UNIVERSIDAD DE LA REPÚBLICA URUGUAY

FACULTAD DE INGENIERÍA

Alternativas multi-dron para el pastoreo de bandadas

Informe de Proyecto de Grado presentado por

Camilo Aramburu, Santiago Cancela y Maximiliano Anselmi

en cumplimiento parcial de los requerimientos para la graduación de la carrera de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de la República

Supervisor

Facundo Benavides

Montevideo, 4 de junio de 2026

Agradecimientos

A nuestras familias y amigos, por ser un soporte a lo largo de este camino.

A nuestro tutor, Facundo Benavides, por su guía, apoyo y experiencia durante el desarrollo de este trabajo.

A la Dra. Matilde Alfaro, por sus conocimientos sobre comportamientos de aves que fueron de gran ayuda para el proyecto.

Resumen

El presente proyecto aborda la problemática de la protección de cultivos frutales frente al daño ocasionado por aves plaga, particularmente la especie *Myiopsitta monachus* (cotorra argentina). Con el objetivo de maximizar el aprovechamiento de los cultivos y superar la limitada eficacia de las técnicas tradicionales, se investiga la viabilidad de una solución tecnológica destinada a detectar y expulsar aves mediante una flota coordinada de drones. En esta instancia, el trabajo se centra en el desarrollo del módulo encargado del pastoreo de la bandada, es decir, la programación del comportamiento colectivo de la flota.

El sistema desarrollado se estructura en diversos módulos interconectados que interactúan para recrear el escenario descrito en un entorno simulado, dichos módulos serán detallados en el capítulo 3. El módulo *Controlador del dron* implementa diversos algoritmos de pastoreo de aves y se comunica con el software de control de vuelo PX4 para enviar instrucciones de movimiento a cada dron. El módulo *Modelo de comportamiento de bandada* utiliza el algoritmo de *boids* (sigla en inglés de "bird-oid object", objeto con comportamiento de ave) para simular el comportamiento grupal de las aves mediante un modelo teórico. El módulo *Interfaz de simulación* gestiona la comunicación con el entorno de pruebas de Gazebo Garden, el software de simulación utilizado. Todos los módulos se desarrollan bajo el marco de trabajo ROS 2 (Robot Operating System 2).

Las soluciones desarrolladas para implementar la lógica de pastoreo son extensiones multi-agente de un algoritmo altamente utilizado en este campo de investigación, el algoritmo de Strömbom. Partiendo de él, se aplican diversas técnicas para extender su funcionamiento a una flota coordinada de drones. Mediante el análisis teórico y la experimentación se presentan fortalezas y debilidades de la aplicabilidad en un entorno real de cada una de estas alternativas.

Para evaluar las soluciones se diseñó un conjunto de experimentos simulados que analizan la efectividad del algoritmo de pastoreo frente a distintas configuraciones del entorno como la cantidad de aves, el tamaño de la flota de drones y la distribución de los cultivos. Las etapas de experimentación comprenden: un ajuste paramétrico para determinar la configuración óptima de cada solución, la evaluación de las soluciones en distintos escenarios y un análisis de sensibilidad del sistema para evaluar la influencia de ciertos parámetros relativos al modelo de bandada. Entre las métricas consideradas se incluyen el tiempo de simulación hasta la expulsión exitosa de la bandada, la distancia recorrida por los drones,

la dispersión de las aves y la proporción de éxito.

Finalmente, se identifican posibles mejoras, como el cálculo dinámico de ciertos parámetros críticos y la adición de estrategias de vuelo adecuadas para un entorno real. En conjunto, el proyecto demuestra la factibilidad de aplicar técnicas de robótica multi-agente para la protección de cultivos agrícolas mediante el control coordinado de drones autónomos.

Palabras clave: Algoritmo de pastoreo, comportamiento de bandadas, robótica de enjambre, UAV, protección de cultivos, ROS 2, Gazebo, PX4.

Índice general

1. Introducción **1**

2. Revisión de antecedentes **4**
2.1. Control de aves plaga en la agricultura 4
2.2. Técnicas de pastoreo mediante drones 5
2.2.1. Técnicas preventivas y conductivas 6
2.2.2. Aprendizaje por refuerzo profundo 6
2.3. Modelos de comportamiento de bandadas 7
2.4. Métricas para evaluar el pastoreo 7
2.5. Herramientas y entornos tecnológicos 8
2.6. Síntesis de los antecedentes 9

3. Desarrollo **10**
3.1. Especificación del problema 10
3.1.1. Entidades 10
3.1.2. Asunciones y requerimientos 11
3.1.3. Objetivos 12
3.2. Módulos del sistema 12
3.3. Comportamiento de bandada 14
3.3.1. Diseño del modelo 15
3.3.2. Extensiones sobre el modelo 17
3.3.3. Especificación del modelo 19
3.3.4. Emergencia de comportamiento colectivo 25
3.4. Controlador de dron 25
3.4.1. Etapas de la misión de vuelo 26
3.4.2. Diseño general de la solución 27
3.4.3. Algoritmo de Strömbom 30
3.4.4. Solución 1: Clustered Strömbom 32
3.4.5. Solución 2: M-Strömbom 36
3.4.6. Solución 3: Clustered M-Strömbom 39
3.4.7. Dispersión de puntos de pastoreo 42
3.4.8. Asignación de objetivos 43
3.4.9. Cálculo de velocidad objetivo 43
3.4.10. Observaciones sobre complejidad computacional 44

4. Experimentación **47**
4.1. Entorno de pruebas 47
4.2. Escenarios y métricas 48
4.2.1. Escenario base 48
4.2.2. Condiciones de parada 49
4.2.3. Métricas de evaluación 49
4.3. Parámetros utilizados 50
4.3.1. Parámetros del controlador de dron 50
4.3.2. Parámetros del modelo de bandada 51
4.4. Ajuste de parámetros 51
4.4.1. Clustered Strömbom 52
4.4.2. M-Strömbom 54
4.4.3. Clustered M-Strömbom 55
4.4.4. Resultados del ajuste paramétrico 57
4.5. Evaluación de las soluciones 57
4.5.1. Solución para un agente 58
4.5.2. Clustered Strömbom 59
4.5.3. M-Strömbom 61
4.5.4. Clustered M-Strömbom 63
4.5.5. Comentarios generales 64
4.6. Análisis de sensibilidad 66
4.6.1. Peso de fuerza de alineamiento 67
4.6.2. Peso de fuerza de cohesión 68
4.6.3. Peso de fuerza de separación 70
4.6.4. Peso de fuerza de atracción a objetivo 71
4.6.5. Radio de percepción 72
4.6.6. Resultados del análisis de sensibilidad 73
4.7. Conclusiones preliminares 73

5. Conclusiones **75**
5.1. Trabajo a Futuro 76
5.1.1. Optimización algorítmica 76
5.1.2. Consideraciones para la implantación del sistema 77

Referencias **79**

Anexo A **82**
A. Glosario 82
A.1. Paquete 82
A.2. Nodos 82
A.3. Tópicos 82
A.4. Agentes 82
A.5. Clusters 82
A.6. Centroides 83
A.7. PCA (Principal Component Analysis) 83
A.8. BIC (Bayesian Information Criterion) 83

A.9. Sistema distribuido 83
A.10. Sistema centralizado 83
A.11. DDS (Data Distribution Service) 83
A.12. Firmware 84
A.13. Framework 84
A.14. Middleware 84

Anexo B **85**
B. Tecnologías utilizadas 85
B.1. ROS 2 (Robot Operating System 2) 85
B.2. Gazebo 85
B.3. C++ 85
B.4. Python 85
B.5. PX4 85
B.6. rqt_graph 86
B.7. MAVSDK 86
B.8. MicroXRCEAgent 86
B.9. QGroundControl 86

Capítulo 1

Introducción

La agricultura constituye una de las actividades económicas más importantes y estratégicas para el desarrollo de los países, ya que garantiza la seguridad alimentaria y genera una parte sustancial del empleo rural. En el contexto uruguayo, los cultivos frutales representan una parte significativa del PBI del país, aportando unos 1.500 millones de dólares al año y generando un total de 3,6 % del empleo entre trabajadores de la producción, el transporte, el empaque, el comercio mayorista y minorista, así como de la industria derivada. Estos valores son comparables con los de otros sectores relevantes en la actividad económica del país como lo es la producción de carne bovina (2.000 millones) y la soja (1.200 millones) (MGAP, 2024).

Figura 1.1: La cotorra argentina (*Myiopsitta monachus*).

Sin embargo, en los últimos años se ha observado un aumento considerable en el daño ocasionado por aves plaga (MGAP, 2021) alcanzando pérdidas valuadas en 11 millones de dólares (INIA, 2025). La principal especie de ave a la que se le atribuyen estos daños es la *Myiopsitta monachus* (figura 1.1), comúnmente conocida como cotorra argentina. Estas aves, al alimentarse de frutas y brotes, afectan la productividad y la sostenibilidad de los cultivos frutales.

Los métodos tradicionales de control, tales como espantapájaros, redes protectoras o dispositivos acústicos, han demostrado una eficacia limitada (King et al., 2023; Olivera & Rodriguez, 2018). De forma similar, los métodos letales como la aplicación de venenos, la destrucción de los nidos y el uso de adhesivos han generado oposición por parte de organizaciones ambientalistas (MercoPress, 2025), además de implicar altos costos de mantenimiento y afectar negativamente el entorno natural. Frente a esta problemática, surge la necesidad de explorar alternativas tecnológicas más eficientes, sostenibles y escalables.

En este contexto, el presente trabajo propone el desarrollo de un sistema automatizado de disuasión de aves plaga mediante la utilización de una flota cooperativa de drones que ahuyente a las bandadas de cotorras cuando se detecte su presencia en la zona de cultivos. La motivación central radica en investigar la factibilidad de aplicar principios de robótica multi-agente y simulación del comportamiento animal para crear una herramienta adaptable y económicamente viable para el sector agrícola uruguayo.

El objetivo general de este proyecto radica en desarrollar y evaluar la viabilidad del sistema descrito. Este trabajo se enfoca en el desarrollo del módulo de comportamiento de drones, dejando por fuera del alcance la lógica relativa a la detección de las aves. De forma similar, la evaluación del sistema se lleva a cabo en un entorno de simulación, las pruebas en entornos reales se dejan para instancias futuras. Más allá de esto, se tiene en cuenta las limitaciones y discrepancias que pueden existir entre el sistema desarrollado en esta instancia y su versión final implantada en un cultivo real. Para esto, se desarrollan varias consideraciones relativas al diseño de la solución y su evaluación, además de incluir sugerencias y análisis sobre los desafíos que pueden aparecer en etapas futuras.

De lo descrito, se derivan los siguientes objetivos específicos:

- 1) Analizar la viabilidad técnica de mecanismos de disuasión basados en drones.
- 2) Explorar estrategias de pastoreo y dispersión de aves inspiradas en comportamientos cooperativos multi-agente.
- 3) Experimentar con un entorno simulado que reproduzca las condiciones reales de un cultivo frutal, permitiendo validar distintos enfoques de control.

Se espera como resultado que el sistema propuesto sea capaz de gestionar un conjunto representativo de situaciones, basadas en datos verídicos de los cultivos de Uruguay y del comportamiento de las bandadas de cotorras. De esta manera se desea demostrar la efectividad del enfoque multi-agente sin la necesidad de abarcar todas las posibles configuraciones del entorno.

En términos generales, las conclusiones del proyecto demuestran la factibilidad de implementar estrategias de disuasión automatizadas basadas en cooperación entre agentes, estableciendo los cimientos para un sistema extensible a escenarios reales. De las soluciones evaluadas, se considera que todas presentan

una eficacia adecuada en distintos escenarios pero presentan diferencias en cuanto a sus requerimientos operacionales. Por otro lado, se detallan las limitaciones técnicas que deben abordarse en desarrollos futuros y se presentan consideraciones para la implantación del sistema. Finalmente, se incorpora como líneas de trabajo futuro la evaluación del impacto potencial sobre el ecosistema del cultivo y la consideración de aspectos éticos asociados al manejo no letal de fauna silvestre, dado que el presente proyecto se desarrolla íntegramente en un entorno simulado.

El presente documento se organiza de la siguiente manera:

- Capítulo 2: Presenta una revisión de antecedentes relevantes, abarcando estudios previos sobre comportamiento de aves, control de plagas y tecnologías aplicadas en robótica agrícola.
- Capítulo 3: Desarrolla la parte central del trabajo, donde se describen las distintas soluciones desarrolladas para implementar el proceso de pastoreo con una flota de drones y el algoritmo de simulación de aves.
- Capítulo 4: Se detallan los experimentos realizados, las métricas empleadas y los resultados obtenidos.
- Capítulo 5: Resume las conclusiones alcanzadas y propone posibles líneas de trabajo futuro.

Capítulo 2

Revisión de antecedentes

La revisión de antecedentes presentada en este capítulo tiene como propósito contextualizar el proyecto dentro del marco de las investigaciones y desarrollos existentes sobre el uso de vehículos aéreos no tripulados para el control de aves y el pastoreo de agentes autónomos. También se busca justificar la pertinencia de este enfoque, ahondando en los problemas presentados por los métodos de control tradicionales y en las ventajas que promete la aplicación de un sistema robótico. Por otro lado, se introducen los principales conceptos tecnológicos que sustentan la implementación de la solución propuesta, incluyendo el uso de ROS 2, PX4 y Gazebo Garden. La información que aquí se expone se basa en estudios previos de carácter académico y en la revisión de herramientas y plataformas de software ampliamente utilizadas en el campo de la robótica.

