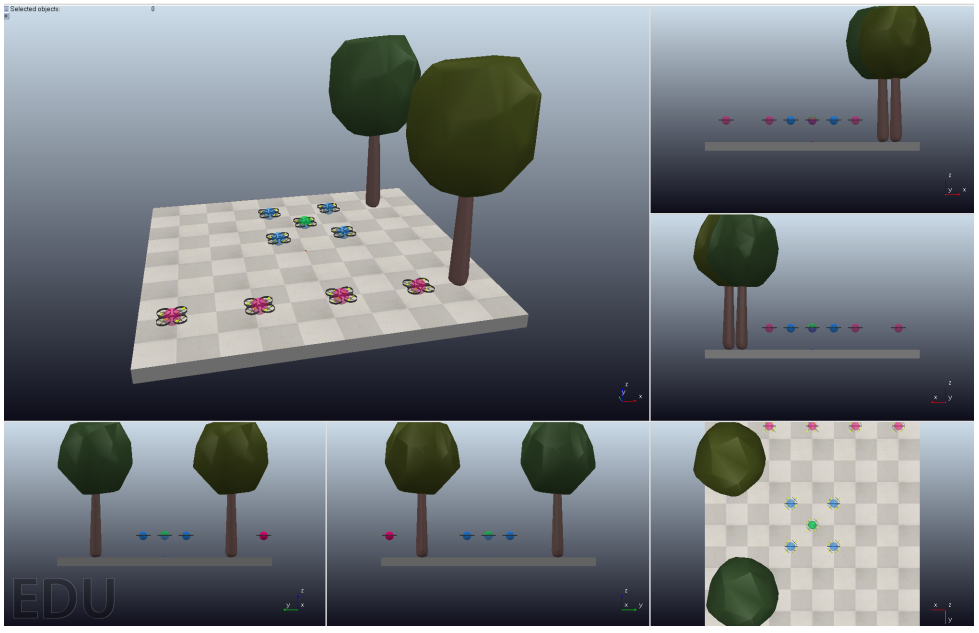

FIGURA 6: Entorno CoppeliaSIM [Autor]

7.4. Validación Experimental

Se realiza una validación experimental de los resultados mediante técnicas estadísticas que permite comparar la técnica del pastoreo con otras técnicas como la Boids Vs. Boids y algunas otras encontradas en el estado del arte, para esto se tienen en cuenta índices de desempeño como porcentajes de cobertura del área de exploración y tiempos de convergencia, estos índices de desempeños se pueden calcular a través de métricas tales como; tiempos de iteración, convergencia entre otras.

Capítulo 8

Diseño e Implementación

Inicialmente, se estableció un sólido fundamento teórico mediante una exhaustiva búsqueda de fuentes bibliográficas. Este primer paso fue crucial para comprender y analizar los conceptos y enfoques propuestos por diversos autores en el campo de estudio, lo cual proporcionó una base sólida para el desarrollo de las siguientes etapas del proyecto. Posterior a esto, el desarrollo del proyecto se llevó a cabo siguiendo una serie de pasos meticulosamente planificados, descritos a continuación:

8.1. Simulaciones

Se realizaron simulaciones iniciales utilizando entornos de bajo costo gráfico (Matlab) para probar diversos algoritmos y técnicas. Los algoritmos evaluados fueron los siguientes:

- Boids: simulación del movimiento aleatorio de las ovejas basado en las leyes de Reynolds.
- Pastoreo de Strombom: en este modelo las ovejas se mueven aleatoriamente en función de un centro de masa y son repelidas del pastor si están dentro del radio de detección.
- Optimización Particle Swarm (**PSO**): un conjunto de posibles soluciones, llamadas partículas, se mueven a través del espacio de búsqueda. Cada partícula ajusta su posición en función de su propia mejor experiencia y de la experiencia colectiva del enjambre
- Optimización Border Collie (**BCO**): utiliza algoritmos de optimización para coordinar el movimiento de múltiples drones de manera óptima, de esta forma mejora el rendimiento y la eficiencia de los sistemas de drones en diversas aplicaciones.

- Q-learning: con este algoritmo el agente aprende a tomar decisiones óptimas en función de las recompensas esperadas. El agente aprende una función Q, que estima el valor esperado de tomar una acción en un estado dado y sigue una política que maximiza esta función Q.

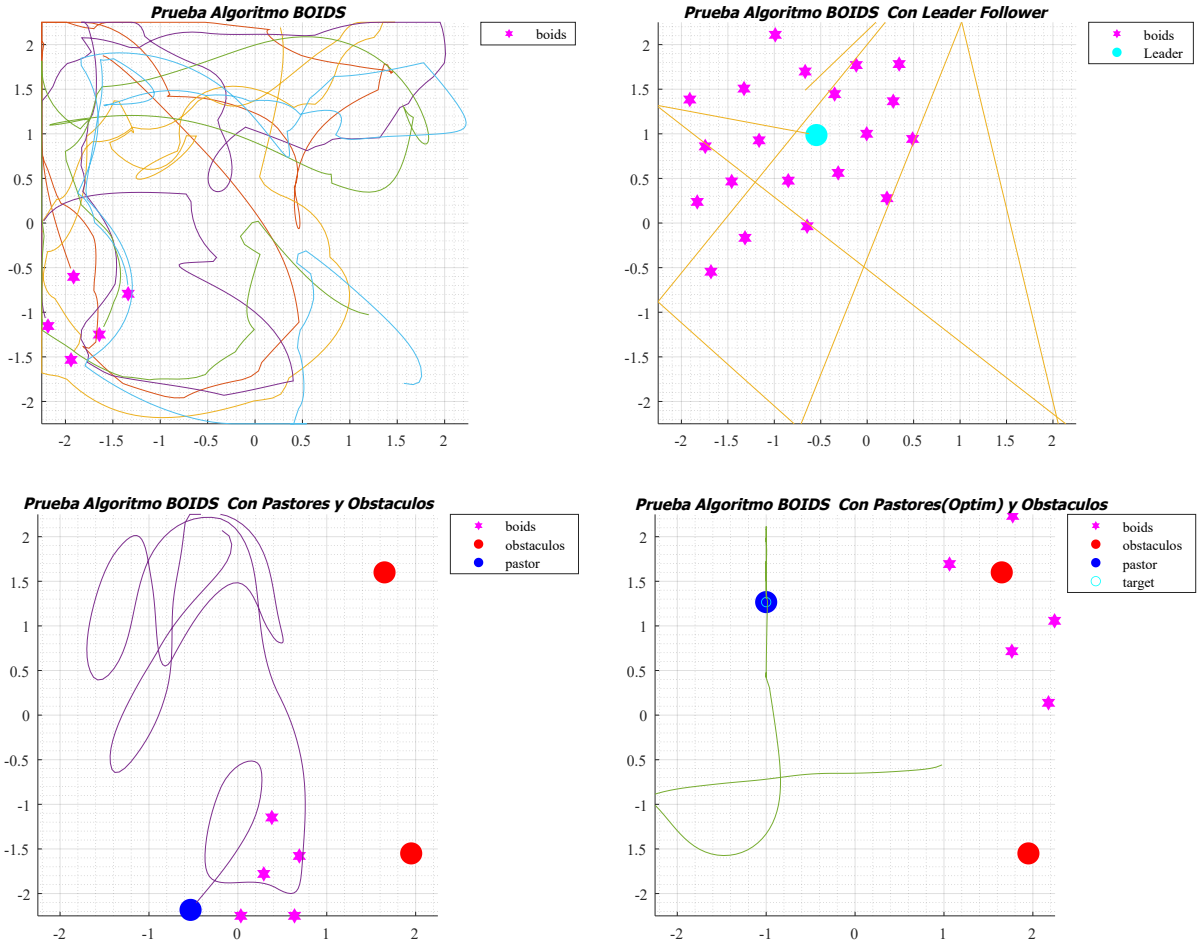

FIGURA 7: Pruebas de Algoritmos de Diferentes Técnicas [Autor]

Estas simulaciones preliminares permitieron identificar las ventajas y limitaciones de cada algoritmo en un entorno controlado y económico en términos de recursos computacionales, asimismo permitieron evaluar tiempos de convergencia y tiempos de ejecución de cada algoritmo.

8.2. Diseño De Métodos De Pastoreo E Interacción

8.2.1. Diseño de Método para Selección de Target Basado en Optimización

El término "*Target*" hace referencia a las posiciones que debe alcanzar el pastor para ejercer el pastoreo sobre el enjambre. Para la selección del Target de los pastores, se utilizaron dos técnicas. La primera, denominada "distancias cuadráticas inversas", verifica cuál oveja se encuentra más cercana a la frontera de la región de interés. La distancia del pastor hacia esa oveja se calcula como $1/D^2$ de la distancia de la oveja al centro de la región de interés. De esta manera, si la oveja está muy alejada de la región de interés, el pastor ejercerá una mayor repulsión sobre ella, y si la oveja está muy cerca de la región de interés, el pastor se alejará para dejarla explorar dentro de esa área.

El segundo modelo es el de "posicionamiento óptimo". Esta estrategia consiste en posicionar al pastor en las coordenadas (x, y) óptimas para que ejerza una fuerza de repulsión optimizada sobre el enjambre, de manera que pueda encerrar a las ovejas mucho más rápido.

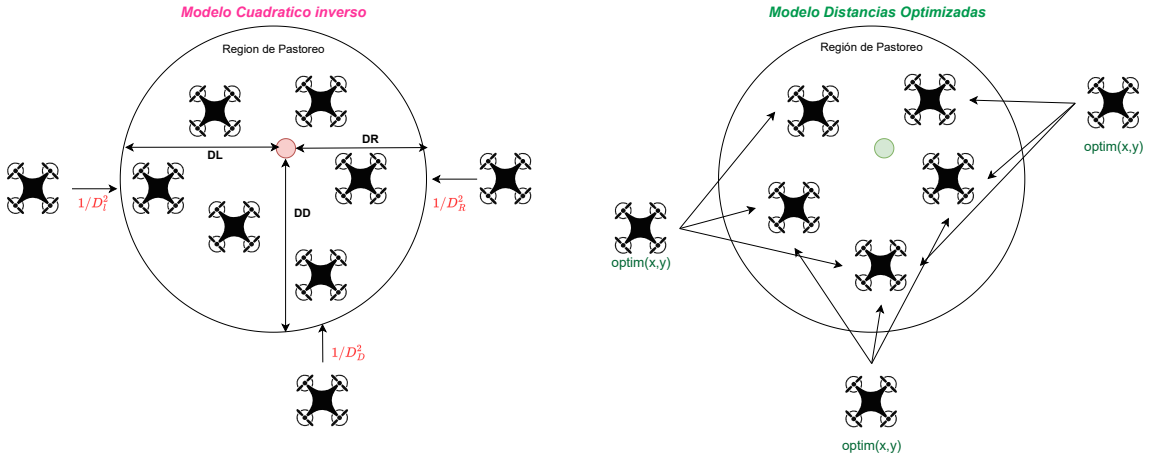

FIGURA 8: Métodos de selección de objetivo o *Target* para pastores [Autor]

Para la selección de la función objetivo o *Target* del método *optimizado*, se tuvo en cuenta la distancia entre los pastores y el enjambre, buscando minimizar esta distancia. Por lo cual:

$$Target(x, y) = min\left(\sum_{o=1}^{n} \sqrt{(x_s - X_o)^2 + (y_s - Y_o)^2}\right)$$

donde (x_s, y_s) son los *Target* para los pastores, y (X_o, Y_o) son los vectores que contienen las posiciones de las ovejas pertenecientes al enjambre, para este algoritmo de acuerdo al pastor se debe tener en cuenta las restricciones de acercamiento de cada pastor de acuerdo a un radio de acercamiento mínimo y la fuerza de repulsión aplicada. Para encontrar esta distancia mínima se utilizo la función *fmincon* de Matlab, la cual genera tiempos más cortos de solución en comparación a las técnicas BCO y PSO.

8.2.2. Diseño de Enjambre Basado en Boids

Una vez analizados los comportamientos de los enjambres, se desarrolló un algoritmo de pastoreo para las ovejas, utilizando la información de los pastores y la fuerza de repulsión que estos deben generar hacia el enjambre. Además, el algoritmo Boids modificado también considera las condiciones del terreno para evitar colisiones con obstáculos. Este procedimiento se puede encontrar en el Apéndice A algoritmo 2.

8.2.3. Diseño de Algoritmo de Mapeo Basado en Q-Learning

Inicialmente, se llevó a cabo un mapeo detallado para la generación de la matriz de recompensas *R*. Este proceso implicó la identificación y asignación de recompensas específicas a diferentes estados dentro del entorno, lo cual es fundamental para la correcta implementación del algoritmo de aprendizaje por refuerzo.

Posteriormente, tras la generación de la matriz *R*, se realizó la proyección de trayectorias "secuestradas", es decir que se interponen posiciones generando error de convergencia del algoritmo. Esto se hizo para verificar que los *Target* proporcionados por el algoritmo de proyección de objetivos o *Target* no coincidieran con la posición ocupada por las ovejas, otros pastores u obstáculos. Si esto llegara a suceder, el algoritmo de Q-learning podría quedar atrapado en un bucle indeterminado, haciendo que el tiempo de entrenamiento tienda a infinito y saturando los valores de la tabla *Q* y no obtendríamos un resultado convergente. Finalmente, se procedió con el entrenamiento de la tabla *Q*. Este proceso involucró iteraciones continuas y ajustes finos a medida que los agentes aprendían a maximizar sus recompensas acumuladas.

Cada una de estas etapas fue crucial para la integración del algoritmo de Q-learning con el generador de trayectorias y el algoritmo de movimiento del enjambre de ovejas exploradoras. Este resultado puede verse en el Apéndice B, Algoritmo 3.

8.2.4. Diseño Pastoreo Basado en Q_Learning

Basado en los conocimientos adquiridos y las simulaciones previas, se diseñaron diversas estrategias de pastoreo que incorporan técnicas de algoritmo Boids-modificado, Strombom y Q-learning, junto con cálculos de objetivos o *Target*. Como resultado de la investigación y después de diferentes pruebas, se logró crear un nuevo modelo de pastoreo que integra estas técnicas y consigue contener a las ovejas en el menor tiempo posible. El pastoreo con Q-learning arrojó los mejores resultados, logrando que, gracias a la acción de la fuerzas repulsión de los pastores, las ovejas adoptaran una formación adaptativa dentro de la región de interés.

Esta técnica de pastoreo con Q-learning aprovecha la información del entorno obtenida por las ovejas con el fin de alimentar la matriz de recompensas R con puntuaciones negativas. Asimismo, aprovecha las posiciones de los demás pastores para alimentar la matriz de recompensas R de cada pastor con puntuaciones negativas y evitar colisiones entre ellos. También utiliza los resultados de la técnica de cálculo de objetivos o *Target* para alimentar la matriz R, generando la recompensa máxima para cada pastor. Al tratarse de un entorno cambiante debido al movimiento de las ovejas, se optó por utilizar el aprendizaje por refuerzo de múltiples agentes con aprendizaje independiente para cada pastor. Esto se puede observar en la *figura 9* y en el Algoritmo 1, que es el resultado de esta investigación:

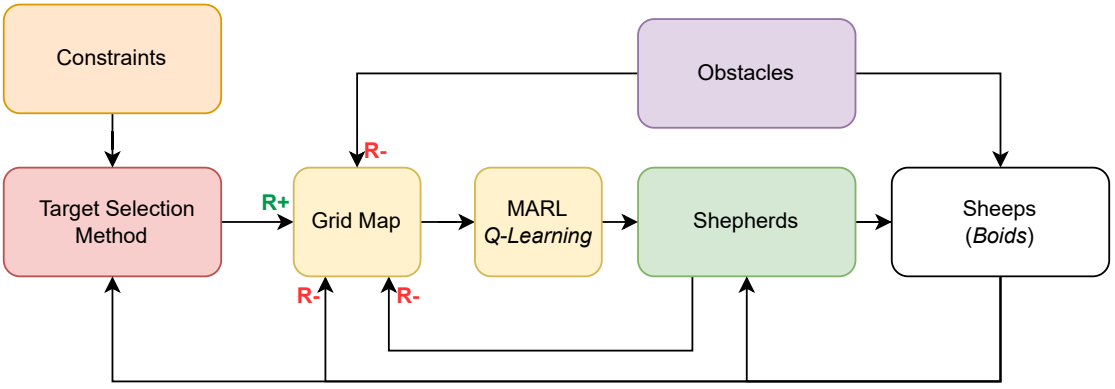

FIGURA 9: Tecnica Pastoreo Q-Learning [Autor]

Algorithm 1 Técnica pastoreo con Q_Learning

Input: num_iterations, num_boids, positions, cohesion_factor, separation_factor
Input: alignment_factor, obstacles_avoidance_factor, shepherds_avoidance_factor, L, R, U, D
for $t = 1$ **to** num_iterations **do**
  $center\_of\_mass \leftarrow$ **mean**$(positions)$
  for $boid = 1$ **to** num_boids **do**
    $cohesion \leftarrow cohesion \times cohesion\_factor$
    $separation \leftarrow separation \times separation\_factor$
    $alignment \leftarrow alignment \times alignment\_factor$
    $obstacles\_avoidance \leftarrow obstacles\_avoidance \times obstacles\_avoidance\_factor$
    $shepherds\_avoidance \leftarrow shepherds\_avoidance \times shepherds\_avoidance\_factor$
    $Vels \leftarrow Vels + cohesion + separation + alignment + obstacles\_avoidance + shepherds\_avoidance$
    $pos\_boid \leftarrow pos\_boid + Vels$
  end for
  $target\_pastores(4, 2) \leftarrow$ **calc_target**(L, R, U, D)
  $map(4) \leftarrow$ **map_scenario**$(target\_pastores, pos\_boid, pos\_shepperds)$
  $shepherd\_L \leftarrow$ **Q_Learn_Train_Agent**$(map(1))$
  $shepherd\_R \leftarrow$ **Q_Learn_Train_Agent**$(map(2))$
  $shepherd\_U \leftarrow$ **Q_Learn_Train_Agent**$(map(3))$
  $shepherd\_D \leftarrow$ **Q_Learn_Train_Agent**$(map(4))$
end for

8.3. Validación Experimental de la Nueva Técnica

La nueva técnica de pastoreo se sometió a una validación experimental rigurosa. Se realizaron comparaciones con otras técnicas existentes, como Boids Vs. Boids y Boids Vs. Leader Follower. Los criterios de evaluación incluyeron el tiempo de cobertura de área y el número de iteraciones necesarias para lograr el pastoreo. Estas comparaciones permitieron evaluar el desempeño de la nueva técnica en términos de eficiencia y efectividad, demostrando su superioridad en los aspectos evaluados.

Esta metodología estructurada garantizó un enfoque sistemático y riguroso en el desarrollo y validación del nuevo algoritmo de pastoreo, proporcionando resultados valiosos para el campo de estudio.

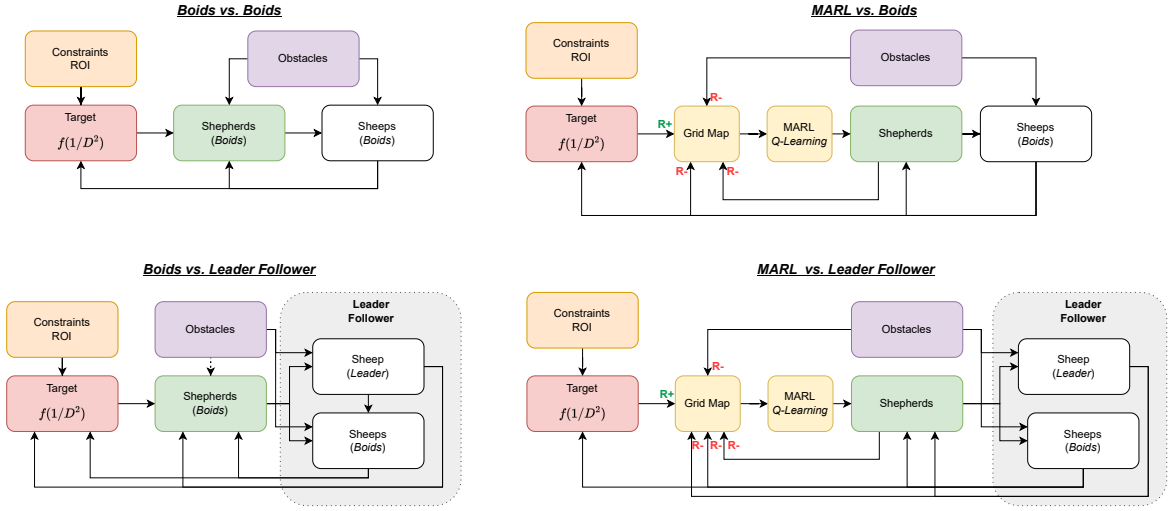

FIGURA 10: Métodos de experimentación con objetivos $1/D^2$ [Autor]

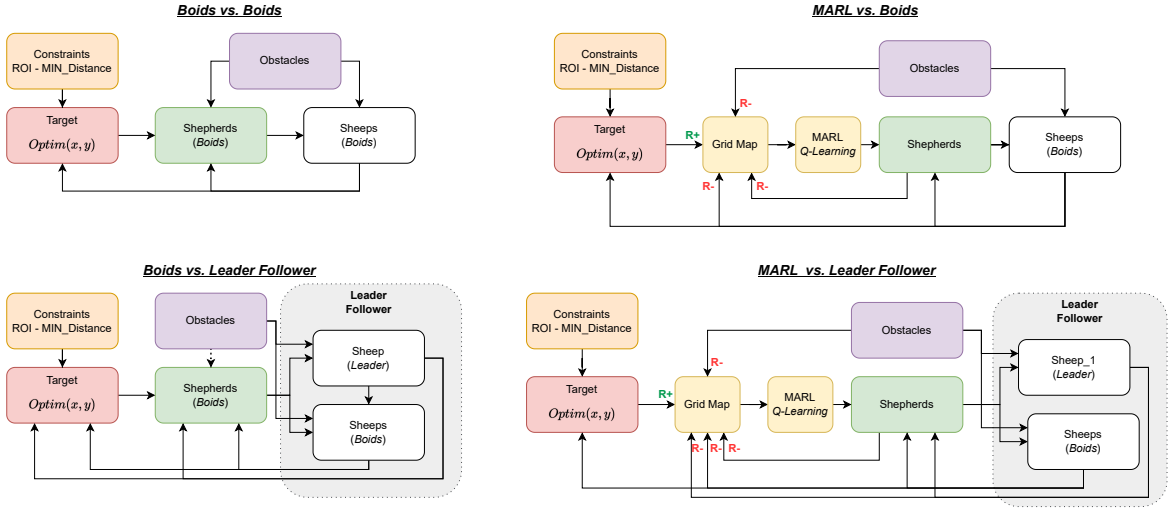

FIGURA 11: Métodos de experimentación con objetivos $fmincon$ [Autor]

Capítulo 9

Resultados y Discusión

9.0.1. Simulación de Enjambre con Técnica de Pastoreo

Para modelar un sistema de enjambre mediante un entorno de simulación basado en el comportamiento de un rebaño que extraiga la información necesaria del terreno, se exploraron diversas técnicas. Sin embargo, las que generaron mayor interés y baja complejidad computacional fueron las técnicas basadas en los algoritmos Boids y Leader-Follower. Estas técnicas, con un buen ajuste de ganancias, pueden lograr una buena dispersión sobre la región de interés, cubriendo el área de manera eficiente y extrayendo información acerca de los posibles obstáculos que se encuentran en el lugar.

En cuanto a las técnicas de pastoreo, se sometieron a prueba diferentes técnicas de pastoreo y optimización con el fin de verificar su convergencia y costo computacional. Entre ellas, se encuentran las propuestas por Strömbom, BCO (Border collie Optimization), PSO (Particle Swarm Optimization) y, como resultado final, una técnica nueva basada en IA (Inteligencia Artificial) de Aprendizaje por Refuerzo.