Para el desarrollo de esta revisión se siguieron las pautas y recomendaciones presentadas en (Okoli, 2015). En particular, se establecieron protocolos para la búsqueda y lectura de material bibliográfico, se crearon formularios para la extracción de datos relevantes y se elaboró una síntesis bibliográfica a partir de los resultados obtenidos.

2.1. Control de aves plaga en la agricultura

El daño ocasionado por aves plaga, como la cotorra argentina (*Myiopsitta monachus*), representa un problema persistente para la agricultura en diversas regiones de Sudamérica. Estas aves se alimentan de frutas y brotes tiernos, generando pérdidas significativas en cultivos frutales. Tradicionalmente, los métodos empleados para mitigar el impacto de aves plaga han sido rudimentarios o de eficacia limitada.

Las principales desventajas de estos métodos varían dependiendo del caso. Un ejemplo de esto es el uso de aves rapaces entrenadas, este método implica altos costos de mantenimiento ya que se debe emplear personal capacitado para entrenarlas y cuidar de ellas. También se han utilizado dispositivos acústicos como cañones de gas propano o bioacústica (reproducción de sonidos de aves

reales con el fin de asustar o desorientar), la principal desventaja de estos métodos radica en su pérdida de eficacia a medida que las aves se acostumbran a los sonidos. Otro ejemplo es el uso de repelentes químicos que causan efectos fisiológicos adversos en las cotorras luego de consumir la fruta, este método implica desperdiciar una parte de los cultivos para que resulte efectivo. Por último, un método ampliamente utilizado es el de las mallas antipájaro, donde se cubre el área de la parcela con una red de forma de prevenir completamente el acceso de las aves, este método presenta costos de mantenimiento significativos en especial para cultivos de gran superficie (Olivera & Rodriguez, 2018).

Con esto en mente, el uso de drones como método de control de aves plaga resulta una alternativa prometedora, ya que las desventajas mencionadas anteriormente podrían ser mitigadas o completamente eliminadas con el sistema adecuado. Además, la incorporación de tecnologías automatizadas ha despertado interés en el ámbito agrícola en los últimos años, particularmente con el uso de drones equipados con sensores o actuadores destinados a la vigilancia, observación y protección de cultivos (Radoglou-Grammatikis et al., 2020).

Hasta la fecha, el uso de drones como mecanismo de disuasión de aves ha sido explorado en contextos específicos, como la protección de aeropuertos frente a aves (Paranjape et al., 2018). Dichos trabajos han demostrado que los drones pueden alterar el comportamiento de las bandadas de manera controlada, reduciendo los riesgos de colisión y daños materiales. Sin embargo, las aplicaciones orientadas a la protección de cultivos presentan desafíos adicionales, dado que las aves poseen un incentivo real (el de la obtención de sustento) que las impulsa a mantenerse en el área de interés, lo cual requiere otro enfoque por parte de las estrategias de disuasión.

2.2. Técnicas de pastoreo mediante drones

La literatura presenta múltiples aproximaciones al problema del pastoreo, las cuales pueden clasificarse según el tipo de modelo empleado. Los modelos basados en distancia priorizan la interacción con los individuos más cercanos al agente de pastoreo (Lee & Kim, 2017). Una debilidad de estos métodos es que tienden a generar dispersión si no se introducen mecanismos de reagrupamiento.

Por otra parte, los modelos heurísticos, como el propuesto por (Strömbom et al., 2014), introducen comportamientos alternantes de "conducción" y "recolección", que permiten mantener la cohesión del grupo mientras se lo guía hacia una zona objetivo. Este enfoque se destaca por su interpretabilidad y bajo costo de cómputo, ha sido ampliamente adoptado en simulaciones y experimentos con robots autónomos.

Los modelos basados en presión calculan zonas de influencia entre los drones y el enjambre, desplazando los drones hacia las áreas de mayor densidad de aves (Shi & Wang, 2022). Para esto, la flota asume una formación para efectuar el pastoreo y se calcula la influencia combinada de todos los drones para calcular los puntos de presión. Se ha comprobado mediante experimentos en simulaciones tridimensionales que la estrategia de mantener una formación en arco o "media

luna" permite optimizar la dirección del grupo y reducir el tiempo necesario para alcanzar la meta.

Más recientemente, se han desarrollado modelos basados en aprendizaje por refuerzo profundo, que entrenan políticas compartidas entre los drones utilizando observaciones locales (Hasan et al., 2022). Estos sistemas demuestran una notable adaptabilidad ante variaciones en la dinámica del entorno, aunque requieren un proceso de entrenamiento computacionalmente costoso.

2.2.1. Técnicas preventivas y conductivas

En la literatura, se pueden distinguir dos enfoques principales a la hora de proteger una cierta zona de la presencia de aves: preventivo y conductivo. En el primero se emplean técnicas que buscan evitar que la bandada ingrese en la zona, para esto se intenta desviar la dirección de vuelo original de forma que el grupo se mantenga por fuera del área. Un ejemplo de una solución desarrollada con este enfoque es el algoritmo de m-waypoints presentado en (Paranjape et al., 2018).

El segundo enfoque no considera una zona de protección directamente sino que, partiendo de una cierta posición inicial, busca llevar a la bandada hacia una posición objetivo. La solución presentada en (Strömbom et al., 2014) es un ejemplo de esto.

En el contexto de este proyecto, los enfoques preventivos presentan una limitación clara: se debe contar con la capacidad de detectar a las aves antes de que ingresen en la zona de los cultivos. Más aún, cuando se las detecta, se debe contar con un margen de tiempo lo suficientemente grande como para que los drones logren interceptar a la bandada antes de que llegue a la zona de protección. Si se quisiera adoptar una solución con este enfoque, se debería asegurar que el sistema de detección cuente con la infraestructura necesaria para cumplir con estas restricciones.

2.2.2. Aprendizaje por refuerzo profundo

Los modelos basados en aprendizaje por refuerzo profundo constituyen una alternativa moderna orientada a la toma de decisiones autónoma en escenarios complejos. Mediante algoritmos como PPO (Proximal Policy Optimization) o GAE (Generalized Advantage Estimation), los drones aprenden estrategias cooperativas que no dependen de reglas predefinidas (Hasan et al., 2022). Estos sistemas son capaces de guiar a grandes bandadas incluso cuando cada agente solo dispone de información limitada o incompleta del estado global del entorno (por ejemplo, percibir únicamente una porción de la bandada o un subconjunto de drones vecinos), lo que refleja de forma más realista las capacidades sensoriales presentes en aplicaciones robóticas. Este enfoque ha demostrado robustez y escalabilidad, aunque su aplicabilidad práctica en entornos agrícolas reales aún se encuentra en fase experimental.

2.3. Modelos de comportamiento de bandadas

El modelado computacional del comportamiento colectivo de las aves es esencial para simular y evaluar las estrategias de pastoreo. El modelo de Reynolds (Reynolds, 1987), conocido como modelo de "boids" (figura 3.8), describe el movimiento grupal de una bandada mediante tres reglas básicas: cohesión, separación y alineamiento. Este enfoque se ha convertido en la base de la mayoría de las simulaciones modernas de comportamiento de bandadas. Sin embargo, el modelo original no contempla aspectos relevantes para este proyecto como la atracción a un cierto objetivo o la huida frente a una amenaza.

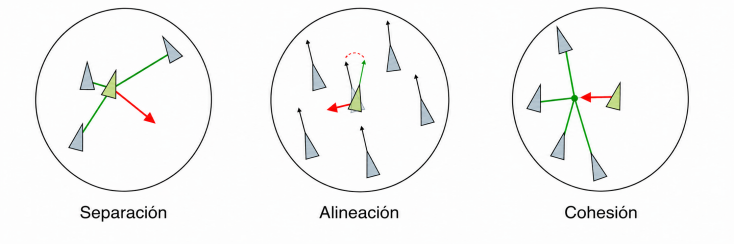

Figura 2.1: Modelo de boids en funcionamiento, cada triángulo representa un individuo de la bandada. Se representan las 3 reglas que regulan el comportamiento: separación, alineación y cohesión.

Otros modelos, como el de (Strömbom et al., 2014), incorporan estados discretos que imitan la reacción de los animales ante amenazas o estímulos externos. Este modelo tampoco contempla la atracción de los animales hacia cierto objetivo.

Dadas estas limitaciones, se desarrollaron extensiones sobre el modelo seleccionado para ajustarlo al contexto de esta investigación. La implementación de dichas extensiones será desarrollada en detalle en el siguiente capítulo.

Por otro lado, ambos modelos cuentan con parámetros que permiten ajustar el comportamiento colectivo de los animales y sus capacidades de movimiento dependiendo de la especie que se busque simular. Es importante remarcar que durante la investigación se detectó una alta escasez de información sobre el comportamiento de la especie de ave en la cual se enfoca este proyecto (*Myiopsitta monachus*).

2.4. Métricas para evaluar el pastoreo

Las métricas utilizadas para evaluar la efectividad de las estrategias de pastoreo varían según el tipo de experimento. Entre las más comunes se encuentran el tiempo total necesario para completar el objetivo, la distancia recorrida por los pastores, el grado de cohesión del grupo a pastorear y la desviación angular de su movimiento respecto al objetivo deseado (Shi & Wang, 2022; Strömbom et al., 2014). Asimismo, se emplean medidas estadísticas, como la desviación

estándar de las posiciones o el área ocupada por la bandada, para determinar la eficacia del control.

2.5. Herramientas y entornos tecnológicos

La implementación de un sistema de control cooperativo de drones requiere de un conjunto de herramientas de software robustas. En este proyecto se emplea **ROS 2**, un marco de trabajo que permite la comunicación entre módulos de software distribuidos mediante el intercambio de mensajes. Estos módulos se implementan como procesos llamados nodos, los cuales ejecutan su lógica interna múltiples veces por segundo para lograr comportamientos altamente responsivos. ROS 2 se destaca por su arquitectura basada en el protocolo DDS (Data Distribution Service), que facilita la integración en sistemas robóticos multi-agente.

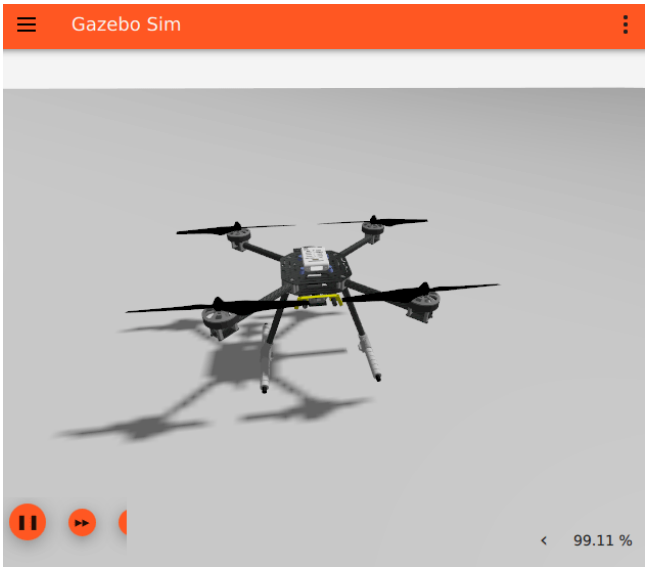

Figura 2.2: Modelo 3D de dron x500 provisto por PX4 dentro de un entorno de simulación de Gazebo.

El sistema de control de vuelo **PX4** actúa como el firmware encargado de recibir las instrucciones de navegación generadas por los controladores de alto nivel, ejecutando las maniobras de vuelo correspondientes. Además, PX4 provee herramientas para entornos de simulación como modelos realistas de drones y mundos personalizados de prueba, la figura 2.2 muestra el modelo de dron utilizado para este trabajo. Una alternativa es el controlador ArduPilot el cual provee funcionalidades similares, sin embargo, para el desarrollo en sistemas con integración avanzada de ROS 2 se recomienda PX4 (ThinkRobotics, 2025).

Finalmente, el simulador **Gazebo Garden** permite recrear entornos tridimensionales con físicas realistas, ofreciendo una plataforma ideal para la validación previa a pruebas de campo.

El stack tecnológico descrito es ampliamente utilizado en el área de la robótica y existe gran cantidad de documentación relevante tanto para el entrenamiento como para el desarrollo sobre estas herramientas. Por estas razones, se decidió utilizar este estándar.

2.6. Síntesis de los antecedentes

A partir de la revisión realizada, se concluye que existen fundamentos teóricos y empíricos que respaldan la viabilidad del pastoreo de aves mediante drones autónomos. En el plano teórico, la literatura destaca que reglas simples de interacción local, como las del modelo de Reynolds, permiten predecir y manipular comportamientos colectivos complejos. Desde una perspectiva empírica, investigaciones en entornos críticos como aeropuertos han demostrado que los drones logran alterar y desviar el vuelo de las bandadas de manera controlada (Paranjape et al., 2018). Los modelos clásicos ofrecen soluciones controlables, mientras que los enfoques basados en aprendizaje por refuerzo abren la puerta a estrategias más adaptativas. No obstante, la literatura actual se enfoca mayormente en escenarios de protección de aeropuertos o en el pastoreo de mamíferos, por lo que la aplicación a cultivos agrícolas continúa siendo un campo abierto de investigación. Este proyecto se propone contribuir a dicho vacío, adaptando las técnicas y modelos revisados a un contexto agrícola y evaluando su desempeño en un entorno de simulación realista.