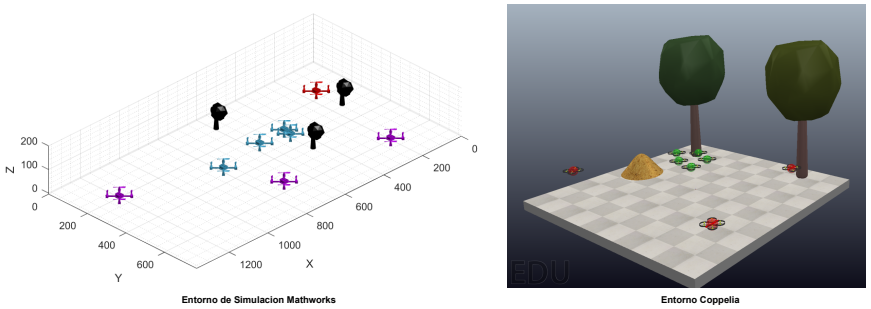

FIGURA 12: Entornos de simulación probados [Autor]

9.0.2. Estrategia para Mayor Cobertura y Estructura en Pastoreo

Luego de evaluar diferentes técnicas, se realizó el diseño de una técnica de pastoreo basada en la combinación del algoritmo Boids modificado, con separación de los pastores (Shepherds Avoidance Factor), un algoritmo de cálculo de objetivos o *Target* (por funciones cuadráticas inversas o puntos de acercamiento óptimos mediante el algoritmo fmincon) y la aplicación de MARL (Multi-Agent Reinforcement Learning) con un enfoque de aprendizaje independiente para cada pastor.

9.0.3. Comparativas de Pastoreo entre Diferentes Modelos

Después de implementar diversas técnicas de pastoreo y establecer objetivos claros para los pastores, se llevó a cabo una evaluación exhaustiva del rendimiento de cada una de ellas. El propósito principal fue identificar la técnica más eficaz para mantener a las ovejas dentro del área de pastoreo designada. Las técnicas se evaluaron según el siguiente cuadro:

MODELO	ACRÓNIMO	ALGORITMO OVEJAS	ALGORITMO PASTORES	SELECCIÓN OBJETIVOS
Modelo 1	BBM-CI	Boids	Boids Modificado	Cuadrático Inverso
Modelo 2	LFBM-CI	Leader Follower	Boids Modificado	Cuadrático Inverso
Modelo 3	BM-CI	Boids	MARL	Cuadrático Inverso
Modelo 4	LFM-CI	Leader Follower	MARL	Cuadrático Inverso
Modelo 5	BBM-OPT	Boids	Boids Modificado	Optimización fmincon
Modelo 6	LFBM-OPT	Leader Follower	Boids Modificado	Optimización fmincon
Modelo 7	BM-OPT	Boids	MARL	Optimización fmincon
Modelo 8	LFM-OPT	Leader Follower	MARL	Optimización fmincon

CUADRO 1: Modelos de Comparación para Simulaciones

9.0.3.1. Resultados Modelo BBM-CI

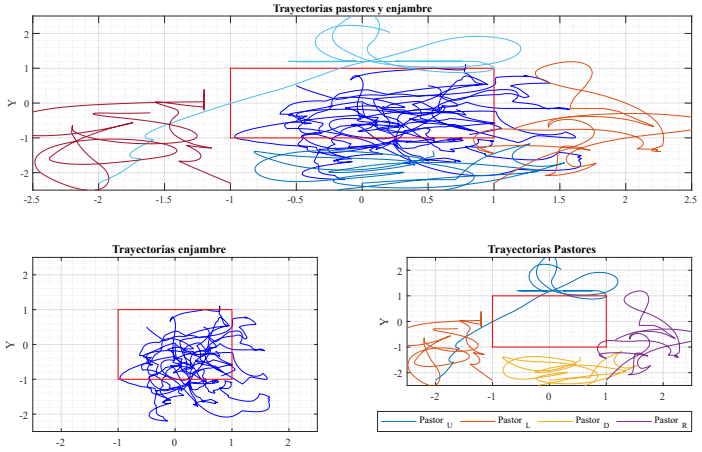

FIGURA 13: Trayectorias BBM-CI [Autor]

Según se puede observar en la *Figura 13*, las trayectorias del algoritmo BBM-CI muestran que el enjambre intenta ser contenido en la región de interés, aunque con poca precisión. Esto se debe a que el comportamiento de los pastores es muy primitivo y poco estratégico. Sin embargo, el uso de la estrategia mejorada de boids en los pastores les permite alcanzar su objetivo. A pesar de ello, las rutas seguidas por los pastores atraviesan el enjambre, generando una dispersión o separación durante una parte de la simulación, lo que aumenta el tiempo de convergencia del algoritmo. Se puede notar que el pastor superior (pastor_U) pasa por el área de interés, lo que hace que el enjambre se desplace hacia el sureste, incluso saliéndose de la región de interés.

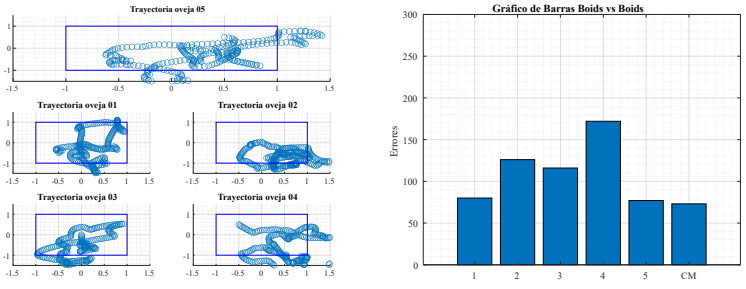

FIGURA 14: Trayectorias Ovejas BBM-CI [Autor]

Al realizar un análisis de las trayectorias de las ovejas, se observa que en *Figura 14*, durante un total de 300 iteraciones, se midió para cada oveja las veces que estuvieron fuera del área de interés, lo que denota un error acumulado del 38 %. Además, hubo momentos en que el centro de masas del enjambre tendía a estar fuera del área de interés.

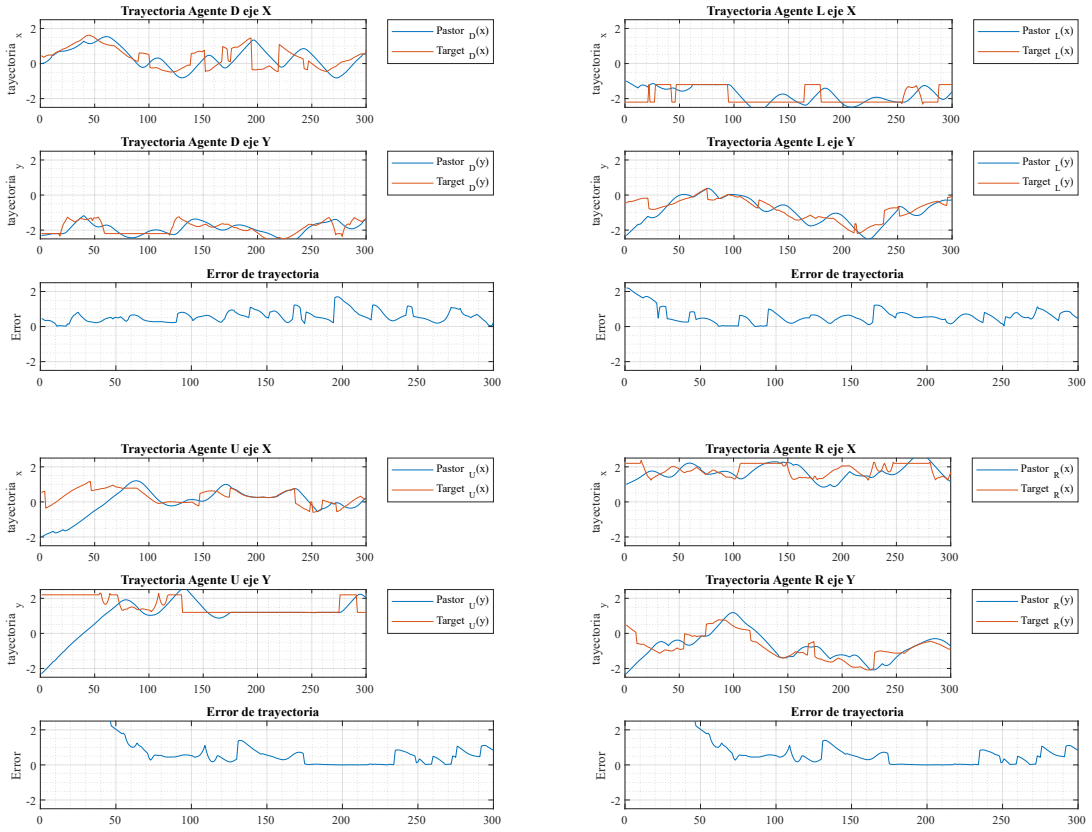

FIGURA 15: Trayectorias Pastores BBM-CI [Autor]

Según se puede ver en la *Figura 15*, el pastor más lejano (pastor_U) demora hasta 110 iteraciones para alcanzar el objetivo en las posiciones (x, y). Es notable que el error de los pastores fluctúa durante la simulación, y difícilmente los pastores alcanzan un valor de error de 0.

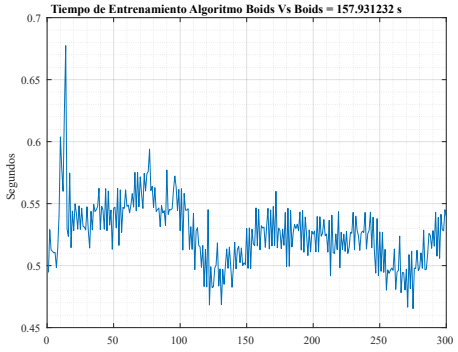

FIGURA 16: Tiempo de Compilación Modelo BBM-CI [Autor]

Midiendo el desempeño del algoritmo BBM-CI en términos de gasto computacional, se realizó la medición del tiempo necesario para completar 300 iteraciones (*Figura 16*), notando que tomó 157.93 segundos, lo que equivale a unos 526 ms por iteración. Este gasto computacional depende de la cohesión del enjambre y de la repulsión ejercida por los pastores, ya que, a medida que más fuerzas actúan sobre el algoritmo y se superan los radios de separación y repulsión, se requieren más cálculos por iteración

9.0.3.2. Resultados Modelo LFBM-CI

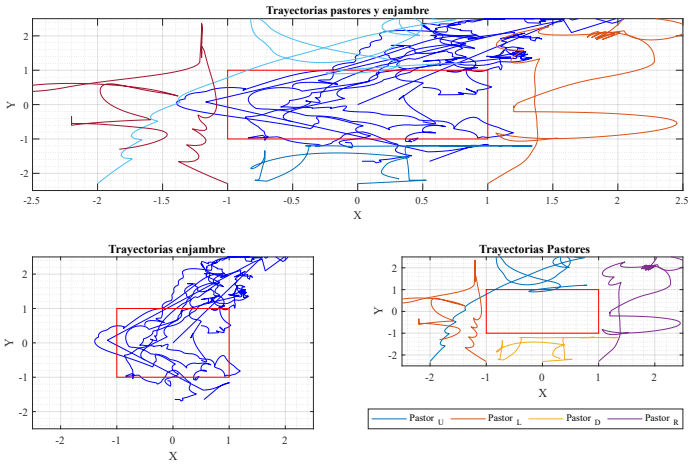

FIGURA 17: Trayectorias Modelo LFBM-CI [Autor]

Según se puede observar en la *Figura 17* , las trayectorias del algoritmo LFBM-CI muestran que el enjambre intenta ser contenido en la región de interés, aunque con poca precisión. En esta ocasión, la fuga se da hacia la parte superior, que es la región controlada por el pastor U. El enjambre aprovecha el tiempo que tarda el pastor U en alcanzar su objetivo y, debido a que la oveja 5 actúa como líder del enjambre, este tiende a buscar el área que ofrece menos resistencia.