Para esto, se decidió utilizar el algoritmo propuesto por Strömbom como base para la lógica de los drones. Este enfoque heurístico es particularmente viable debido a su bajo costo computacional y su capacidad para mantener la cohesión del grupo, alternando entre los estados de "conducción" y "recolección", mientras se le guía hacia un objetivo, imitando fielmente la lógica de un pastor real. Sin embargo, este algoritmo fue diseñado originalmente para ser utilizado por un solo agente pastor, por lo que se buscaron distintas estrategias y enfoques para extender su funcionalidad a un conjunto de agentes cooperativos.

En cuanto al modelo de comportamiento de bandada, se decidió utilizar boids. Esto se debe a que el modelo de Strömbom para simulación de comportamiento animal solamente contempla el movimiento de los animales en estado de huida mientras que boids presenta características más adecuadas para la representación del movimiento de aves y, por lo tanto, puede adaptarse fácilmente al escenario que motiva este proyecto.

De esta manera, la presente investigación combina diversas partes del trabajo desarrollado en el campo hasta la fecha para introducir nuevas soluciones y evaluar su aplicabilidad.

Capítulo 3

Desarrollo

En este capítulo se describe la solución desarrollada para el modelado de la problemática descrita. Primeramente se especifica el problema a resolver, sus parámetros, variables y entidades relevantes. Luego, se describe la arquitectura del sistema desarrollado, contemplando los distintos módulos que lo integran, sus propósitos e interacciones. Finalmente, se detallan los módulos de comportamiento de bandada y de control de drones.

3.1. Especificación del problema

Se busca modelar la problemática descrita en el capítulo 1, para esto se deben tener en cuenta los elementos relevantes que componen un escenario de cultivos: los árboles y el área que abarcan. También se deben contemplar los principales actores: los drones y las aves. Para cada uno de estos actores se debe considerar la naturaleza de su comportamiento y las interacciones que se dan entre ellos. Finalmente, se debe establecer un objetivo claro destinado a asegurar la protección de los cultivos frente a la presencia de las aves. La figura 3.1 muestra un esquema del modelado descrito a continuación.

3.1.1. Entidades

El problema se modela mediante un conjunto de aves $\mathcal{L}$, un conjunto de árboles $\mathcal{T}$ y un conjunto de drones $\mathcal{D}$. Cada elemento $d_i \in \mathcal{D}$, $l_i \in \mathcal{L}$ y $k_i \in \mathcal{T}$ tiene una cierta posición $p_{li}(t)$, $p_{di}(t)$, $p_{ki}(t)$ en cada instante de tiempo t. Además, cada dron y cada ave tiene una velocidad $v_{di}(t)$ y $v_{li}(t)$ en cada t.

Por otro lado, se define una zona o área a proteger Z mediante los siguientes parámetros:

$$x_{min},\ x_{max},\ y_{min},\ y_{max} \text{ y } z_{max} \ (z_{min} \text{ se asume } 0).$$

Luego,

$$Z = \{(x, y, z) \in \mathbb{R}^3 : x_{min} \leq x \leq x_{max}, y_{min} \leq y \leq y_{max}, 0 \leq z \leq z_{max}\}$$

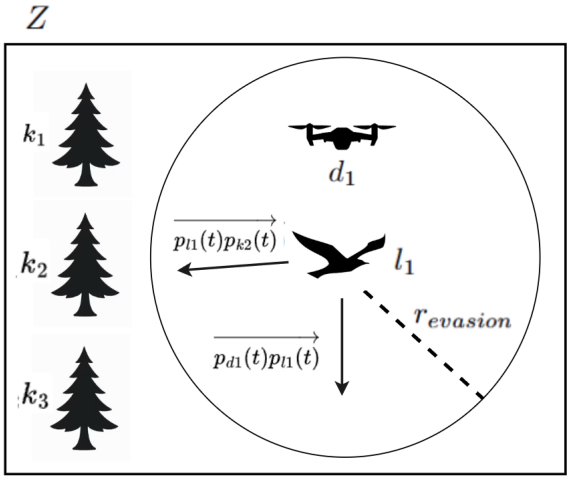

Figura 3.1: Esquema de los elementos principales del modelo y sus interacciones. Se puede ver $l_1 \in \mathcal{L}$, las fuerzas de atracción y repulsión que actúan sobre l_1 y el parámetro $r_{evasion}$. Notar que k_2 es el árbol más cercano.

3.1.2. Asunciones y requerimientos

En esta instancia se asume que se puede detectar a todas las aves que se encuentren dentro de Z.

El movimiento de las aves sigue las reglas del modelo que se describe en mayor detalle en la sección 3.3, Por ahora es relevante resaltar que la velocidad de las aves se ve influenciada por la presencia de los drones, huyendo en dirección opuesta a aquellos que se encuentren a una distancia menor a un parámetro $r_{evasion}$. Además, cada ave se ve atraída hacia el árbol más cercano a ella. A estos comportamientos se los denomina repulsión y atracción. Cabe destacar que la expresión presentada a continuación contempla únicamente estos efectos, mientras que las reglas completas del modelo se introducen posteriormente en la Sección 3.3.

Esto es, dado un tiempo t y un ave l_i

$$v_{li}(t) = \overrightarrow{p_{li}(t)p_{kc}(t)} + \sum_{dj \in \mathcal{D}_{li}(t)} \overrightarrow{p_{dj}(t)p_{li}(t)}$$

Donde:

$$k_c = \arg\min_{k \in \mathcal{T}} \|\mathbf{p}_{li}(t) - \mathbf{p}_k(t)\|$$

$$\mathcal{D}_{li}(t) = \{d_j \in \mathcal{D} : \|p_{li}(t) - p_{dj}(t)\| < r_{evasion}\}.$$

Por otro lado, la velocidad de los drones deberá ser programada de forma de determinar en cada momento el movimiento deseado. Para esto, cada dron conoce los conjuntos

$$\mathcal{L}'(t) = \{l_i : l_i \in L, p_{li}(t) \in Z\}$$

$$\mathcal{D}'(t) = \{d_i : d_i \in D, p_{di}(t) \in Z\}$$

$\mathcal{L}'(t)$ son las posiciones reportadas por el módulo de detección de aves y $\mathcal{D}'(t)$ las posiciones reportadas por el sistema de localización de cada dron.

3.1.3. Objetivos

Se busca controlar el comportamiento de los drones de forma de dotarlos de una lógica de movimiento que logre expulsar a las aves de la zona protegida en una cierta cantidad de tiempo menor a un parámetro t_{max}. Esto es:

$$\mathcal{L}'(t_f) = \emptyset \text{ con } t_f < t_{max}$$

Para eso se deberá balancear los comportamientos de atracción a los árboles y repulsión a los drones para guiar a la bandada. Además, se deberán establecer mecanismos para prevenir colisiones entre los drones durante el vuelo.

3.2. Módulos del sistema

Con el objetivo de modelar y resolver la problemática planteada se diseñó un sistema integrado por tres módulos: interfaz de simulación, controlador de dron y comportamiento de bandada. Se implementan como nodos de ROS 2, lo cual permite mantener un flujo constante de información entre los tres procesos a través de tópicos (canales de comunicación definidos por ROS 2 para el intercambio de información entre nodos), en la figura 3.2 se puede apreciar el flujo de esta comunicación.

El módulo de controlador de dron se encarga de realizar las operaciones de inicialización, despegue y vuelo mediante el envío de instrucciones a PX4, el cual gestiona el control del movimiento de los drones dentro del entorno simulado en Gazebo. Asimismo, recibe las posiciones de las aves desde el módulo de comportamiento de bandada para ejecutar el algoritmo de pastoreo.

Adicionalmente, cada instancia del controlador de dron se comunica con los demás nodos homólogos para mantener actualizada la posición de la flota en todo momento. Esto permite evitar colisiones entre los agentes y coordinar la distribución de las tareas de pastoreo.

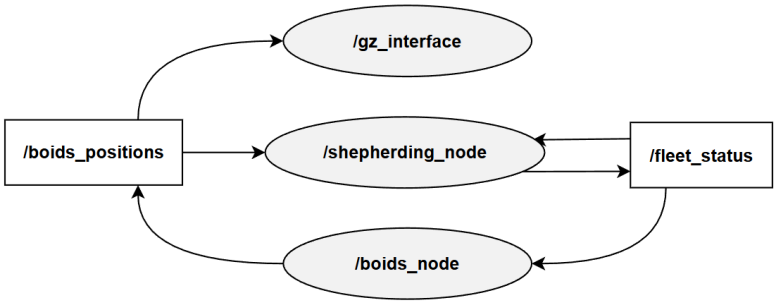

Figura 3.2: Diagrama de nodos ROS 2 del sistema en funcionamiento extraído con la herramienta rqt_graph. El nodo *boids_node* corresponde con el módulo de comportamiento de bandada, el nodo *shepherding_node* con el controlador de dron y el nodo *gz_interface* con la interfaz de simulación. Los rectángulos representan los tópicos que comunican estos nodos, *boids_positions* transmite las posiciones de las aves y *fleet_status* el estado y posición de cada dron. No se incluyen los tópicos relacionados a la comunicación con PX4.

El módulo de comportamiento de bandada implementa el algoritmo *boids* de Reynolds para modelar la dinámica colectiva del grupo. Recibe la posición de las distintas instancias del controlador de dron, información que se emplea para simular comportamientos de huida y evasión frente a la presencia de los agentes. Asimismo, publica la posición exacta de cada miembro de la bandada, permitiendo que los demás nodos del sistema dispongan de información actualizada para la ejecución de sus respectivas tareas.

Finalmente, el módulo de interfaz de simulación es el encargado de inicializar los escenarios en Gazebo, actualizar las posiciones de las aves dentro del entorno de simulación y mantener los datos necesarios para el cálculo de las métricas en cada experimento. Para ello, recibe las posiciones de los drones y de la bandada. En la figura 3.3 se presentan las interacciones descritas entre los distintos componentes del sistema.

Cabe señalar que tanto el módulo de comportamiento de bandada como el módulo de interfaz de simulación no serían requeridos para la implementación del sistema en un entorno real. En su lugar, sería necesario contar con un sistema de detección de aves que permita al controlador de dron conocer las posiciones de los miembros de la bandada de manera actualizada y periódica.

Por otro lado, el módulo de controlador de dron sí sería indispensable en un despliegue real. Por esta razón, se decidió desarrollar su lógica en el lenguaje C++ en lugar de Python (ambos lenguajes soportados por ROS 2), debido a su mayor eficiencia en tiempo de cómputo (Zehra et al., 2020). Esto hace que C++ sea una alternativa más adecuada para sistemas de control como el que se busca desarrollar, los cuales requieren baja latencia de respuesta.

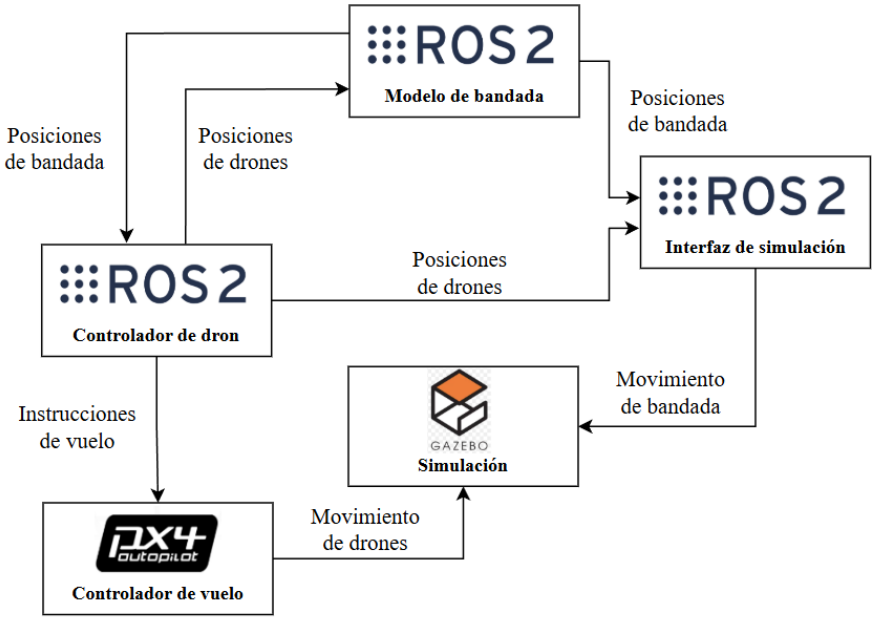

Figura 3.3: Diagrama de componentes del sistema. Para cada componente se indica la tecnología utilizada para su implementación y la información que comparte con el resto.

3.3. Comportamiento de bandada

La simulación de la bandada se basa en el modelo *boids* de Reynolds, el cual define reglas que determinan el movimiento de cada individuo dentro de un grupo de aves. Los detalles específicos de estas reglas se describen en la Sección 3.3.3. El modelo desarrollado (figura 3.4) extiende la lógica definida en estas reglas, especificando cada ave como un agente autónomo que exhibe comportamientos reactivos ante otros individuos (miembros de la bandada en su proximidad), fuentes de alimento (árboles) y amenazas (drones).

El módulo se implementa como un nodo ROS 2, lo cual permite un intercambio constante de información con los demás nodos que componen el sistema. En cada paso de la simulación, el nodo de comportamiento de bandada recibe las posiciones de los drones de cada nodo del controlador de dron.

Luego se aplican las reglas del modelo de *boids* para simular el comportamiento de las aves. Finalmente, envía las posiciones de cada miembro de la bandada a las instancias del controlador de dron y a la interfaz de simulación para que se visualicen dentro del entorno simulado.

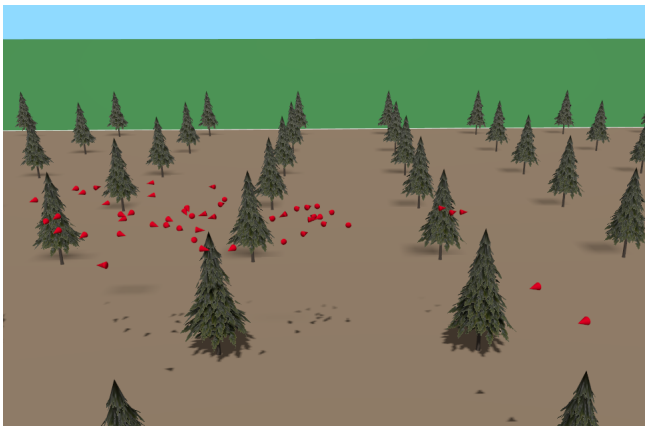

Figura 3.4: El modelo de bandada en funcionamiento dentro del ambiente de Gazebo. Cada cono rojo representa un individuo de la bandada.

3.3.1. Diseño del modelo

El diseño final del modelo de simulación de aves es el resultado de un proceso iterativo que combinó análisis teórico y experimentación computacional. A continuación se describen las etapas principales del desarrollo:

Requerimientos del modelo

Para garantizar que la simulación sea una herramienta válida de prueba para el algoritmo de pastoreo propuesto, el modelo biológico debe replicar las dinámicas fundamentales que ocurren en un escenario real de invasión de cultivos. A partir del análisis del problema operativo, se definieron los siguientes comportamientos esenciales que los agentes (aves) deben exhibir:

- Formación de bandadas: Dado que la problemática de las plagas aviares reside en el daño masivo causado por grupos y no por individuos aislados, el modelo debe simular la cohesión social. Además, la cotorra es un ave gregaria por lo que se debe simular su comportamiento bajo este contexto. La implementación de estos aspectos sociales es requisito indispensable para validar si el algoritmo de pastoreo logra controlar al grupo entero o si, por el contrario, provoca su dispersión (fragmentación de la bandada), lo cual sería un resultado no deseado.

- Búsqueda de recursos: Para evaluar la eficacia del sistema de repulsión, los agentes deben tener una motivación intrínseca para permanecer en la zona de cultivo. El modelo debe incluir una fuerza de atracción hacia los árboles frutales que compita con la fuerza de repulsión del dron, simulando así la resistencia real de la plaga a abandonar su fuente de alimento.

- Evasión: El núcleo del proyecto es la interacción entre el agente robótico y la fauna. Por tanto, los agentes biológicos deben poseer un mecanismo de detección y respuesta ante amenazas.
- Restricciones físicas: Para que las conclusiones sobre la viabilidad de la solución sean transferibles a la realidad, el movimiento de las aves no puede ser arbitrario. El modelo debe respetar límites físicos de velocidad máxima, aceleración y radio de giro (inercia). Esto evita resultados donde el dron logre pastorear agentes que se mueven de manera antinatural o que carecen de momento lineal.

Selección del modelo base

Se realizó una revisión de los modelos de simulación de comportamiento colectivo en animales. Se evaluaron los siguientes enfoques:

- Modelo de Strömbom: el modelo no contempla los movimientos de los animales por fuera del estado de huida ante un pastor (Strömbom et al., 2014) por lo que se decidió descartar esta alternativa.
- Modelo de boids: seleccionado por su simplicidad, realismo y amplia validación en simulaciones (Paranjape et al., 2018; Shi & Wang, 2022).

El modelo de boids introduce tres reglas básicas para modelar el comportamiento de una bandada: separación, cohesión y alineamiento (Bajec et al., 2007). Estas reglas pueden verse como fuerzas de interacción entre los miembros de la bandada que se encuentran a una cierta proximidad el uno del otro.

- Separación: El individuo tiende a moverse de forma tal de evitar el amontonamiento local con otros miembros del grupo.
- Alineamiento: La dirección de movimiento del individuo se aproxima a la dirección de movimiento promedio de los miembros cercanos del grupo.
- Cohesión: El individuo tiende a moverse hacia el centro de masa formado por las posiciones de los miembros cercanos del grupo.

Se muestra una representación de estas fuerzas en la figura 3.5. A su vez, el modelo cuenta con algunos parámetros que permiten configurar el funcionamiento de sus reglas. Estos parámetros son:

- w_{ali}: Peso de la fuerza de alineamiento.
- w_{coh}: Peso de la fuerza de cohesión.
- w_{sep}: Peso de la fuerza de separación.
- $r_{percepcion}$: Radio en el cual se considera que un individuo influye sobre otro.

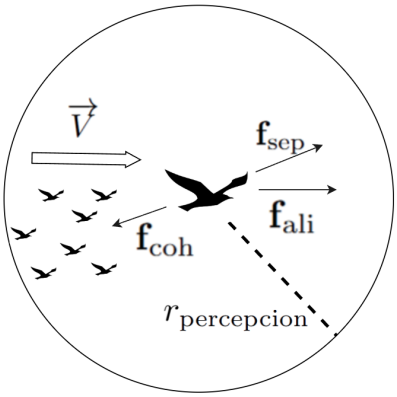

Figura 3.5: Diagrama de fuerzas que definen el comportamiento del modelo de Reynolds original. El vector $\overrightarrow{V}$ representa la velocidad promedio de los individuos dentro del radio $r_{evasion}$.

Extensiones requeridas

En principio, las reglas de boids proveen un modelo realista para simular el movimiento grupal de la bandada. Sin embargo, en base a los requerimientos detectados, se decidió extender el modelo para obtener un comportamiento más ajustado al contexto de este proyecto. Para esto, se incorporó la interacción con recursos discretos estáticos (árboles), amenazas dinámicas (drones) y estados internos (posada, en huida, en vuelo).

3.3.2. Extensiones sobre el modelo

A continuación se detallan las extensiones realizadas sobre el modelo de boids a fin de adaptarlo a los requerimientos mencionados.

Integración física básica

La mecánica del movimiento de cada ave se modela en base a las siguientes componentes:

- Movimiento con aceleraciones: Interpolación de la velocidad entre un ciclo y el siguiente para prevenir cambios bruscos e irreales de la dinámica de movimiento.
- Límites de velocidad: Cota máxima de velocidad de movimiento para prevenir valores exagerados.
- Orientación gradual mediante la limitación de las velocidades angulares de las aves: Inicialmente no existía orientación gradual, lo que provocaba

giros instantáneos e irreales. Ante la presencia de un dron, las aves cambiaban la dirección de forma inmediata, algo físicamente imposible. En la biomecánica aviar, cualquier viraje requiere de un alabeo para generar una fuerza centrípeta mediante la sustentación de sus alas. Este proceso está limitado por la inercia, la fuerza muscular y las leyes de la aerodinámica (Warrick & Dial, 1998). Al incorporar un límite de velocidad angular, se obtiene un comportamiento que respeta estas restricciones de vuelo (Reynolds et al., 1999).

Incluyendo estos mecanismos se logra modelar una dinámica de vuelo realista y disminuye la brecha entre el modelo y el comportamiento real de una bandada.

Selección de árboles

En principio se modeló la búsqueda de alimento como una simple fuerza de atracción que se ejerce sobre cada ave hacia el árbol más cercano. Sin embargo, esto provocaba que toda la bandada convergiera sobre el mismo árbol, causando saturación.

Este comportamiento no tiene en cuenta restricciones de espacio y cantidad de alimento que existen en escenarios reales. Por esta razón, se decidió incorporar la siguiente lógica de reservación distribuida:

- Cada árbol tiene un límite de "alimento" o aves que puede albergar
- Cada árbol mantiene un conteo de aves que lo han elegido
- Un ave puede cambiar de objetivo si detecta que un árbol está saturado

De esta forma se logra que las aves se distribuyan entre los árboles más cercanos en lugar de agruparse en uno solo.

Evasión de drones

Para modelar el comportamiento de evasión, se consideran todos los drones que se encuentren dentro de un radio $r_{evasion}$ de cada individuo. Luego, se aplica una fuerza repulsiva por cada uno de ellos. Este comportamiento de huida se modela como una mecánica del individuo y no de la bandada. Bajo este supuesto, pueden existir individuos que se encuentren en distintos estados dentro de una misma bandada y en el mismo instante de tiempo.

Si bien esto generaba un comportamiento de huida razonable, se observó que inmediatamente después de que las aves escapaban (salieran del radio de evasión) intentaban volver a su árbol objetivo por una ruta que era interceptada automáticamente por el dron. Esto provocaba que oscilaran entre el estado de "huida" y "en vuelo" sin lograr determinar un comportamiento coherente.

Para evitar esto, se añadieron controles de "accesibilidad de trayectorias", descartando las rutas que atraviesen áreas controladas por drones.

3.3.3. Especificación del modelo

A continuación se detalla la implementación de los aspectos del modelo previamente descritos.

Representación del estado

Cada ave l_i en el instante t se caracteriza por las siguientes variables:

- Posición ($\mathbf{p}_{li}(t) \in \mathbb{R}^3$): Ubicación en coordenadas cartesianas dentro del mundo de Gazebo. Es la base para los cálculos de vecindad y detección de amenazas.
- Velocidad ($\mathbf{v}_{li}(t) \in \mathbb{R}^3$): Vector de desplazamiento lineal. Su magnitud está limitada por una constante v_{max} m/s, asegurando que las aves mantengan un desplazamiento realista.
- Aceleración ($\mathbf{a}_{li}(t) \in \mathbb{R}^3$): Resultante de la suma de fuerzas de comportamiento (boids, atracción a recursos y evasión). Se utiliza para actualizar la velocidad mediante la integración de Euler: $\mathbf{v}(t + dt) = \mathbf{v}(t) + \mathbf{a}(t)dt$.
- Orientación (θ_{li}, ϕ_{li}): Representan los ángulos de inclinación y guiñada (rotación de un agente alrededor de su eje vertical). Los cambios en estos ángulos se derivan de las velocidades angulares, cuya magnitud está limitada por un parámetro $\omega_{\max}$. Esto permite evitar cambios de dirección instantáneos que romperían la verosimilitud de la simulación.
- Estado ($s_{li}(t)$): Variable discreta que determina el modo de operación del agente. Los estados (*volando, posada, huyendo*) actúan como filtros que ponderan de manera distinta las fuerzas de aceleración según el contexto (por ejemplo, ignorar la atracción a los árboles cuando se está en estado "huyendo").

Algoritmo principal

El algoritmo implementa una jerarquía de prioridades que refleja el comportamiento natural de las aves mediante un sistema multi-agente que integra múltiples capas de decisión y comportamiento.

Primero se evalúa la supervivencia mediante la detección de amenazas. Si se detecta una amenaza dentro del radio de evasión $r_{\text{evasión}}$ (línea 5), el comportamiento se enfoca en el escape e ignora la obtención de alimento (líneas 6-7).

Por otro lado, si no se detectan drones en la cercanía, el comportamiento se enfoca en acercarse al objetivo (línea 18). Este objetivo se obtiene a través de la búsqueda de árboles (línea 15) disponibles que se detalla más adelante. El parámetro w_{obj} regula la fuerza de atracción de cada ave hacia su árbol objetivo.

Algoritmo 1 Simulación de Aves

Entrada: Conjunto de aves $\mathcal{L}$, árboles $\mathcal{T}$, drones $\mathcal{D}$
Salida: Posiciones actualizadas de todas las aves

1: **for all** $l_i \in \mathcal{L}$ **do**
2: $\quad \mathbf{a}_{li}(t) \leftarrow \mathbf{0}$
3: $\quad$ **for all** $d_j \in \mathcal{D}$ **do**
4: $\quad\quad d \leftarrow \|p_{li}(t) - p_{dj}(t)\|$
5: $\quad\quad$ **if** $d < r_{\text{evasión}}$ **then**
6: $\quad\quad\quad \mathbf{a}_{li}(t) \leftarrow \mathbf{a}_{li}(t) + \text{Evadir}(l_i, d_j)$
7: $\quad\quad\quad s_{li}(t) \leftarrow \textit{huyendo}$
8: $\quad\quad$ **end if**
9: $\quad$ **end for all**
10: $\quad$ **if** $s_{li}(t) = \textit{huyendo}$ **then**
11: $\quad\quad \mathbf{a}_{li}(t) \leftarrow \mathbf{a}_{li}(t) + \text{Boids}(l_i, \mathcal{L})$
12: $\quad$ **else**
13: $\quad\quad$ **if** $s_{li}(t) \neq \textit{posada}$ **then**
14: $\quad\quad\quad$ **if** objetivo $= \emptyset$ **then**
15: $\quad\quad\quad\quad$ objetivo $\leftarrow \text{BuscarArbol}(l_i, \mathcal{T}, \mathcal{L}, \mathcal{D})$
16: $\quad\quad\quad$ **end if**
17: $\quad\quad$ **end if**
18: $\quad\quad \mathbf{a}_{li}(t) \leftarrow w_{obj} \cdot \text{AtraccionObjetivo}(l_i) + \text{Boids}(l_i, \mathcal{L})$
19: $\quad$ **end if**
20: $\quad \text{ActualizarMovimiento}(l_i, \mathbf{a}_{li}(t))$
21: **end for all**

En ambos casos, se añaden las reglas de boids al cálculo del movimiento para mantener el comportamiento colectivo independientemente del estado individual de cada ave (líneas 11 y 18).