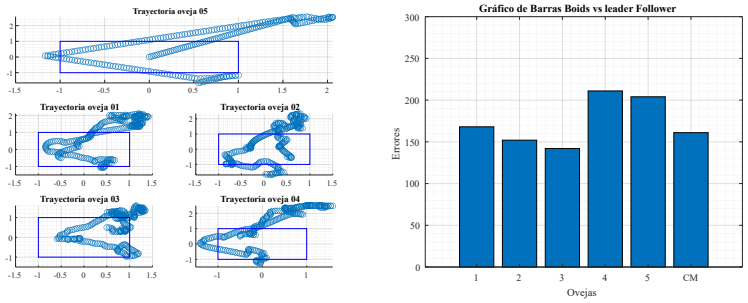

FIGURA 18: Trayectorias Ovejas Modelo LFBM-CI [Autor]

Al realizar un análisis de las trayectorias de las ovejas, es de especial interés observar la manera en que la oveja 05 (*Figura 18*), quien actúa como líder, tiene trayectorias rectas, mientras que el resto del enjambre ajusta su formación para seguirla. Esta estrategia resultó ser más difícil de contener por parte de los pastores, generando un error superior al 58 %.

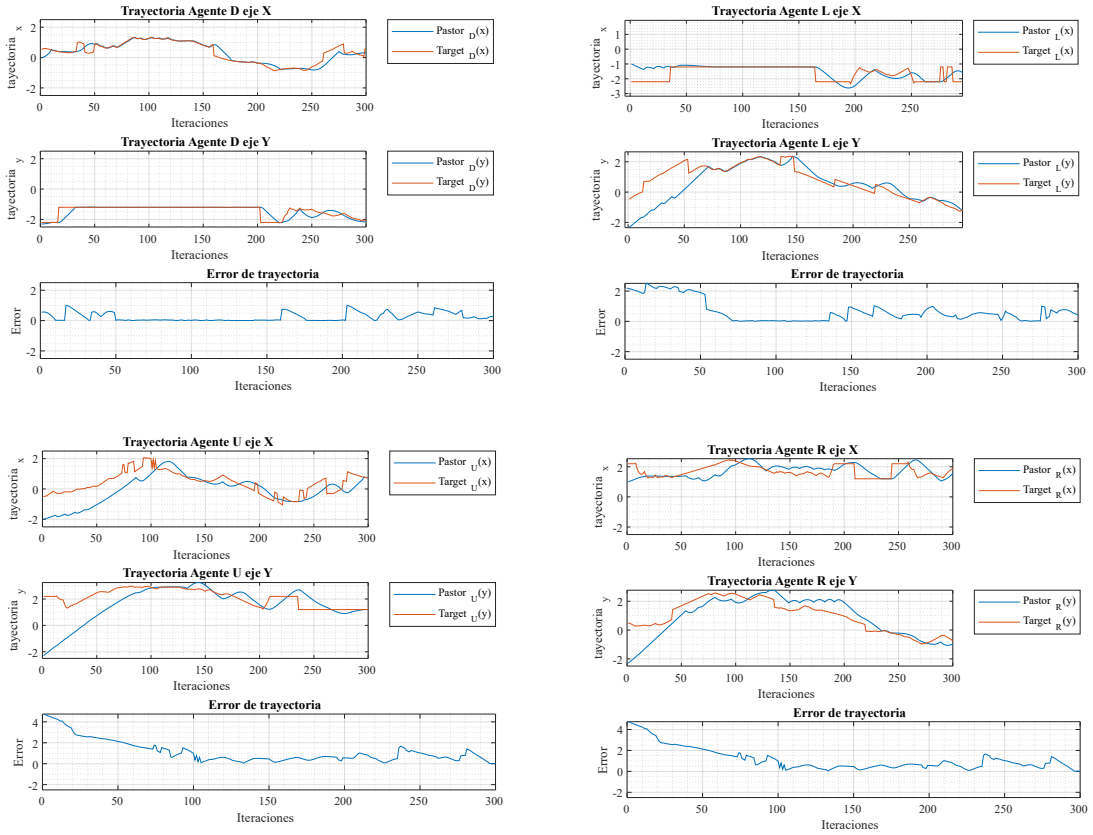

FIGURA 19: Trayectorias Pastores Modelo LFBM-CI [Autor]

Según se puede ver en la *Figura 19*, el pastor más lejano (pastor_U) demora hasta 110 iteraciones para alcanzar el objetivo en las posiciones (x, y). Es notable que el error de los pastores fluctúa sin embargo, en este método, se puede contener el error y mantenerse en 0 una vez los pastores alcanzan sus Targets.

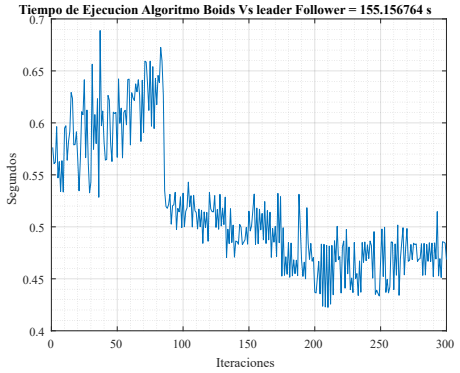

FIGURA 20: Tiempo ejecución Modelo LFBM-CI [Autor]

Midiendo el desempeño del algoritmo BBM-CI en términos de gasto computacional, se realizó la medición del tiempo necesario para completar 300 iteraciones, notando que tomó 154.156 segundos (*Figura 20*), lo que equivale a unos 517 ms por iteración. para este caso, una vez los pastores han alcanzado los *Targets*, el tiempo de ejecución por iteración disminuye.

9.0.3.3. Resultados Modelo BM-CI

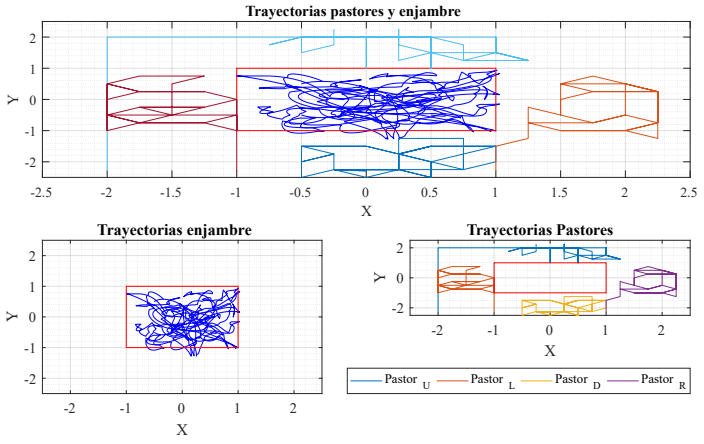

FIGURA 21: Trayectorias Modelo BM-CI [Autor]

Para este modelo (BM-CI)(*Figura 21*), se incorpora el uso de Aprendizaje por Refuerzo junto con el algoritmo Boids para las ovejas, ideado durante el desarrollo de este proyecto. Se puede notar a simple vista que hubo una acción de pastoreo eficiente por parte de los pastores. Las ovejas fueron contenidas dentro del área de interés y los pastores no invadieron la región de interés para evitar generar fuerzas de repulsión que interfieran en la exploración de las ovejas.

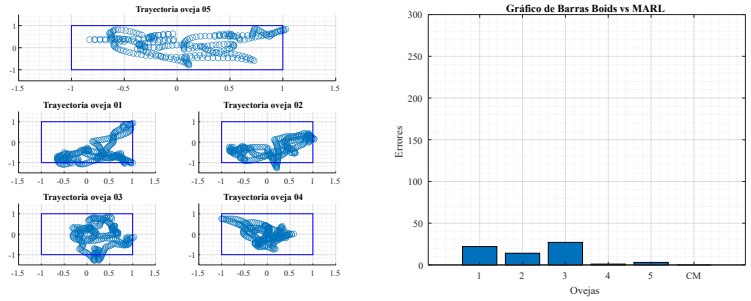

FIGURA 22: Trayectorias Ovejas Modelo BM-CI [Autor]

Al realizar un análisis de las trayectorias de las ovejas, se nota que estas fueron contenidas dentro del área de interés (*Figura 22*), acumulando un error del 4.46 % en 300 iteraciones y en especial el centro de masas logra ser contenido en la región de interés.

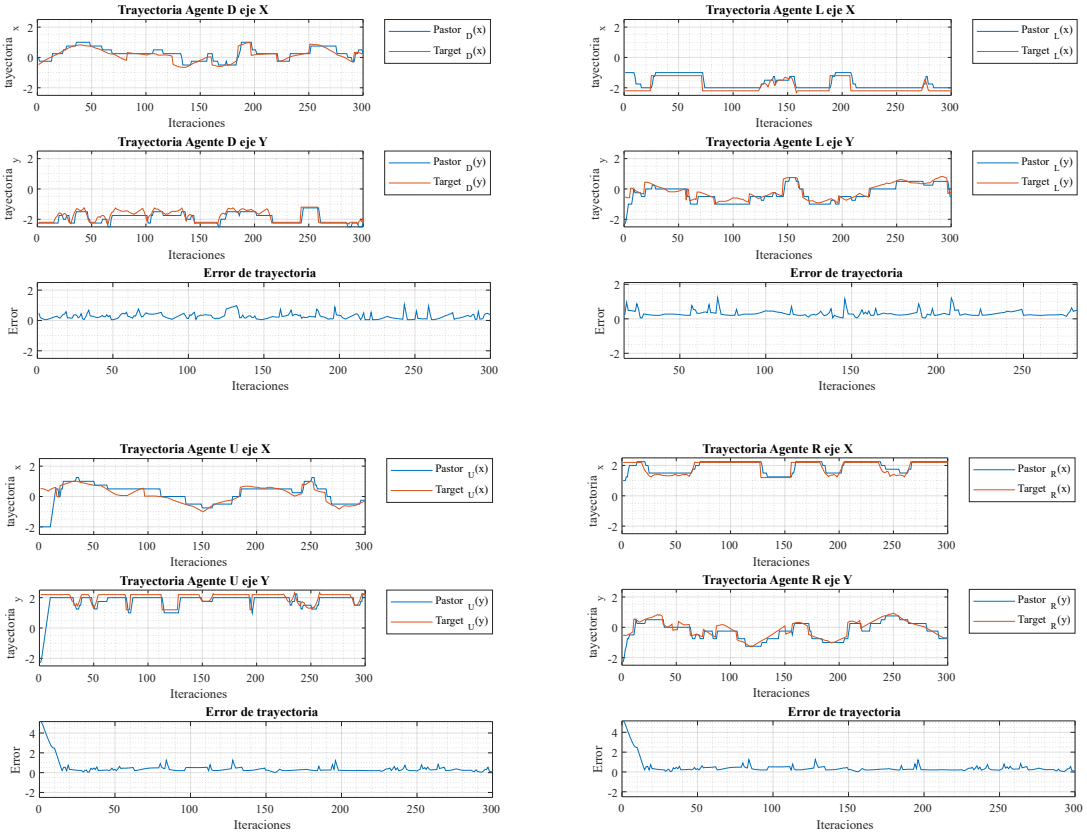

FIGURA 23: Trayectorias Pastores Modelo BM-CI [Autor]

Según se puede ver en la Figura 23, el pastor más lejano (pastor_U) demora solo 11 iteraciones para alcanzar el objetivo en las posiciones (x, y) y el error de posicionamiento para alcanzar los targets se mantiene en 0 durante el desarrollo de la simulación, sin embargo para algunos pastores existe un error de estado estacionario producto del mapeo de sitio por la aplicación de la grilla de exploración.

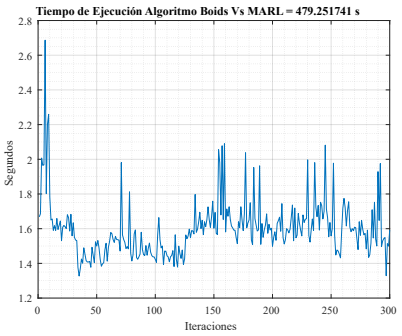

FIGURA 24: Tiempo Ejecución Modelo BM-CI [Autor]

Debido al tiempo de entrenamiento del Q_Learning, este algoritmo tiene un tiempo de ejecución de este método es de 479.25 s llegando a tomar hasta 2 segundos por iteración, 4 veces mayor a las otras versiones probadas anteriormente. Aunque es un tiempo alto en comparación de las versiones BBM-CI y LFBM-CI, su desempeño en el pastoreo compensa el tiempo de ejecución.

9.0.3.4. Resultados Modelo LFM-CI

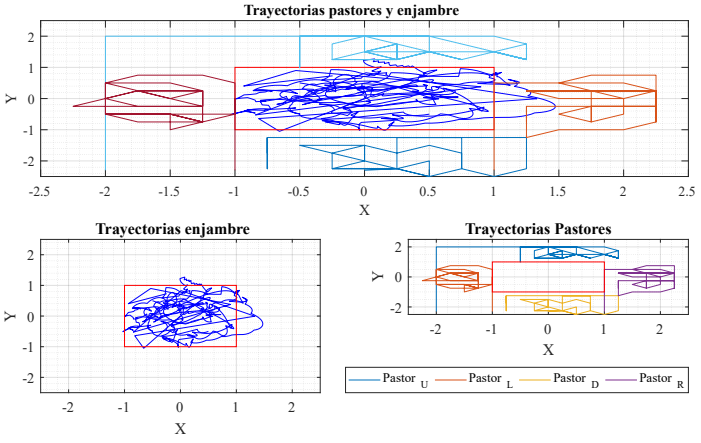

FIGURA 25: Trayectorias Modelo LFM-CI [Autor]

Para este modelo (LFM-CI), también se incorpora el uso de Aprendizaje por Refuerzo junto con el algoritmo leader follower para las ovejas, ideado durante el desarrollo de este proyecto. Se puede notar a simple vista que hubo una acción de pastoreo eficiente por parte de los pastores gracias a la incorporación del Q-learning.

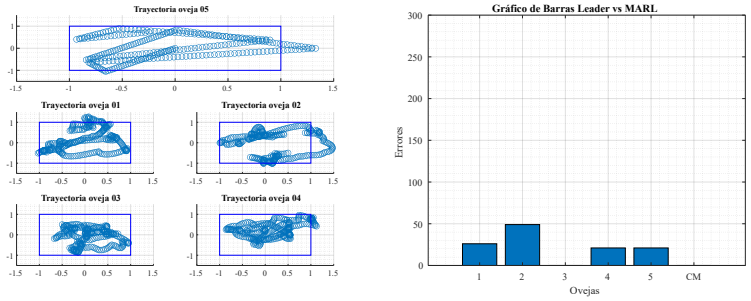

FIGURA 26: Trayectorias Ovejas Modelo LFM-CI [Autor]

Al realizar un análisis de las trayectorias de las ovejas, se nota que estas fueron contenidas dentro del área de interés, acumulando un error del 7.66 % en 300 iteraciones, es de notar que la acción del líder dificulta un poco el pastoreo.

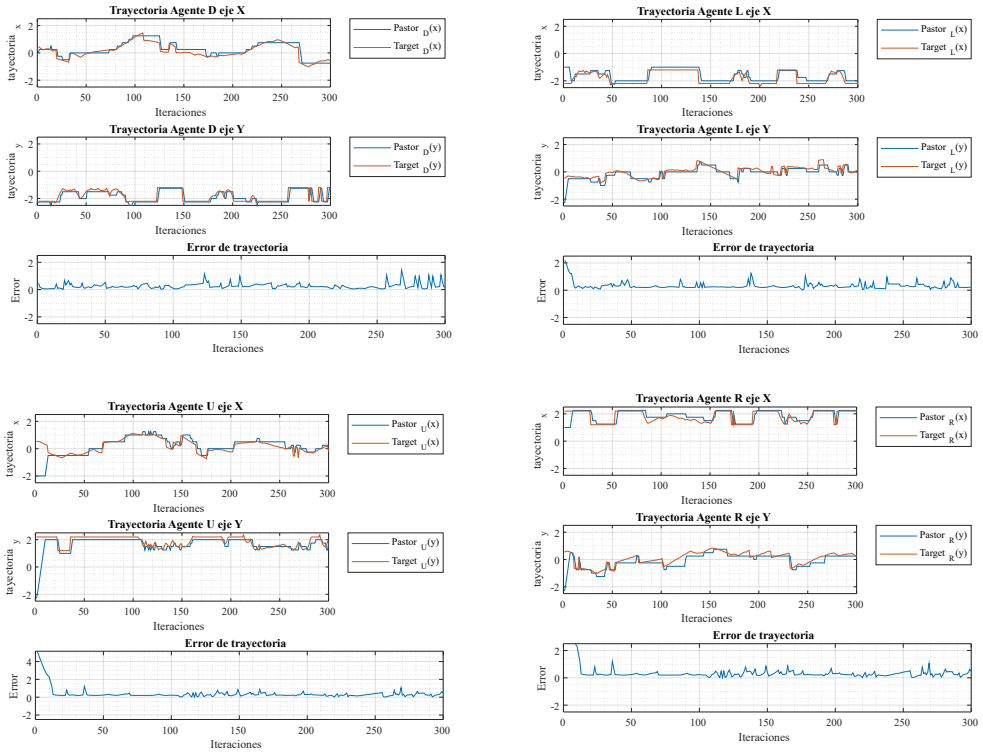

FIGURA 27: Trayectorias Pastores Modelo LFM-CI [Autor]

Según se puede ver en la Figura 27, el pastor más lejano (pastor_U) demora 15 iteraciones para alcanzar el objetivo en las posiciones (x, y) y el error de posicionamiento para alcanzar los *Targets* se mantiene cercano a 0 durante el pastoreo lo que demuestra la buena acción del algoritmo Q-learning.

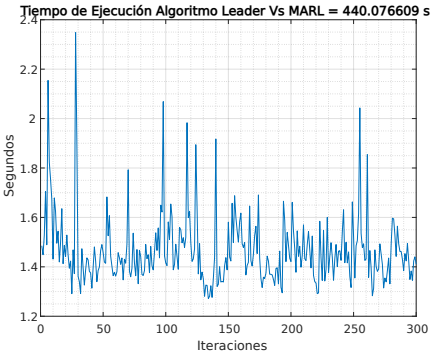

FIGURA 28: Tiempo Ejecución Modelo LFM-CI [Autor]

Debido al tiempo de entrenamiento del Q_Learning, este algoritmo tiene un tiempo de ejecución de método de 440.07 s llegando a tomar hasta 2.5 segundos por iteración, 4 veces mayor a las otras versiones probadas anteriormente y muy similar al BM-CI.

9.0.3.5. Resultados Modelo BBM-OPT

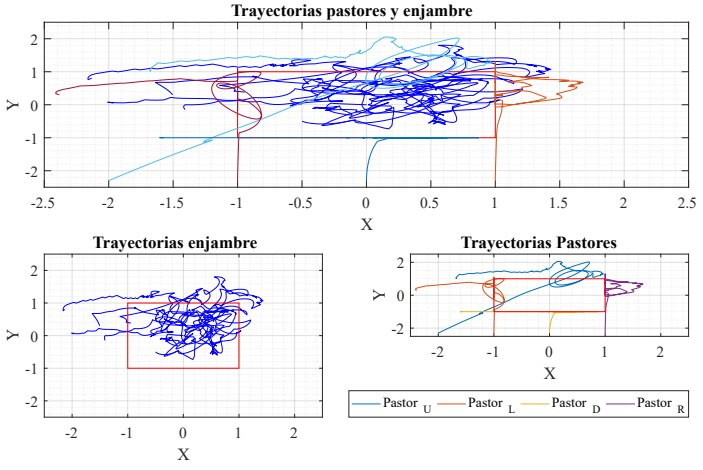

FIGURA 29: Trayectorias Modelo BBM-OPT [Autor]

Según se puede observar en la Figura 29, las trayectorias del algoritmo BBM-OPT muestran que el enjambre intenta ser contenido en la región de interés, con una mejor precisión que la variante BBM-CI.

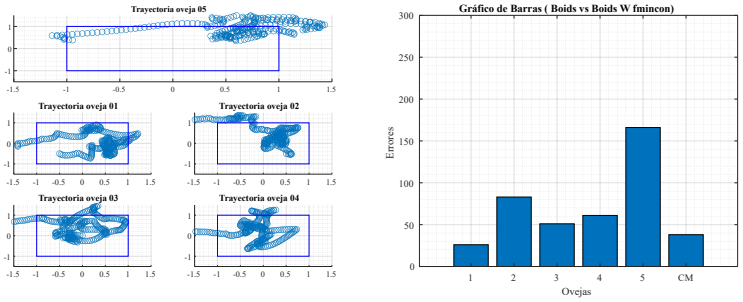

FIGURA 30: Trayectorias Ovejas Modelo BBM-OPT [Autor]

Al realizar un análisis de las trayectorias de las ovejas, se observa que, durante un total de 300 iteraciones, se midió para cada oveja las veces que estuvieron fuera del área de interés, lo que denota un error acumulado del 25 %. Además, hubo momentos en que el centro de masas del enjambre tendía a estar fuera del área de interés. Sin embargo, esto demuestra una mejora en comparación con el 38 % de error que presenta el modelo BBM-CI.

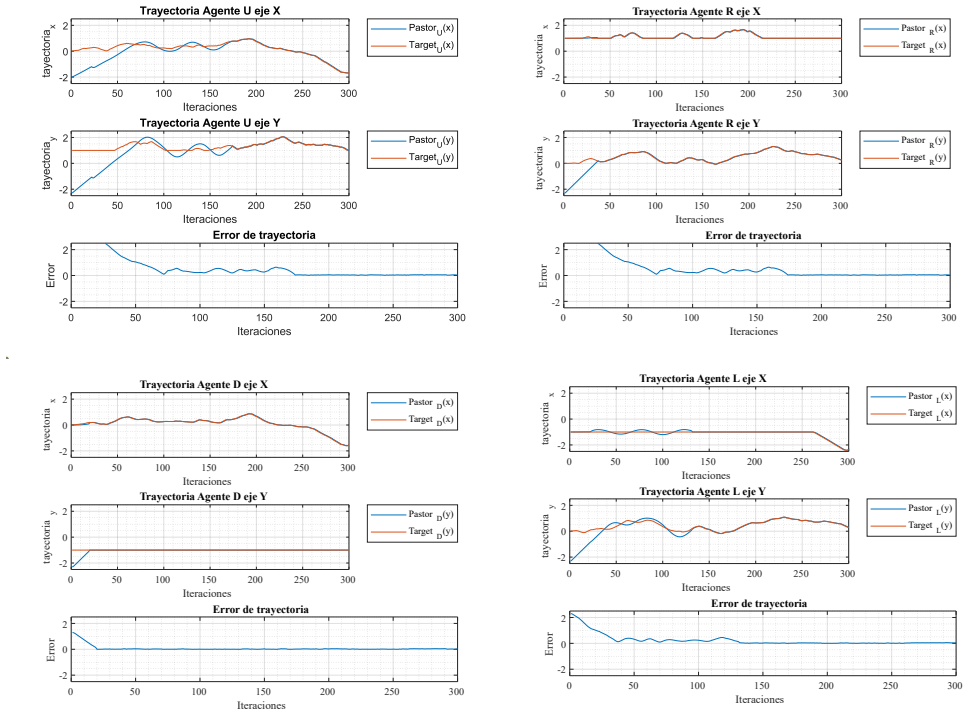

FIGURA 31: Trayectorias Pastores Modelo BBM-OPT [Autor]

Según se puede ver en la Figura 31, el pastor más lejano (pastor_U) demora hasta 170 iteraciones para alcanzar el objetivo en las posiciones (x, y). Sin embargo, el error de los pastores complementarios es 0 durante la simulación. Esto demuestra una gran mejora en el seguimiento de trayectorias de manera global en comparación con BBM-CI.

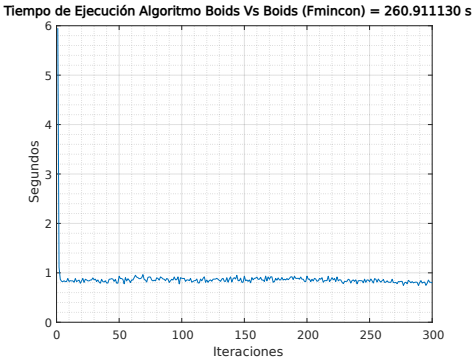

FIGURA 32: Tiempo Ejecución Modelo BBM-OPT [Autor]

Midiendo el desempeño del algoritmo BBM-OPT en términos de gasto computacional, se realizó la medición del tiempo necesario para completar 300 iteraciones, notando que tomó 260.91 segundos, lo que equivale a unos 869 ms por iteración. Este gasto computacional aumenta en comparación al algoritmo BBM-IC debido al calculo de las posiciones óptimas para los *Targets* de los pastores.