Reglas de boids

El comportamiento de la bandada se ve influido constantemente por las tres reglas clásicas del modelo de Reynolds. El algoritmo 2 muestra la implementación de estas reglas.

El algoritmo recorre el conjunto $\mathcal{L}$ y, para cada individuo que se encuentre a una distancia menor de $r_{evasion}$, calcula las fuerzas que representan las reglas definidas anteriormente. Al aplicar estas interacciones se genera el comportamiento grupal de la bandada donde, sin importar el estado u objetivo de cada individuo, el movimiento siempre se ve influenciado por el resto del grupo. De forma similar, las reglas de boids causan que los movimientos individuales causados por el estado de huida afecten indirectamente en la bandada a través de las fuerzas de alineamiento y cohesión.

Algoritmo 2 Comportamiento de boids

Entrada: Ave l_i, Conjunto de aves $\mathcal{L}$

function BOIDS
  $\mathbf{f}_{\text{sep}} \leftarrow \mathbf{0}$, $\mathbf{f}_{\text{ali}} \leftarrow \mathbf{0}$, $\mathbf{f}_{\text{coh}} \leftarrow \mathbf{0}$
  $N \leftarrow 0$
  for all $l_j \in \mathcal{L} : l_j \neq l_i$ y $s_{lj}(t) \neq \textit{posada}$ **do**
    $d \leftarrow \|\mathbf{p}_{li}(t) - \mathbf{p}_{lj}(t)\|$
    if $0 < d < r_{\text{percepcion}}$ **then**
      $\mathbf{f}_{\text{ali}} \leftarrow \mathbf{f}_{\text{ali}} + \mathbf{v}_{lj}$
      $\mathbf{f}_{\text{coh}} \leftarrow \mathbf{f}_{\text{coh}} + \mathbf{p}_{lj}(t)$
      $\mathbf{f}_{\text{sep}} \leftarrow \mathbf{f}_{\text{sep}} + \frac{\mathbf{p}_{li}(t) - \mathbf{p}_{lj}(t)}{d}$
      $N \leftarrow N + 1$
    end if
  end for all
  if $N > 0$ **then**
    $\mathbf{f}_{\text{ali}} \leftarrow w_{\text{ali}} \cdot \text{Normalizar}\left(\frac{\mathbf{f}_{\text{ali}}}{N}\right)$
    $\mathbf{f}_{\text{coh}} \leftarrow w_{\text{coh}} \cdot \text{Normalizar}\left(\frac{\mathbf{f}_{\text{coh}}}{N} - \mathbf{p}_{li}(t)\right)$
    $\mathbf{f}_{\text{sep}} \leftarrow w_{\text{sep}} \cdot \text{Normalizar}\left(\frac{\mathbf{f}_{\text{sep}}}{N}\right)$
  end if
  return $\mathbf{f}_{\text{ali}} + \mathbf{f}_{\text{coh}} + \mathbf{f}_{\text{sep}}$
end function

Búsqueda de árboles

La selección de árboles considera tanto la disponibilidad como la accesibilidad de los recursos alimenticios.

El sistema de reservas de árboles previene la sobresaturación mediante un mecanismo distribuido. Cada árbol $k \in \mathcal{T}$ mantiene un contador de reservas que se incrementa cuando un ave lo selecciona como objetivo y se decrementa cuando el ave cambia de objetivo. Este mecanismo evita que un ave elija como objetivo un árbol sin reservas disponibles.

Por otro lado, se realizan algunos controles para evitar que se seleccionen árboles que se encuentran protegidos por los drones. La función nominada NoIntersectaDron$(l_i, k, \mathcal{D})$ implementa un algoritmo de detección de intersección de trayectorias que evita que las aves vuelen directamente hacia zonas de influencia de los drones. De forma similar, la función FueraDeAlcanceDron$(k, \mathcal{D})$ verifica si el árbol k se encuentra a una distancia menor a $r_{evasion}$ de algún dron. La figura 3.6 presenta ejemplos de ambas lógicas de selección de objetivo.

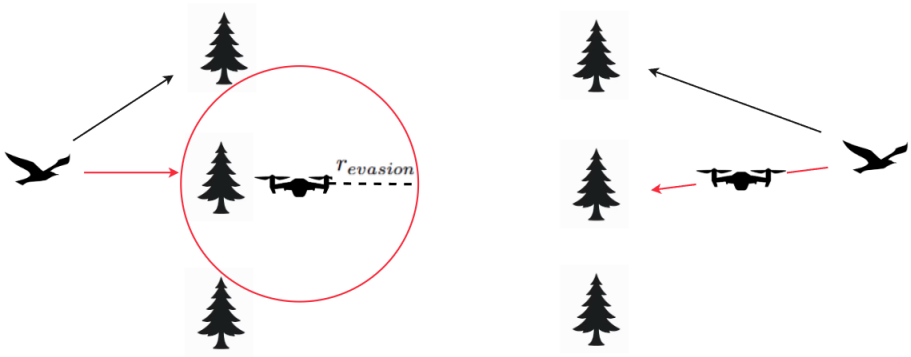

Figura 3.6: Diagrama de lógicas de selección de árbol objetivo, a la izquierda se presenta un ejemplo de FueraDeAlcanceDron y a la derecha un ejemplo de NoIntersectaDron. En ambos diagramas se muestra en rojo la ruta que tomaría el ave si no estuviera presente el dron y en negro la ruta que elige como alternativa.

Algoritmo 3 Búsqueda de Árbol Disponible

Entrada: Ave l_i, Conjunto de árboles $\mathcal{T}$, Conjunto de aves $\mathcal{L}$, Conjunto de drones $\mathcal{D}$

function BUSCARARBOL
  $\mathcal{T}_{\text{candidatos}} \leftarrow \emptyset$
  for all $k \in \mathcal{T}$ **do**
    $n_{\text{ocupadas}} \leftarrow |\{l_j \in \mathcal{L} : \text{arbol}(l_j) = k\}|$
    if $n_{\text{ocupadas}} < arbol_{max}$ **then**
      if FueraDeAlcanceDron$(k, \mathcal{D})$ **y** NoIntersectaDron$(l_i, k, \mathcal{D})$ **then**
        $\mathcal{T}_{\text{candidatos}} \leftarrow \mathcal{T}_{\text{candidatos}} \cup \{k\}$
      end if
    end if
  end for all
  if $\mathcal{T}_{\text{candidatos}} \neq \emptyset$ **then**
    return $\arg \min_{k \in \mathcal{T}_{\text{candidatos}}} \|\mathbf{p}_{li}(t) - \mathbf{p}_k(t)\|$
  else
    return $\emptyset$
  end if
end function

Evasión de drones

Se aplica la siguiente fuerza por cada dron d_j que se encuentre a una distancia menor a $r_{evasion}$ del ave l_i:

$$\text{Evadir}(l_i, d_j) = \frac{r_{\text{evasión}} - \|\mathbf{p}_{li}(t) - \mathbf{p}_{dj}(t)\|}{\|\mathbf{p}_{li}(t) - \mathbf{p}_{dj}(t)\|^2} \cdot \frac{\mathbf{p}_{li}(t) - \mathbf{p}_{dj}(t)}{\|\mathbf{p}_{li}(t) - \mathbf{p}_{dj}(t)\|} \tag{3.1}$$

Esta ecuación define la fuerza de evasión como un campo repulsivo (representado en la figura 3.6 por el círculo rojo) que actúa exclusivamente dentro del radio crítico $r_{evasion}$. Es básicamente la fuerza de repulsión que un dron ejerce sobre un ave, pero a niveles de representación sería la respuesta de las aves ante la amenaza del dron.

A continuación se explican los componentes de la ecuación. Los términos se pueden leer como: Fuerza = Magnitud × Dirección.

El vector de dirección (normalizado) corresponde con el siguiente término:

$$\frac{\mathbf{p}_{li}(t) - \mathbf{p}_{dj}(t)}{\|\mathbf{p}_{li}(t) - \mathbf{p}_{dj}(t)\|} \tag{3.2}$$

Este término es un vector unitario. Se obtiene restando la posición del dron a la del ave, esto asegura que el ave siempre intente alejarse en dirección contraria a la amenaza.

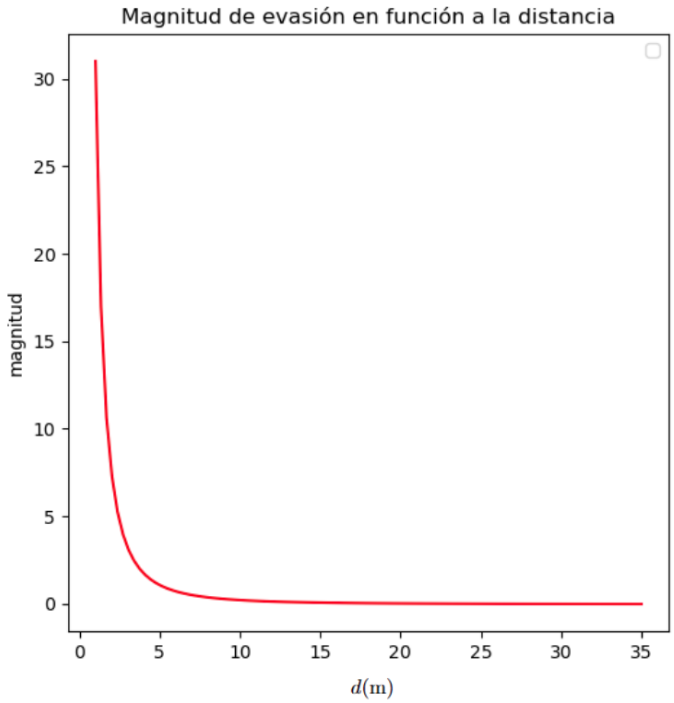

Figura 3.7: Gráfica de magnitud de la fuerza de evasión en función a la distancia entre el ave y el dron para $r_{\text{evasión}} = 32{,}0m$ (valor utilizado en los experimentos finales).

La magnitud (intensidad de la fuerza) corresponde con el siguiente término:

$$\frac{r_{\text{evasión}} - \|\mathbf{p}_{li}(t) - \mathbf{p}_{dj}(t)\|}{\|\mathbf{p}_{li}(t) - \mathbf{p}_{dj}(t)\|^2} \tag{3.3}$$

Representa la lógica de "huida" del ave. El numerador $r_{\text{evasión}} - d$ (con $d = \|\mathbf{p}_{li}(t) - \mathbf{p}_{dj}(t)\|$) garantiza que la fuerza sea cero cuando el ave está exactamente en el límite del radio de evasión. A medida que el dron se interna más en el radio, la distancia d disminuye y el numerador crece.

El denominador d^2 implementa una ley exponencial inversa, cuyo crecimiento en función a d se muestra en la figura 3.7. Como se puede apreciar, para valores mayores de d la influencia de la fuerza es mínima y para valores menores la fuerza crece de forma cuadrática forzando maniobras evasivas con mayor agresividad.

Actualización del movimiento

Una vez calculada la aceleración resultante $a_{li}(t)$ mediante la suma ponderada de las fuerzas de boids y las fuerzas de evasión y atracción, el sistema actualiza el estado de movimiento de cada ave. Este proceso no se realiza de forma directa sobre la posición, sino que atraviesa una etapa de filtrado para garantizar que el movimiento respete limitaciones físicas.

La actualización se rige por las siguientes ecuaciones de integración y control:

1. Integración de la velocidad: Se utiliza un esquema de Euler simple para obtener la velocidad tentativa:

$$\mathbf{v}_i(t + \Delta t) = \mathbf{v}_i(t) + \mathbf{a}_i(t) \cdot \Delta t \tag{3.4}$$

2. Límite de aceleración: Cota máxima $a_{\max}$ de la magnitud de la aceleración incluida en la fórmula de Euler.

3. Límite de velocidad lineal: Para evitar que las fuerzas acumuladas generen velocidades demasiado grandes, se aplica un límite superior $v_{\max}$:

$$\|\mathbf{v}_i(t + \Delta t)\| = \text{mín}\left(\|\mathbf{v}_i(t + \Delta t)\|, v_{\max}\right) \tag{3.5}$$

4. Restricción de velocidad angular (Giro gradual): Esta es la extensión clave para evitar cambios de dirección instantáneos. Si se define $\Delta\theta$ y $\Delta\phi$ como la diferencia entre la orientación actual y la dirección del nuevo vector de velocidad, la actualización de los ángulos de inclinación y guiñada se limita por una velocidad angular máxima $\omega_{\max}$:

$$\theta_i(t + \Delta t) = \theta_i(t) + \text{sign}(\Delta\theta) \cdot \text{mín}\left(|\Delta\theta|, \omega_{\max} \cdot \Delta t\right) \tag{3.6}$$

$$\phi_i(t + \Delta t) = \phi_i(t) + \text{sign}(\Delta\phi) \cdot \text{mín}\left(|\Delta\phi|, \omega_{\max} \cdot \Delta t\right) \tag{3.7}$$

Donde la función $\text{sign}(\cdot)$ asegura que el giro se realice en la dirección correcta hacia el objetivo. $\Delta\theta$ es la diferencia angular entre la orientación actual y la deseada, y $\omega_{\max}$ la velocidad angular máxima.

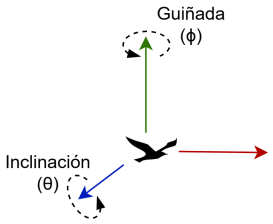

Figura 3.8: Ángulos de guiñada (ϕ) e inclinación (θ) empleados para modelar el giro gradual de cada ave.

5. Actualización de posición: Finalmente, se integra la posición utilizando la velocidad corregida por los límites angulares y lineales:

$$\mathbf{p}_i(t + \Delta t) = \mathbf{p}_i(t) + \mathbf{v}_i(t + \Delta t) \cdot \Delta t \tag{3.8}$$

En resumen, el modelo incorpora las siguientes restricciones de movimiento: velocidad máxima $v_{\max}$, aceleración máxima $a_{\max}$, velocidad angular máxima $\omega_{\max}$ y orientación gradual mediante $\theta_i(t + \Delta t)$.

Cabe señalar que dichas restricciones introducen cierta divergencia entre el comportamiento "ideal" formulado por el modelo de boids y el movimiento final. Por ejemplo, la fuerza de alineamiento depende de la dirección de la velocidad promedio de la bandada, pero al interpolar los cambios de dirección mediante un límite de velocidad de giro se introduce una discrepancia entre la dirección deseada y la real en ciertos momentos de la simulación, se pueden formular analogías similares para las fuerzas de cohesión y separación.

3.3.4. Emergencia de comportamiento colectivo

El comportamiento colectivo emerge de las interacciones locales entre individuos sin requerir coordinación centralizada. Las aves toman decisiones basándose únicamente en información local: posiciones y velocidades de vecinas cercanas, ubicación de árboles visibles y detección de amenazas dentro de su radio de percepción. Esta arquitectura descentralizada genera patrones complejos como la formación espontánea de bandadas y sub-grupos, la sincronización de movimientos de evasión y la distribución eficiente entre fuentes de alimento.

Este enfoque multi-escala, que combina decisiones individuales con dinámicas emergentes de grupo, provee una base para la simulación de bandadas realistas, dando soporte al desarrollo de estrategias de pastoreo asistidas por drones.

3.4. Controlador de dron

El controlador de dron se encarga de ejecutar toda la lógica necesaria para manejar el comportamiento autónomo de cada miembro de la flota. Para esto,

se ejecuta una instancia del controlador por cada agente pastor que participe en la simulación. A su vez, cada controlador se comunica con una instancia particular de PX4 con el objetivo de enviar las instrucciones de vuelo. La figura 3.9 muestra 3 instancias del controlador en funcionamiento.

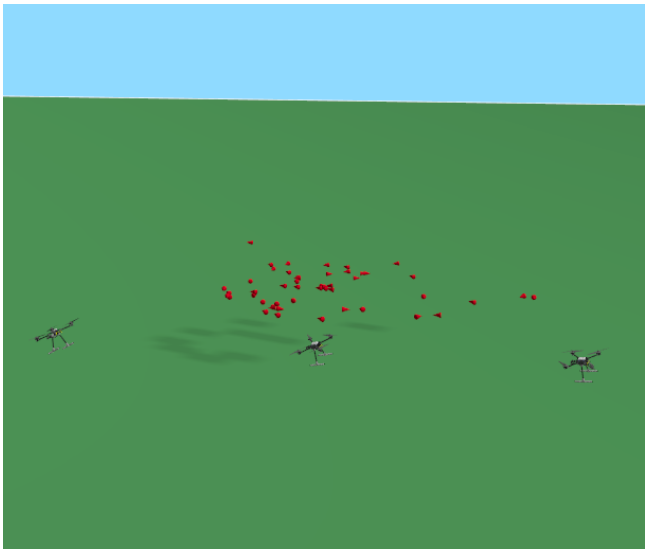

Figura 3.9: Una flota de 3 drones en formación coordinada dentro del ambiente de Gazebo.

Como se mencionó previamente, cada instancia del controlador corresponde a un nodo ROS 2, lo cual permite un intercambio constante de información con los demás nodos que componen el sistema. En particular, el controlador recibe las posiciones de los miembros de la bandada del modelo descrito anteriormente. Por otro lado, intercambia la posición y estado del dron con los demás nodos de controlador que se encuentren activos.

3.4.1. Etapas de la misión de vuelo

Las etapas definidas para la misión de vuelo son: Inicial, Armado, En espera y Vuelo. La figura 3.10 muestra el orden de ejecución de estas etapas.

Inicial Al comenzar la simulación, el controlador se encuentra en estado "Inicial". En este estado, se realizan las operaciones de inicialización del dron, esto constituye enviar la instrucción de armado y despegue a PX4. Una vez que se confirma el armado y despegue se pasa al estado de "Armado".

Armado En el estado de "Armado", el dron asciende hasta llegar a una cierta altura z_{segura} que corresponde con la altitud mínima para evitar colisiones con los cultivos. Cuando se detecta que se alcanza dicha altitud, se pasa al estado de "En espera".

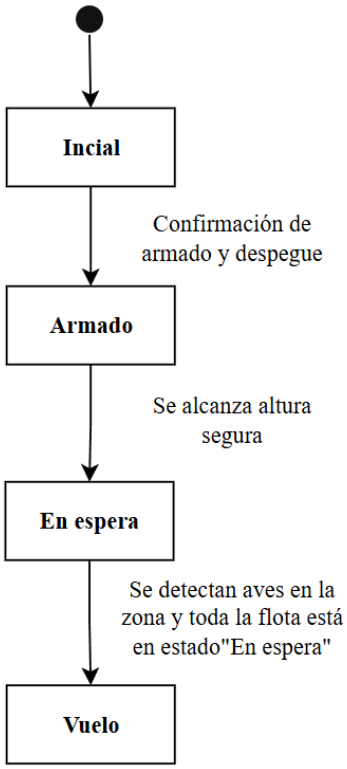

Figura 3.10: Diagrama de estados de la misión de vuelo. Para cada uno se indica la condición de transición.

En espera En este punto los drones mantienen su posición mientras aguardan que el resto de la flota haya llegado al mismo estado. Esto sirve para coordinar el comienzo del vuelo y asegurar que el proceso de pastoreo comience solamente cuando todos los drones estén listos. Una vez que se cumplen estas condiciones, se pasa al estado de "Vuelo".

Vuelo Durante la etapa de vuelo se ejecuta toda la lógica relativa al proceso de pastoreo.

3.4.2. Diseño general de la solución

Una vez comenzada la etapa de vuelo, se determina el movimiento que debe seguir el dron para pastorear a la bandada. Para esto, se calcula la velocidad objetivo en cada instante.

Se describe a continuación el orden de las operaciones que se realizan para calcular dicha velocidad:

- 1) Actualizar las posiciones de la bandada y la flota: Se reciben estos datos a través de tópicos ROS.
- 2) Calcular posiciones de pastoreo: Se utiliza una de las soluciones implementadas para obtener un conjunto de objetivos $\{p_1, \ldots, p_m\} \in \mathbb{R}^3$.
- 3) Dispersión de puntos de pastoreo: Una vez calculadas las posiciones de pastoreo, se les aplica un algoritmo de dispersión para evitar colisiones en el caso de que se encuentren muy cercanas.
- 4) Asignación de puntos de pastoreo: Se asigna cada punto a un agente, minimizando la distancia total que deben recorrer para llegar a su objetivo.
- 5) Cálculo de velocidad objetivo: En base al punto de pastoreo asignado, se calcula la velocidad que debe tener el dron para llegar a él.

Cada instancia de controlador realiza estos cálculos de forma autónoma, logrando así un esquema de solución distribuida. Esto significa que cada dron determina su propio movimiento en base a los datos que recibe, no se cuenta con un nodo central que dirija el comportamiento de la flota. Por lo tanto, el correcto funcionamiento de este esquema depende de que el dron pueda acceder a los datos de entrada en todo momento (las posiciones de los miembros de la flota y las posiciones de los individuos de la bandada).

En esta instancia se asume que el alcance de la comunicación d_{com} es mayor a la mayor distancia dentro de Z d_{max}, la figura 3.11 muestra una representación de estas distancias. Esto quiere decir que los drones son capaces de recibir la información necesaria sin importar su posición dentro de la zona de cultivos, en instancias futuras puede ser necesario agregar controles adicionales que aseguren que los agentes permanezcan dentro de la zona en la cual pueden recibir estos datos correctamente.

Por otro lado, el diseño de la solución asume que todos los drones cuentan con el mismo conjunto de información en cada instante. Esto implica que no existen diferencias entre las posiciones de las aves que conoce cada dron en cada momento, lo mismo aplica para las posiciones de los miembros de la flota. Partiendo de esta asunción, las distintas instancias del controlador realizan las operaciones detalladas anteriormente, manteniendo coordinación en los resultados de sus cálculos. De esta forma, cada dron es capaz de determinar la asignación de puntos de pastoreo de cada miembro de la flota sin necesidad de emitir instrucciones explícitas de coordinación, a este mecanismo se lo conoce como coordinación implícita.

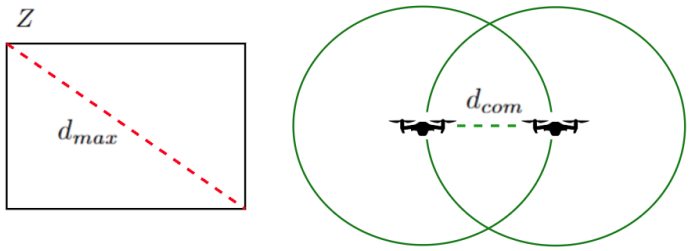

Figura 3.11: A la izquierda representación de d_{max}, a la derecha representación de d_{com}.

Cabe aclarar que los cálculos realizados de la etapa 1 a la 4 coinciden para todos los agentes mientras que la etapa 5 diverge ya que se asigna un objetivo distinto a cada uno y, por lo tanto, el cálculo de su velocidad también es distinto.

Otro aspecto relevante de la coordinación en un sistema robótico distribuido es la sincronización entre agentes. Puede ser deseable contar con mecanismos que aseguren que los agentes tomen decisiones al mismo tiempo para asegurar una operativa sincronizada. En esta instancia no se consideró necesaria la implementación de dichos mecanismos.

El objetivo del diseño descrito es proveer una solución escalable a la vez que robusta. En este sentido, se evitan problemas clave del diseño centralizado (Jawhar et al., 2018):

- El funcionamiento del sistema no depende de un único nodo central. Si ocurren fallas en un nodo, el sistema puede seguir funcionando sin afectar el comportamiento de los demás.
- Se distribuyen los costos computacionales de comunicación entre nodos, de forma que la carga no se concentra en un nodo central. Esto permite una mejor escalabilidad en cantidad de agentes.

Por otro lado, la coordinación implícita descrita anteriormente impone restricciones significativas sobre la disponibilidad de la información para los agentes ya que deben contar con acceso al mismo conjunto de datos en todo momento. Esto implica que el diseño es sensible a problemas típicos de comunicación como son la pérdida o retraso de paquetes. Se deberán tener en cuenta todos estos aspectos si se busca implantar este sistema en un entorno real y a la hora de integrar el módulo de detección de aves.

A continuación se describen las etapas mencionadas anteriormente en mayor detalle. Se empieza por describir el cálculo de posiciones de pastoreo, para el cual se diseñaron e implementaron varios métodos de resolución.

3.4.3. Algoritmo de Strömbom

Se decidió utilizar el algoritmo de Strömbom como base para el cálculo de posiciones de pastoreo debido a que se encontraron múltiples estudios que reportaban resultados satisfactorios con este método (Hoshi et al., 2018; Long et al., 2020; Strömbom et al., 2014). También, por lo explicado en el capítulo anterior sobre técnicas preventivas y conductivas, resulta preferible partir de una solución que no imponga restricciones tan severas sobre el sistema de detección. Recordar que para el caso de la solución presentada en (Paranjape et al., 2018) se requiere conocer las posiciones de las aves en un área mayor que Z, lo cual impacta directamente en la infraestructura requerida para el sistema de detección (con cuyo diseño no se cuenta en esta etapa del proyecto). Para el algoritmo de Strömbom, basta con conocer las posiciones de las aves que se encuentren dentro de Z.