9.0.3.6. Resultados Modelo LFBM-OPT

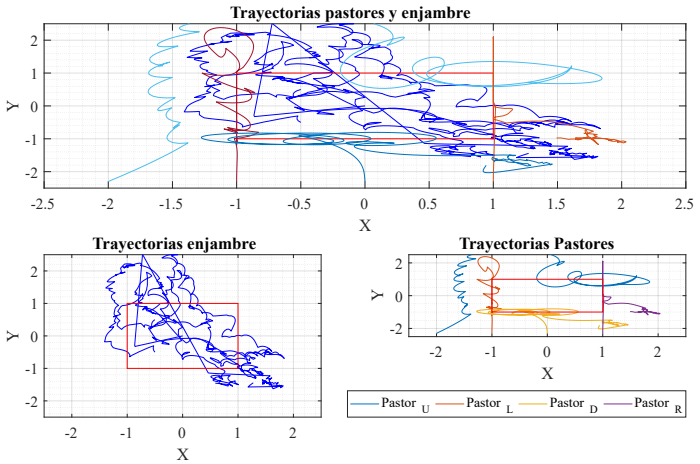

FIGURA 33: Trayectorias Modelo LFBM-OPT [Autor]

Según se puede observar en la Figura 33, las trayectorias del algoritmo LFBM-OPT muestran que el enjambre intenta ser contenido en la región de interés, aunque con poca precisión. En esta ocasión, la fuga se da hacia la parte superior como inferior, que es la región controlada pastor_U y Pastor_D .

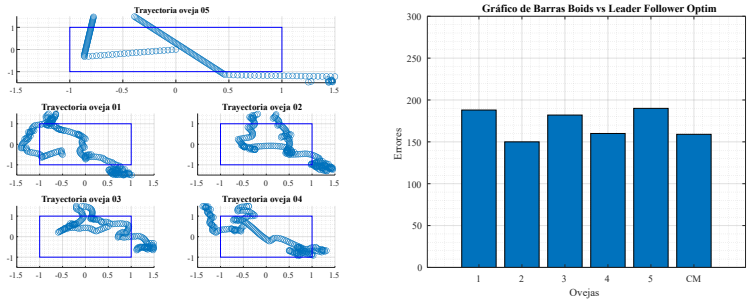

FIGURA 34: Trayectorias Ovejas Modelo LFBM-OPT [Autor]

Al realizar un análisis de las trayectorias de las ovejas, es de especial interés observar la manera en que la oveja 05, quien actúa como líder, tiene trayectorias rectas, mientras que el resto del enjambre ajusta su formación para seguirla. Esta estrategia resultó ser más difícil de contener por parte de los pastores, generando un error superior al 58 %.

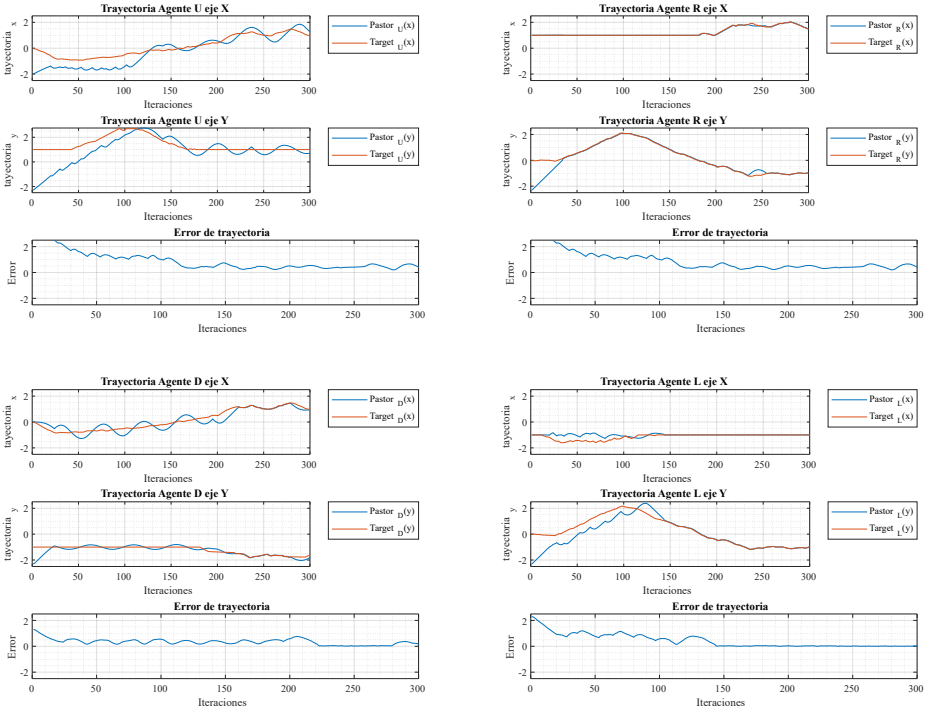

FIGURA 35: Trayectorias Pastores Modelo LFBM-OPT [Autor]

Según se puede ver en la Figura 35, el pastor más lejano (pastor_U) demora mas de 200 iteraciones para alcanzar el objetivo en las posiciones (x, y). Es notable que el error de los pastores fluctúa demasiado en comparación con el método LFBM-IC.

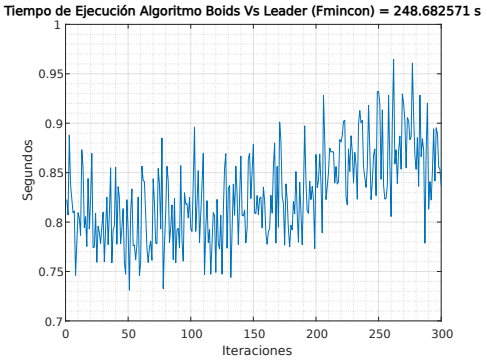

FIGURA 36: Tiempo Ejecución Modelo LFBM-OPT [Autor]

Midiendo el desempeño del algoritmo LFBM-OPT en términos de gasto computacional, se realizó la medición del tiempo necesario para completar 300 iteraciones, notando que tomó 248.68 segundos, lo que equivale a unos 828 ms por iteración.Un tiempo Superior al LFBM-IC .

9.0.3.7. Resultados Modelo BM-OPT

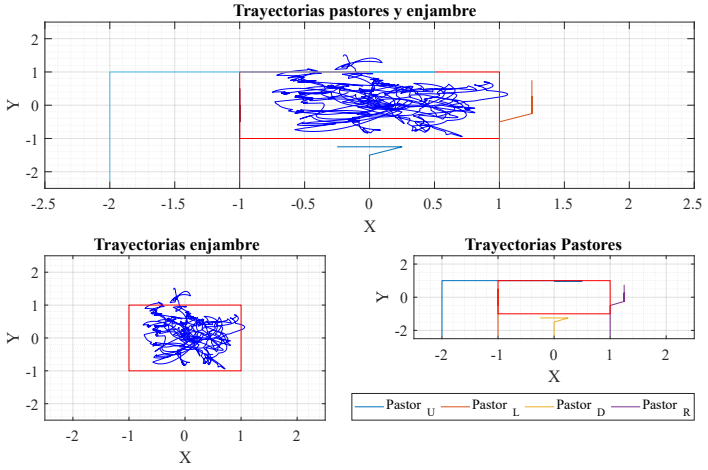

FIGURA 37: Trayectorias Modelo BM-OPT [Autor]

Para este modelo (BM-OPT), se incorpora el uso de Aprendizaje por Refuerzo junto con el algoritmo Boids para las ovejas, ideado durante el desarrollo de este proyecto. Se puede notar a simple vista que hubo una acción de pastoreo eficiente por parte de los pastores. Las ovejas

fueron contenidas dentro del área de interés y los pastores no invadieron la región de interés para evitar generar fuerzas de repulsión que interfieran en la exploración de las ovejas.

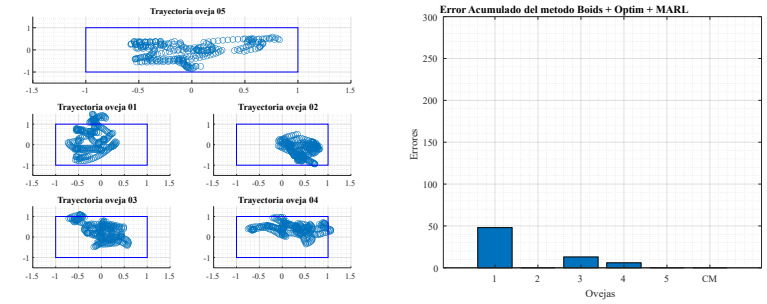

FIGURA 38: Trayectorias Ovejas Modelo BM-OPT [Autor]

Al realizar un análisis de las trayectorias de las ovejas, se nota que estas fueron contenidas dentro del área de interés, acumulando un error del 4.46 % en 300 iteraciones y en especial el centro de masas logra ser contenido en la región de interés.

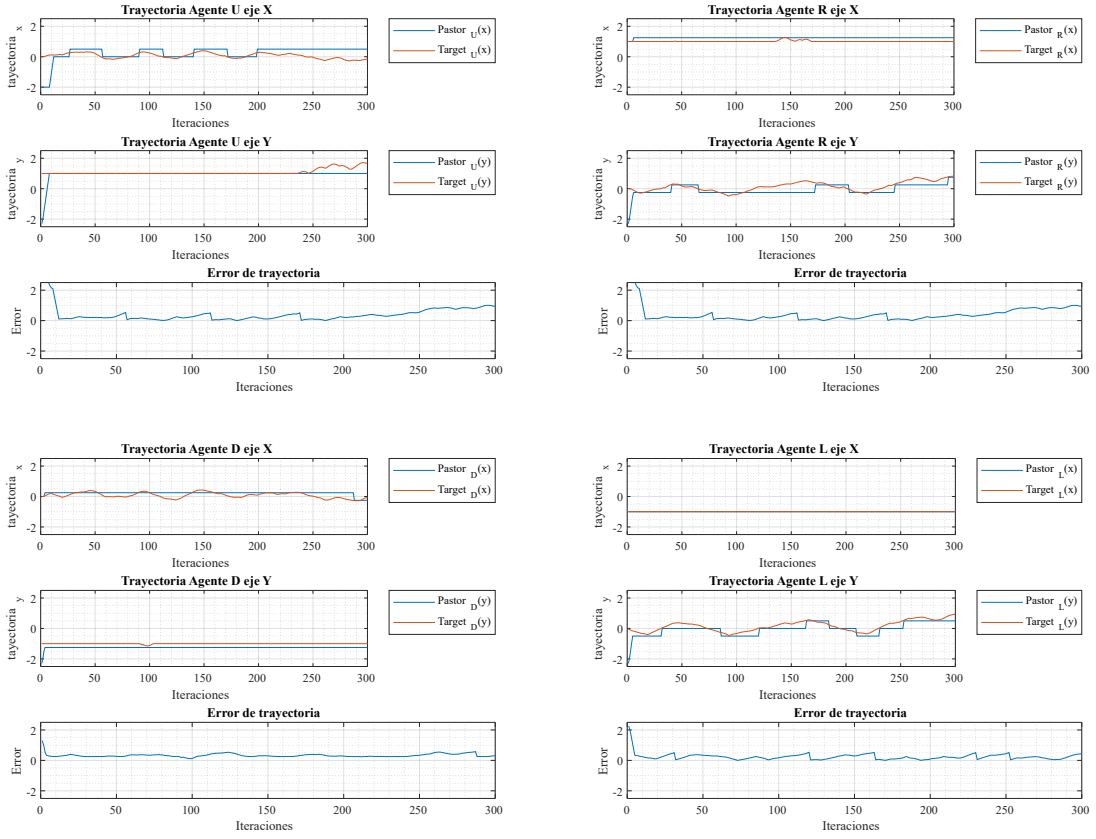

FIGURA 39: Trayectorias Pastores Modelo BM-OPT [Autor]

Según se puede ver en la figura 39, el pastor más lejano (pastor_U) demora solo 12 iteraciones para alcanzar el objetivo en las posiciones (x, y) y el error de posicionamiento para alcanzar los targets se mantiene en 0 durante el desarrollo de la simulación, sin embargo para algunos pastores existe un error de estado estacionario producto del mapeo de sitio por la aplicación de la grilla de exploración.