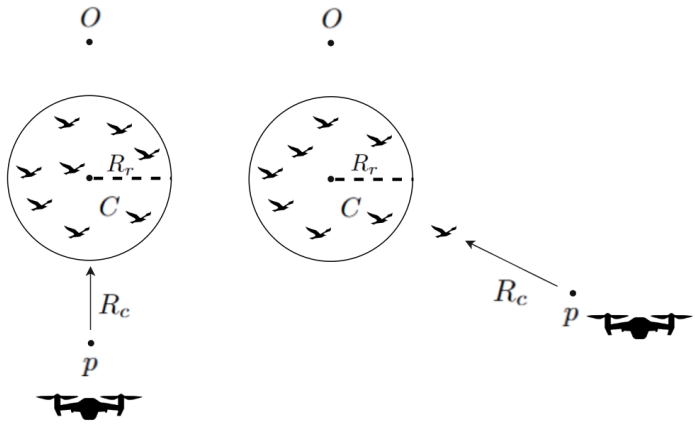

Figura 3.12: Diagrama de funcionamiento del algoritmo de Strömbom. A la izquierda se visualiza el comportamiento del estado de Conducción y a la derecha el de Recolección.

El funcionamiento del algoritmo de Strömbom es el siguiente: se cuenta con un punto objetivo O, un radio de recolección R_r, un radio de conducción R_c y el conjunto de aves detectadas L', en base a las posiciones de L' se calcula el centro de masa de la bandada C. Para realizar el pastoreo se alterna entre dos estados: Recolección y Conducción, la figura 3.12 muestra un diagrama de estos comportamientos. Mientras que el agente se encuentra en el estado de Conducción, se posiciona de forma de ejercer una fuerza sobre C que dirija a la bandada hacia el punto objetivo O. Cuando se detecta que existen individuos de la bandada que se encuentran a una distancia superior a R_r del centro de masas, se pasa al estado de Recolección. En dicho estado, el agente se posiciona de forma de ejercer una fuerza sobre el individuo l_f más apartado del centro de masas que lo dirija hacia C, cuando se detecta que no hay más individuos por

fuera del radio descrito anteriormente, se vuelve al estado de Conducción.

Para posicionar al agente y ejercer las fuerzas necesarias se calcula un vector dirección D como la resta entre p_{lf} y C para el estado de Recolección o la resta entre C y O para el estado de Conducción, luego se lo normaliza para obtener el vector ND. Finalmente, la posición de pastoreo p se calcula como $p_{lf} + ND.R_c$ para la Recolección o $C + ND.R_c$ para la Conducción. La figura 3.13 muestra un diagrama de estos términos.

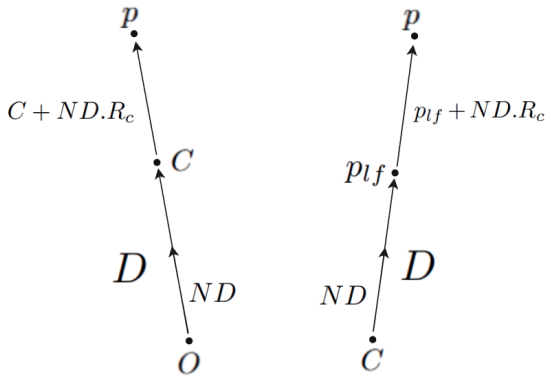

Figura 3.13: Diagrama de cálculo de posiciones de pastoreo para el algoritmo de Strömbom. A la izquierda el cálculo de posición de Conducción, a la derecha el cálculo de posición de Recolección.

Cabe resaltar la importancia del parámetro R_c, ya que determina la distancia desde la cual se ejerce la presión a la bandada en ambos estados. Como se puede ver en (Paranjape et al., 2018) esta distancia es sumamente importante para la efectividad del proceso de pastoreo, ya que de ser muy pequeña se ejerce demasiada presión sobre las aves y se corre el riesgo de dividir la bandada. Por el contrario, si la distancia es demasiado grande el agente no ejerce la presión suficiente sobre la bandada como para lograr su desplazamiento.

Pseudocódigo Strömbom

El Algoritmo 4 presenta el pseudocódigo del algoritmo de Strömbom original. Como se puede ver, es un algoritmo cuyo orden de tiempo de ejecución es $O(n)$ siendo n la cantidad de aves de la bandada, ya que debe recorrer toda la lista de posiciones de las aves para calcular C y p_{lf}.

Una limitación clara de este algoritmo es que está pensado para escenarios con un solo pastor. Como se mencionó anteriormente, el foco de este proyecto es la aplicación de técnicas multi-agente para resolver el problema de pastoreo. Se busca, entonces, extender el algoritmo de Strömbom para que pueda ser ejecutado por múltiples agentes en simultáneo. Para esto se deben buscar formas

de particionar el problema en un conjunto de tareas que cada agente individual pueda resolver.

Algoritmo 4 Algoritmo de Strömbom

Entrada: Conjunto de aves detectadas $L' = \{l_1, l_2, \ldots, l_n\}$, Punto objetivo O, Radio de recolección R_r, Radio de conducción R_c

1: Calcular centro de masa:

$$C \leftarrow \frac{1}{n} \sum_{i=1}^{n} p_{li}(t)$$

2: $l_f \leftarrow$ ave más lejana de C

3: **if** $\|p(l_f) - C\| > R_r$ **then** ▷ Recolección

4: Calcular posición de Recolección:

$$p \leftarrow p_{lf}(t) + R_c \cdot \frac{p_{lf}(t) - C}{\|p_{lf}(t) - C\|}$$

5: **else** ▷ Conducción

6: Calcular posición de Conducción:

$$p \leftarrow C + R_c \cdot \frac{C - O}{\|C - O\|}$$

7: **end if**

8: **return** p

A raíz de esto se diseñaron 3 soluciones algorítmicas para ser utilizadas por múltiples agentes. La primera involucra dividir la bandada en varios clusters para luego aplicar el algoritmo de pastoreo clásico a cada uno de ellos. La segunda se enfoca en formular las tareas en base a los estados que define Strömbom (Conducir y Recolectar), paralelizarlas y distribuirlas entre los agentes. La tercera solución combina los dos enfoques anteriores en un solo algoritmo.

3.4.4. Solución 1: Clustered Strömbom

Si se cuenta con una flota de m drones, la idea central de esta solución consiste en dividir a L' en m sub-bandadas $SL_1, \ldots, SL_m$ mediante un algoritmo de clustering para luego aplicar Strömbom a cada una de ellas y obtener las posiciones de pastoreo para cada cluster $\{p_1, ..., p_m\}$. La figura 3.14 muestra un diagrama del funcionamiento de esta solución para 3 drones.

El algoritmo de esta solución es equivalente al Algoritmo 4 con la diferencia de que recibe las posiciones de la sub-bandada en lugar de la bandada entera.

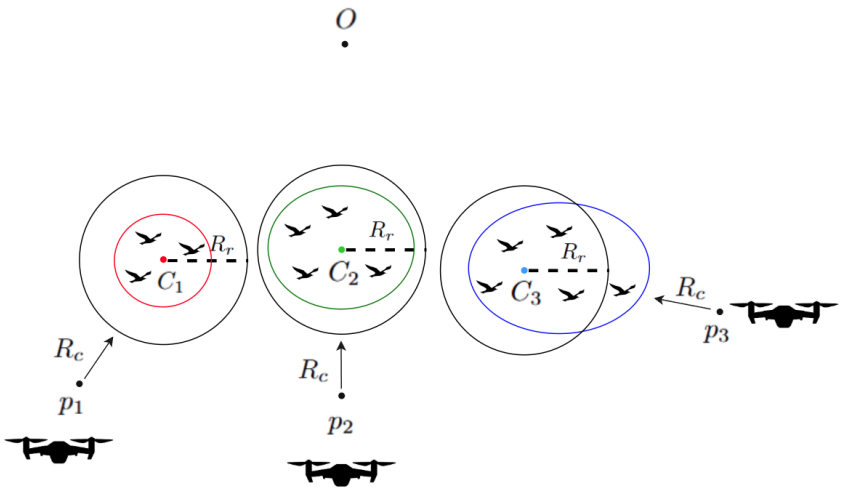

Figura 3.14: Diagrama de funcionamiento de Clustered Strömbom para 3 agentes. Los círculos coloreados corresponden a la partición en clusters de la bandada con sus respectivos centros de masa C_1, C_2, C_3. Los puntos de pastoreo son p_1, p_2, p_3.

Alternativas de algoritmos de clustering - Cantidad fija de clusters

- Sectores Angulares

  El método de Sectores Angulares divide el espacio circundante al centro geométrico de la bandada en un número fijo de secciones como se muestra en la figura 3.15. Notar que para esto, se trabaja únicamente con las coordenadas x e y de las posiciones de cada ave, utilizando una proyección bidimensional del espacio. En particular, se divide dicho espacio en m sectores siendo m la cantidad de drones. Luego, cada ave es asignada a una sección de acuerdo con el ángulo formado entre su posición y el centro de la bandada, utilizando coordenadas polares.

  Desde el punto de vista computacional, este enfoque presenta una alta eficiencia, con una complejidad de $O(n)$, ya que cada ave se procesa solo una vez. Esto lo convierte en una opción especialmente adecuada para simulaciones de bandadas grandes.

  Una de las principales ventajas de este método es su simplicidad en la implementación, así como la distribución angular uniforme que garantiza. No obstante, al no tener en cuenta la densidad ni las distancias internas entre los individuos, puede generar agrupamientos desbalanceados en bandadas con configuraciones no circulares o concentraciones locales de aves. Más aún, en caso de que existan sectores desocupados, los clusters correspondientes quedan vacíos. En tales casos, el algoritmo no es capaz de adaptarse a la forma irregular de la bandada, lo que limita su efectividad

en escenarios más complejos.

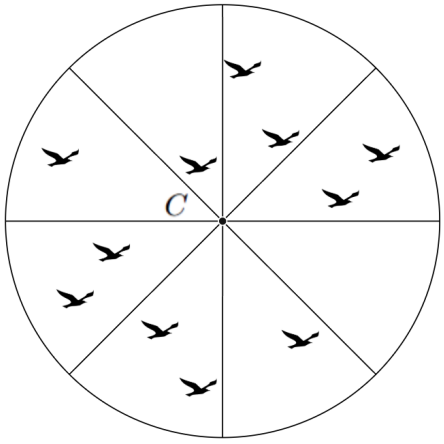

Figura 3.15: Diagrama de la división generada por el algoritmo de sectores angulares, para una división en 8 grupos.

- Sectores Angulares con Asignación Fija de Aves

  En esta variante del método de Sectores Angulares, las aves se ordenan según el ángulo respecto al centro de la bandada y luego la lista de aves se divide para formar sectores. Se mantiene la división en m secciones pero cada uno tiene un tamaño variable, dependiendo de la dispersión de las aves. Este enfoque asegura que todos los sectores estén ocupados y que cada uno contenga la misma cantidad de aves. El algoritmo se ejecuta en $O(n \times log(n))$ ya que se debe ordenar la lista de posiciones.

  La variante conserva las ventajas del enfoque tradicional, como su simplicidad en la implementación, pero añade la ventaja de evitar sectores vacíos. Sin embargo, una de sus desventajas es que, al no tener en cuenta la proximidad entre las aves, dentro de cada sector pueden quedar aves dispersas, lo que puede generar agrupamientos menos coherentes, especialmente en bandadas con configuraciones complejas. Por otro lado, al requerir un ordenamiento de los ángulos introduce mayor complejidad computacional.

- K-means

  El algoritmo K-means (Hartigan & Wong, 1979) agrupa a las aves en K clusters mediante un proceso iterativo basado en centroides, la figura 3.16 muestra un ejemplo para 3 clusters. Inicialmente, se seleccionan K centroides de manera aleatoria y, en cada iteración, cada ave es asignada al centroide más cercano, recalculando luego las posiciones de los centroides hasta alcanzar la convergencia.

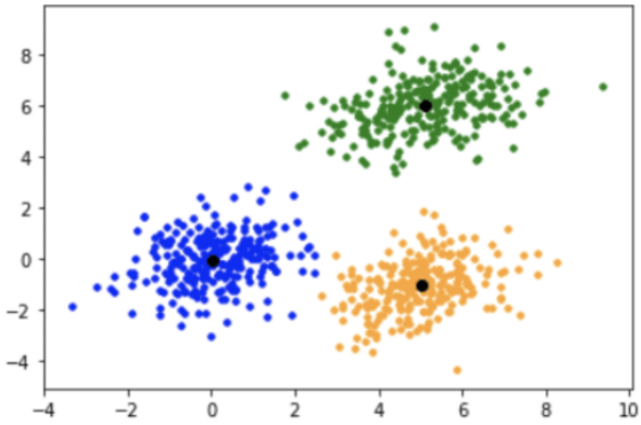

Figura 3.16: Diagrama de una asignación generada por el algoritmo K-means, para una división en 3 grupos.

En términos de eficiencia, K-means presenta una complejidad de $O(n \times k \times kmeans_i)$, donde n es el número de aves, k es la cantidad de clusters (igual a m) y $kmeans_i$ representa el número máximo de iteraciones. Este enfoque resulta adecuado para bandadas de tamaño medio o grande, siempre que se pueda definir un número apropiado de clusters.

El algoritmo produce agrupamientos compactos y bien definidos. Sin embargo, supone que los grupos tienen una distribución aproximadamente esférica para calcular la asignación, lo que puede resultar problemático en bandadas con estructuras irregulares, donde los grupos de aves no se distribuyen de forma uniforme. También es importante destacar que K-means es sensible a la presencia de valores atípicos, lo que puede afectar su rendimiento cuando existen aves aisladas que no forman parte de un grupo claro.