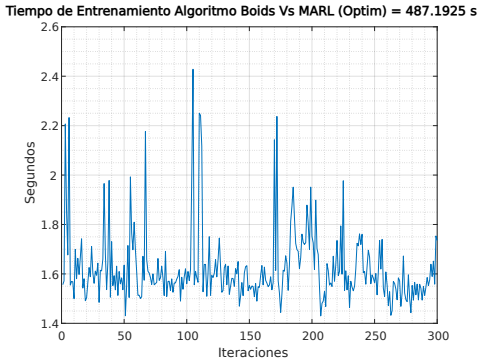

FIGURA 40: Tiempo Ejecución Modelo BM-OPT [Autor]

Debido al tiempo de entrenamiento del Q_Learning y el calculo de posiciones óptimas, el tiempo de ejecución de este método es de 487.19 s llegando a tomar hasta 3 segundos por iteración, 2 veces mayor a las otras versiones probadas anteriormente. Aunque es un tiempo alto en comparación de las versiones BBM-OPT y LFBM-OPT, su desempeño en el pastoreo compensa el tiempo de ejecución.

9.0.3.8. Resultados Modelo LFM-OPT

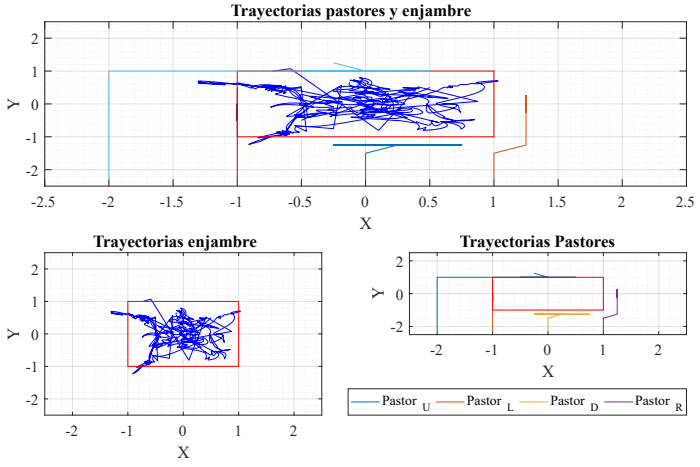

FIGURA 41: Trayectorias Modelo LFM-OPT [Autor]

Para este modelo (LFM-CI), también se incorpora el uso de Aprendizaje por Refuerzo junto con el algoritmo leader follower para las ovejas, ideado durante el desarrollo de este proyecto. Se puede notar a simple vista que hubo una acción de pastoreo eficiente por parte de los pastores gracias a la incorporación del Q-learning.

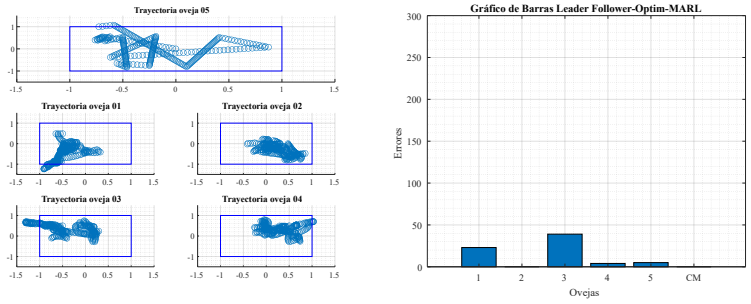

FIGURA 42: Trayectorias Ovejas Modelo LFM-OPT [Autor]

Al realizar un análisis de las trayectorias de las ovejas, se nota que estas fueron contenidas dentro del área de interés, acumulando un error del 4.73 % en 300 iteraciones, es de notar que la acción del líder dificulta un poco el pastoreo pero mejora en comparación al uso del método LFM-CI sin llegar a superar las técnicas BM-CI y BM-OPT.

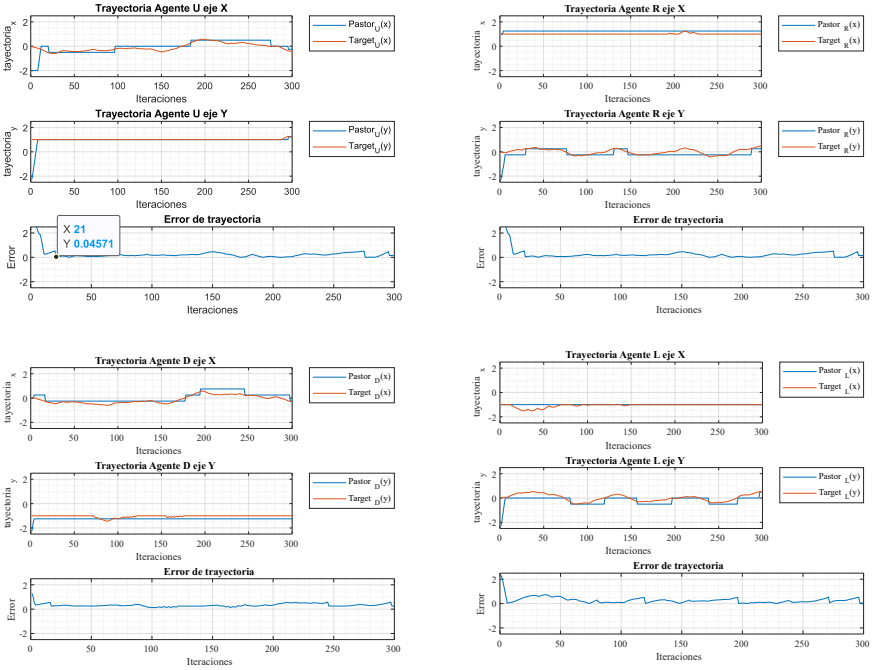

FIGURA 43: Trayectorias Pastores Modelo LFM-OPT [Autor]

Según se puede ver en la Figura 43, el pastor más lejano (pastor_U) demora 21 iteraciones para alcanzar el objetivo en las posiciones (x, y) y el error de posicionamiento para alcanzar los targets se mantiene en cercano a 0 durante el pastoreo lo que demuestra que la buena acción del algoritmo Q-learning.

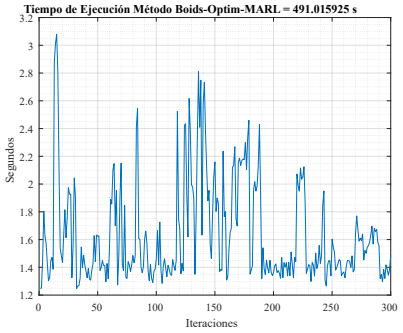

FIGURA 44: Tiempo Ejecución Modelo LFM-OPT [Autor]

Debido al tiempo de entrenamiento del Q_Learning y el calculo de posiciones óptimas, el tiempo de ejecución de este método es de 491.01 s llegando a tomar hasta 3 segundos por iteración, 2 veces mayor a las otras versiones probadas anteriormente. Aunque es un tiempo alto en comparación de las versiones BBM-OPT y LFBM-OPT, su desempeño en el pastoreo compensa el tiempo de ejecución.

9.0.4. Resumen de Índices de Desempeño

En el cuadro 2, se realiza una comparativa de desempeño de los diferentes algoritmos, midiendo su tiempo de ejecución, porcentaje de error acumulado y tiempo requerido para el posicionamiento de los pastores. Se observa que los algoritmos de los modelos BM-CI y BM-OPT muestran un mejor comportamiento en comparación con los otros métodos utilizados.

MODELO	COMPLEJIDAD	Tiempo Ejecución 300 Iteraciones	Error Acumulado	Iteraciones para Pastoreo
Modelo 01	Baja	155.15 s	38.00 %	78
Modelo 02	Baja	157.3 s	58.26 %	110
Modelo 03	Media	479.25 s	4.46 %	11
Modelo 04	Media	440.07 s	7.66 %	15
Modelo 05	Media	260.91 s	25.80 %	174
Modelo 06	Media	248.68 s	58.00 %	>200
Modelo 07	Alta	487.19 s	4.46 %	12
Modelo 08	Alta	491.01 s	4.73 %	21

CUADRO 2: Resultados Comparativa de Algoritmos

Capítulo 10

Conclusiones y Trabajos Futuros

En este proyecto, se evaluaron diversos modelos de pastoreo de enjambres utilizando técnicas de aprendizaje por refuerzo y algoritmos inspirados en el comportamiento de los boids. A lo largo del análisis, se destacaron las siguientes conclusiones principales:

- **Formación Adaptativa del Enjambre:** Los resultados obtenidos demostraron que las ovejas, particularmente aquellas que actúan como líderes, adoptan trayectorias rectas y consistentes, mientras que el resto del enjambre adapta su formación para seguirlas. Esta estrategia presentó desafíos significativos para los pastores, resultando en un error acumulado del 58 % en algunos modelos. Sin embargo, los modelos basados en aprendizaje por refuerzo mostraron mejoras notables en la contención y control del enjambre dentro del área de interés.
- **Cobertura y Contención:** La incorporación del aprendizaje por refuerzo junto con el algoritmo de boids para las ovejas, permitió una acción de pastoreo más eficiente por parte de los pastores. Las ovejas fueron contenidas exitosamente dentro del área de interés, y los pastores evitaron invadir la región de interés para no generar fuerzas de repulsión que pudieran afectar la exploración y formación autónoma del enjambre. Este enfoque redujo el error acumulado a un 4.75 % durante 300 iteraciones, mejorando significativamente en comparación con el modelo BBM-CI, que presentó un 38 % de error y el LFBM-CI con un 58 % de error.
- **Desempeño y Tiempo de Ejecución:** Los modelos que utilizaron técnicas de optimización y posiciones óptimas (BM-CI y BM-OPT) mostraron un mejor comportamiento en términos de tiempo de ejecución y porcentaje de error acumulado. Aunque el tiempo de

ejecución de los métodos basados en Q-Learning fue significativamente mayor (hasta un 75 % superior), el desempeño en términos de pastoreo compensó este incremento. En particular, el modelo LFM-OPT, a pesar de tener un tiempo de ejecución elevado, demostró ser eficiente en la contención y dirección del enjambre.

- **Acción Coordinada de los Pastores:** Los pastores complementarios en los modelos avanzados mostraron un error muy bajo durante las simulaciones, lo que indica una mejora considerable en el seguimiento de trayectorias de manera global. Este resultado subraya la efectividad de las técnicas de aprendizaje por refuerzo y optimización en la coordinación y eficiencia de los pastores para alcanzar sus objetivos.
- **Comparación de Funciones de Pastoreo:** Cuando se utiliza la función cuadrática inversa para el pastoreo, el comportamiento de los pastores es más agresivo en comparación con la función optimizada. Esto mejora la posición de los pastores, pero resulta en más movimientos de los mismos. En cambio, con la función optimizada, los pastores se mueven menos, lo que en una implementación futura podría mejorar el consumo de batería de los pastores.

Para mejorar el desempeño de este algoritmo, se pueden considerar las siguientes técnicas:

- **Improved Q-Learning:** Implementar Improved Q-Learning para disminuir el tiempo de ejecución de la técnica, optimizando el proceso de aprendizaje y reduciendo la cantidad de iteraciones necesarias para alcanzar resultados óptimos.
- **Deep Reinforcement Learning:** Utilizar técnicas de Deep Reinforcement Learning para mejorar la precisión de los objetivos mediante la combinación con redes neuronales. Esto puede ayudar a eliminar errores de estado estacionario y mejorar la precisión durante la acción de pastoreo.
- **Optimización de Posición Adaptativa:** Mejorar la posición adaptativa de las ovejas mediante una función objetivo de optimización orientada a las coordenadas (x,y) de las ovejas y el área de cobertura. Esta técnica podría permitir una distribución más eficiente y efectiva del enjambre dentro del área de interés teniendo en cuenta las posiciones y fuerzas ejercidas por los pastores.

En resumen, la integración de algoritmos de aprendizaje por refuerzo y optimización con modelos basados en el comportamiento de boids ha demostrado ser una estrategia efectiva para

mejorar la formación adaptativa y la cobertura del enjambre en tareas de pastoreo. Estos avances no solo optimizan la eficiencia computacional, sino que también mejoran significativamente la precisión y eficacia del control del enjambre, proporcionando una base sólida para futuras investigaciones y aplicaciones en robótica colaborativa y sistemas multiagente.

Bibliografía

[1] Karthik Elamvazhuthi et al. «Controllability and stabilization for herding a robotic swarm using a leader: A mean-field approach». En: *IEEE Transactions on Robotics* 37.2 (2020), págs. 418-432.

[2] Manuele Brambilla et al. «Swarm robotics: a review from the swarm engineering perspective». En: *Swarm Intelligence* 7.1 (2013), págs. 1-41.

[3] Baptiste Septfons et al. «Swarm Robotics: Moving from Concept to Application». En: *Human Centred Intelligent Systems* (2022), págs. 179-189.

[4] Gustavo A Cardona y Juan M Calderon. «Robot swarm navigation and victim detection using rendezvous consensus in search and rescue operations». En: *Applied Sciences* 9.8 (2019), pág. 1702.

[5] Kendra L.B. Cook. «The silent force multiplier: The history and role of UAVs in warfare». En: 2007. DOI: 10.1109/AERO.2007.352737.

[6] Nathan K Long et al. «A comprehensive review of shepherding as a bio-inspired swarm-robotics guidance approach». En: *IEEE Transactions on Emerging Topics in Computational Intelligence* 4.4 (2020), págs. 523-537.

[7] Yuxin Xie et al. «Adaptive formation tracking control for UAV swarm systems with multiple leaders and switching topologies». En: *2020 Chinese Automation Congress (CAC)*. IEEE. 2020, págs. 5532-5537.

[8] Xiaohui Li et al. «Robotic herding of farm animals using a network of barking aerial drones». En: *Drones* 6.2 (2022), pág. 29.

[9] Jixuan Zhi y Jyh Ming Lien. «Learning to Herd Agents Amongst Obstacles: Training Robust Shepherding Behaviors Using Deep Reinforcement Learning». En: *IEEE Robotics and Automation Letters* 6 (2 2021). ISSN: 23773766. DOI: 10.1109/LRA.2021.3068955.

[10] Aya Hussein et al. «Autonomous Swarm Shepherding Using Curriculum-Based Reinforcement Learning.» En: *AAMAS*. 2022, págs. 633-641.

[11] Saber Elsayed et al. «Path Planning for Shepherding a Swarm in a Cluttered Environment using Differential Evolution». En: *2020 IEEE Symposium Series on Computational Intelligence, SSCI 2020*. 2020, págs. 2194-2201. DOI: 10.1109/SSCI47803.2020.9308572.

[12] Reem E. Mohamed et al. «A Graph-based Approach for Shepherding Swarms with Limited Sensing Range». En: *2021 IEEE Congress on Evolutionary Computation, CEC 2021 - Proceedings*. 2021, págs. 2315-2322. DOI: 10.1109/CEC45853.2021.9504706.

[13] Shuai Zhang y Jia Pan. «Collecting a flock with multiple sub-groups by using multi-robot system». En: *IEEE Robotics and Automation Letters* (2022).

[14] Chen Wang et al. «Revolutionary entrapment model of uniformly distributed swarm robots in morphogenetic formation». En: *Defence Technology* (2022).

[15] Vishnu S. Chipade y Dimitra Panagou. «Multiagent Planning and Control for Swarm Herding in 2-D Obstacle Environments under Bounded Inputs». En: *IEEE Transactions on Robotics* 37 (6 2021). ISSN: 19410468. DOI: 10.1109/TRO.2021.3072026.

[16] Xijing Liu et al. «Constraint-based formation of drone swarms». En: *2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. IEEE. 2022, págs. 73-75.

[17] Mehmet Serdar Guzel et al. «An adaptive pattern formation approach for swarm robots». En: *2017 4th International Conference on Electrical and Electronics Engineering, ICEEE 2017* (2017). DOI: 10.1109/ICEEE2.2017.7935818.

[18] Luigi Feola y Vito Trianni. «Adaptive Strategies for Team Formation in Minimalist Robot Swarms». En: *IEEE Robotics and Automation Letters* 7 (2 2022). ISSN: 23773766. DOI: 10.1109/LRA.2022.3150479.

[19] Sean Lim et al. «Beeground-an open-source simulation platform for large-scale swarm robotics applications». En: *2021 7th International Conference on Automation, Robotics and Applications (ICARA)*. IEEE. 2021, págs. 75-79.

[20] José León et al. «Rendezvous Consensus Algorithm Applied to the Location of Possible Victims in Disaster Zones». En: *International Conference on Artificial Intelligence and Soft Computing*. Springer. 2018, págs. 700-710.

[21] Abdullah Al Redwan Newaz et al. «3D exploration priority based flocking of UAVs». En: *2013 IEEE International Conference on Mechatronics and Automation*. IEEE. 2013, págs. 1534-1539.

[22] S Rooban, M Javaraiu, P Prem Sagar et al. «A Detailed Review of Swarm Robotics and its Significance». En: *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*. IEEE. 2022, págs. 797-802.

[23] Jie Li y Ying Tan. «A probabilistic finite state machine based strategy for multi-target search using swarm robotics». En: *Applied Soft Computing* 77 (2019), págs. 467-483.

[24] Tulika Dutta et al. «Border collie optimization». En: *IEEE Access* 8 (2020), págs. 109177-109197.

[25] Frank L Lewis et al. *Cooperative control of multiagent systems: optimal and adaptive design approaches*. Springer Science & Business Media, 2013.

[26] Ivana Strumberger et al. «Static drone placement by elephant herding optimization algorithm». En: *2017 25th Telecommunication Forum (Telfor)*. IEEE. 2017, págs. 1-4.

[27] Mireia Sempere Tortosa. «Agentes y enjambres artificiales: modelado y comportamientos para sistemas de enjambre robóticos». Tesis doct. Universitat d'Alacant-Universidad de Alicante, 2014.

[28] Daniel Strömbom et al. «Solving the shepherding problem: heuristics for herding autonomous, interacting agents». En: *Journal of the royal society interface* 11.100 (2014), pág. 20140719.

[29] Zhiyong Tan, Tao Wang y Yao Yu. «Mobile robot navigation method based on improved Q-learning algorithm». En: *2021 36th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. IEEE. 2021, págs. 184-188.

[30] Tsutomu Miki y Tetsuya Nakamura. «An effective simple shepherding algorithm suitable for implementation to a multi-mmobile robot system». En: *First International Conference on Innovative Computing, Information and Control-Volume I (ICICIC'06)*. Vol. 3. IEEE. 2006, págs. 161-165.

[31] Timmie Wong. *Boids Stanford University*. Sep 2008. URL: https://cs.stanford.edu/people/eroberts/courses/soco/projects/2008-09/modeling-natural-systems/boids.html#:~:text=Boids&text=Boids%20is%20an%20artificial%20life,behavior%20of%20each%20individual%20bird.

[32] Craig Reynolds. *Boids Background and Update*. July 30, 2007. URL: https://www.red3d.com/cwr/boids/.

[33] CARL-OSCAR Erneholm. «Simulation of the flocking behavior of birds with the boids algorithm». En: *Royal Institute of Technology* (2011).

[34] Inc. The MathWorks. *Optimization Toolbox User's Guide*. MathWorks, 2024.

[35] Richard S Sutton y Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[36] Kaiqing Zhang, Zhuoran Yang y Tamer Başar. «Multi-agent reinforcement learning: A selective overview of theories and algorithms». En: *Handbook of reinforcement learning and control* (2021), págs. 321-384.

[37] MathWorks. *An Introduction to Multi-Agent Reinforcement Learning*. 2022. URL: https://la.mathworks.com/videos/an-introduction-to-multi-agent-reinforcement-learning-1657699091457.html.

[38] Lukas M Schmidt et al. «An introduction to multi-agent reinforcement learning and review of its application to autonomous mobility». En: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2022, págs. 1342-1349.

Apéndice A

Algorithm 2 Algoritmo Boids Modificado Para Pastoreo

Input: num_iterations, num_boids, pos_boids, velocities, cohesion_factor, separation_factor, alignment_factor, shepherds, shepherds_avoidance_factor, shepherd_approach_distance, obstacles_avoid, obstacles_avoid_factor, obstacle_radius, obstacles
for $t = 1$ **to** num_iterations **do**
  $center\_of\_mass \leftarrow \text{mean}(pos\_boids)$
  for $i = 1$ **to** num_boids **do**
    {**Regla de cohesión**}
    $cohesion \leftarrow (center\_of\_mass - pos\_boids[i]) \times cohesion\_factor$
    {**Regla de separación**}
    $separation \leftarrow \mathbf{0}$
    for $j = 1$ **to** num_boids **do**
      if $i \neq j$ **then**
        $dist \leftarrow \text{norm}(pos\_boids[i] - pos\_boids[j])$
        $separation \leftarrow separation - \frac{pos\_boids[j] - pos\_boids[i]}{dist^2}$
      end if
    end for
    $separation \leftarrow separation \times separation\_factor$
    {**Regla de alineación**}
    $alignment \leftarrow \text{mean}(velocities) \times alignment\_factor$
    {**Regla de evasión de Pastores**}
    $shepherds\_avoid \leftarrow \mathbf{0}$
    for $j = 1$ **to** num_shepherds **do**
      if $i \neq j$ **then**
        $dist\_to\_Shepherd \leftarrow \text{norm}(pos\_boids[i] - shepherds[j])$
        if $dist\_to\_Shepherd \leq shepherd\_approach\_distance$ **then**
          $shepherds\_avoid \leftarrow shepherds\_avoid - \frac{pos\_boids[i] - shepherds[j]}{dist\_to\_Shepherd^2}$
        end if
      end if
    end for
    $shepherds\_avoid \leftarrow shepherds\_avoid \times shepherds\_avoidance\_factor$
    {**Regla de evasión de Obstáculos**}
    $obstacles\_avoid \leftarrow \mathbf{0}$
    for $j = 1$ **to** obstacles **do**
      $dist\_to\_obstacles \leftarrow \text{norm}(pos\_boids[i] - obstacles[j])$
      if $dist\_to\_obstacles \leq obstacle\_radius$ **then**
        $obstacles\_avoid \leftarrow obstacles\_avoid - \frac{pos\_boids[i] - obstacles[j]}{dist\_to\_obstacles^2}$
      end if
    end for
    $obstacles\_avoid \leftarrow obstacles\_avoid \times obstacles\_factor$
    {**Actualizar velocidad**}
    $velocities[i] \leftarrow velocities[i] + cohesion + separation + alignment + shepherds\_avoid + obstacles\_avoid$
    {**Actualizar posición**}
    $pos\_boids[i] \leftarrow pos\_boids[i] + velocities[i]$
  end for
end for

Apéndice B

Algorithm 3 Q-Learning para Pastoreo

{**generar Matriz de Recompensas**}
$R\_Matrix \leftarrow$ **GenerateMatrizR**$(neg\_reward, Target, rows, cols, states, actions)$
$disp\_warning\_flag \leftarrow 0$
{**Evitar Entrenamiento Infinito o Posición no alcanzable**}
if $Target = Feasible$ **then**
    {**generar Matriz de calidad Q**}
    Q $\leftarrow rand(states, actions)$
    {**Entrenamiento del Agente política Epsilon-greedy**}
    for $stage = 1$ **to** max_stages **do**
        $s \leftarrow$ **rand**$(states)$
        while $stage \neq Target$ **do**
            if $rand() \leq \epsilon$ **then**
                $a \leftarrow$ **rand**$(actions)$
            else
                $a \leftarrow$ **argmax**(Q, s)
            end if
            $R \leftarrow$ **R_Matrix**(s, a)
            $Q(s, a) \leftarrow Q(s, a) + \alpha * [R + \gamma * ($**argmax**$(Q(next\_s, next\_a) - Q(s, a))]$
            {**Evitar Entrenamiento Infinito**}
            $training\_counter \leftarrow training\_counter + 1$
            if $training\_counter \geq Max\_training\_counter$ **then**
                $disp\_warning\_flag \leftarrow 1$
                $break\_while$
            end if
        end while
    end for
end if