En algunas aplicaciones, se recomienda inicializar los centroides de los clusters de forma aleatoria. En el contexto de la solución que se describe, no se puede utilizar este mecanismo de inicialización ya que provocaría discrepancias entre los cálculos de los drones. Por esta razón, se inicializan los centroides de forma determinista.

Luego de calcular los clusters, se aplica el algoritmo de Strömbom para obtener los puntos de pastoreo deseados. Esta extensión del algoritmo logra distribuir el proceso de pastoreo de forma que cada dron se encarga de una sección de la bandada, dividida según distintos criterios dependiendo del algoritmo de clustering elegido. En principio parecía una buena alternativa que lograba manejar bandadas con alta dispersión, pero rápidamente se encontraron problemas en su funcionamiento. En particular, cuando la bandada se encuentra muy agrupada los drones terminan compitiendo por ubicarse en la misma posición de conducción, causando un agrupamiento sin coordinación y desaprovechando las

capacidades de cada agente. Esto se debe a que las sub-bandadas calculadas en estas situaciones se encuentran muy cerca una de la otra y, por lo tanto, también sus puntos de pastoreo.

Contradictoriamente, esta solución tiene problemas para manejar la situación "ideal" de pastoreo en la cual la bandada se encuentra fuertemente agrupada. Fue claro que se debían incluir mecanismos para solucionar este conflicto, con este objetivo la solución 2 introduce una posible alternativa.

También es importante remarcar que los algoritmos descritos devuelven una cantidad fija de clusters, en el contexto de la solución esta cantidad equivale a m (el tamaño de la flota). Si bien esto facilita la asignación de tareas, la distribución de los individuos de la bandada puede variar ampliamente incluso dentro de una misma instancia de pastoreo. Por esta razón, la división de la bandada en una cantidad fija de clusters puede no ser óptima para todos los casos, la solución 3 descrita más adelante intenta mitigar estas limitaciones.

3.4.5. Solución 2: M-Strömbom

La segunda alternativa se basa en la idea de formular las tareas en base a los estados definidos por el algoritmo clásico de Strömbom. La figura 3.17 muestra un diagrama de funcionamiento de esta solución para 4 drones.

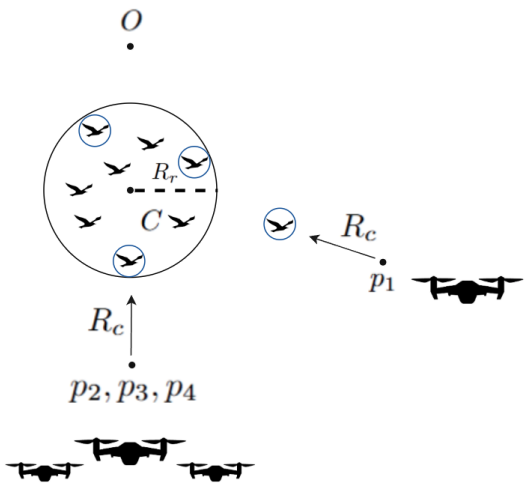

Figura 3.17: Diagrama de funcionamiento de M-Strömbom para 4 agentes. Los círculos azules corresponden a los 4 individuos más lejanos de la bandada, se genera un punto de Recolección p_1 para el ave que está a distancia mayor a R_r de C. Los 3 agentes restantes se encuentran en estado de Conducción y adoptan una formación cuyas posiciones son p_2, p_3, p_4.

Para el estado de Recolección, originalmente se toma el individuo más lejano del centro de masa de la bandada cuando se encuentra a una distancia mayor a

R_r. Para extender esto a m agentes se podrían tomar los k individuos que cumplen esa condición (con $k \leq m$) y calcular los puntos de recolección $\{r_1, \ldots, r_k\}$. Si se cumple que $k < m$ se asigna el resto de los $m - k$ drones al estado de Conducción.

En el estado de Conducción, originalmente se posiciona al agente de forma de ejercer presión sobre el centro de masa de la bandada para conducirla hacia el objetivo. Para extender este comportamiento a múltiples agentes, se puede hacer que la flota adopte una formación como se propone en (Shi & Wang, 2022) (figura 3.18). Calculando las posiciones que componen esta formación se obtienen $\{c_1, \ldots, c_{m-k}\}$. De esta manera, se expande la zona de influencia de los drones sobre la bandada.

Finalmente: $\{p_1, \ldots, p_m\} = \{r_1, \ldots, r_k\} \cup \{c_1, \ldots, c_{m-k}\}$

Notar que se obtiene un punto de pastoreo por cada dron de la flota, esto facilita el proceso de asignación de puntos descrito en la sección 3.4.8.

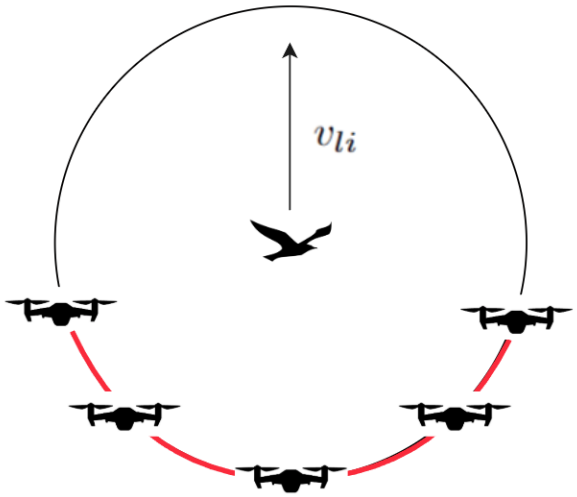

Figura 3.18: Esquema de una de las formaciones propuestas en (Shi & Wang, 2022) para el pastoreo de aves.

Formación de conducción

Se especifica la formación que adoptan los drones en estado de Conducción en el algoritmo de M-Strömbom. Para lograr la formación (representada en la figura 3.19), se ubica a los drones en una línea con dirección perpendicular al vector $\overrightarrow{CO}$, se denomina D a esta dirección. El centro de la formación es p, el punto de pastoreo calculado para el estado de Conducción de la misma manera que en el algoritmo original de Strömbom. Finalmente, se posiciona a los drones a lo largo de D con una separación s, esta separación es proporcional a la desviación estándar de las posiciones de las aves con un factor f_s. De esta

manera, el área que abarca la formación se ajusta dinámicamente a la dispersión de la bandada.

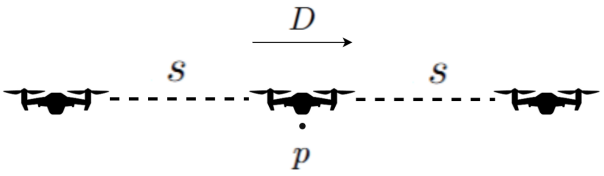

Figura 3.19: Esquema de formación en línea y sus componentes principales.

Las posiciones de esta formación $\{p_1, \ldots, p_n\}$ se calculan como:

$$p_i = p + D.(i - 1 - n/2).f_s$$

Pseudocódigo M-Strömbom

El Algoritmo 5 presenta el pseudocódigo de M-Strömbom. El orden de dicho algoritmo tiene un incremento de complejidad en comparación al original ya que se necesita mantener los m individuos más lejanos a C al recorrer la lista de posiciones. Por lo tanto, el orden de complejidad del algoritmo es $O(n \times log(m))$ siendo n la cantidad de aves de la bandada y m la cantidad de puntos objetivo solicitados [1].

Notar que, con estas modificaciones, se elimina la alternación de estados que sucede en el algoritmo original para pasar a un esquema en el cual se consideran ambos comportamientos como tareas paralelizables.

Con la introducción de una formación explícita para el estado de conducción, se corrigió el problema de agrupamiento observado en la solución anterior. Más aún, adaptando esta formación a la dispersión de la bandada hace que la presión de los drones se distribuya de forma uniforme sobre la bandada.

Sin embargo, se observó que esta solución tiene algunos problemas para lidiar con los casos en los que algunos individuos se separan de la bandada principal. En estos casos, M-Strömbom (al igual que el algoritmo original) propone abandonar el estado de conducción para centrarse en perseguir a los individuos aislados. Si bien lo mencionado busca asegurar la cohesión de la bandada, se pierde todo tipo de presión sobre el grupo (o grupos) principales, permitiendo que las aves se muevan libremente y perdiendo una parte considerable del desplazamiento que se venía efectuando.

[1] Para lograr este tiempo de ejecución se utiliza una cola de prioridad que mantiene la referencia de los m individuos más lejanos a medida que se calcula C.

Algoritmo 5 Algoritmo de M-Strömbom

Entrada: Conjunto de aves detectadas $L' = \{l_1, l_2, \ldots, l_n\}$, Punto objetivo O, Radio de recolección R_r, Radio de conducción R_c, Cantidad de puntos M

Calcular centro de masa:

$$C \leftarrow \frac{1}{n} \sum_{i=1}^{n} pl_i(t)$$

$\{l_{f1}, l_{f2}, \ldots, l_{fm}\} \leftarrow$ aves más lejanas de C
$j \leftarrow 1$
for all $l_{fi} \in \{l_{f1}, l_{f2}, \ldots, l_{fm}\}$ **do** ▷ Recolección
    if $\|p_{lfi}(t) - C\| >_r$ **then**
        Calcular posición de Recolección:

$$p_j \leftarrow p_{lfi}(t) + R_c \cdot \frac{p_{lfi}(t) - C}{\|p_{lfi}(t) - C\|}$$

        $j \leftarrow j + 1$
    end if
end for all
if $j - 1 < M$ **then** ▷ Conducción
    Obtener posiciones de formación:

$$\{p_j, \ldots, p_M\} \leftarrow Formacion(M - j - 1, f_s)$$

end if
return $\{p_1, p_2, \ldots, p_M\}$

Este problema no se observaba con el algoritmo anterior, de forma que se intentó combinar algunos aspectos de cada extensión para formular la solución descrita a continuación.

3.4.6. Solución 3: Clustered M-Strömbom

La tercera alternativa se basa en combinar los dos enfoques anteriores. Para esto se divide la bandada en una cantidad variable de clusters y luego se aplica M-Strömbom a cada uno de ellos. La figura 3.20 muestra un diagrama de funcionamiento de esta solución para 5 drones.

Alternativas de algoritmos de clustering - Cantidad variable de clusters

Para empezar, se debe seleccionar un algoritmo que permita dividir la bandada en una cantidad variable de clusters dependiendo de la mejor agrupación.

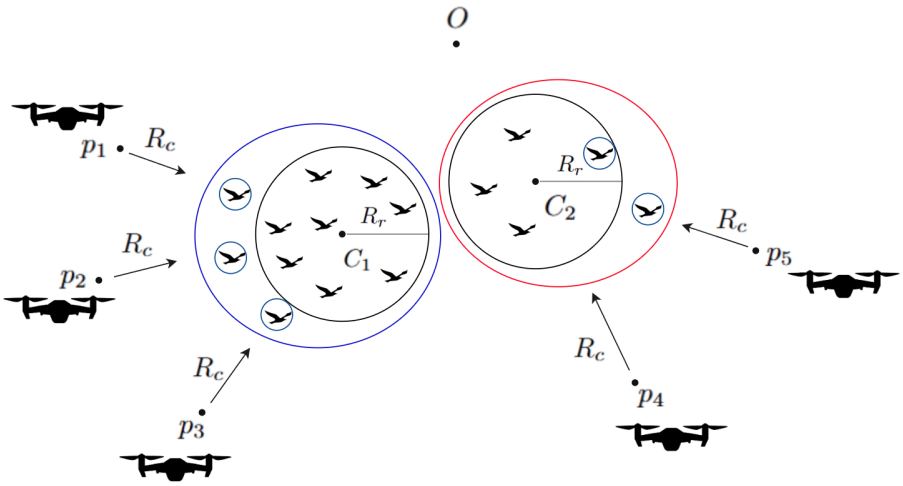

Figura 3.20: Diagrama de funcionamiento de Clustered M-Strömbom. Se cuenta con 5 agentes y 2 clusters a los cuales se les asignan 3 y 2 drones respectivamente.

- X-Means

  El algoritmo X-Means (Pelleg & Moore, 2000) amplía el enfoque de K-means con el objetivo de determinar de manera automática el número óptimo de clusters para una bandada, se establecen los límites de este número con los parámetros k_{min} y k_{max}. Comienza con k_{min} grupos y evalúa iterativamente la división de cada clúster en dos subgrupos, utilizando un análisis PCA (análisis de componentes principales). Este proceso se basa en el criterio estadístico BIC (Bayesian Information Criterion), el cual permite equilibrar la mejora en el ajuste del modelo con su complejidad, evaluando cuál división resulta en una distribución más adecuada de las aves.

  Desde el punto de vista computacional, X-Means presenta una complejidad superior a la de K-means, aproximadamente $O(n \times k_{max} \times kmeans_i \times log(k_{max}))$.

  La ventaja principal de X-Means radica en su flexibilidad y capacidad de ajustar el número de clusters según la estructura de la bandada. No obstante, mantiene la preferencia por agrupamientos esféricos y requiere un ajuste más preciso de parámetros, lo que puede dificultar su configuración en comparación con K-means.

- DBSCAN

  El algoritmo DBSCAN (Density-Based Spatial Clustering of Applications with Noise) (Ester et al., 1996) agrupa a las aves en función de la densidad espacial local. Un ave se considera un punto central si tiene al menos un número mínimo de vecinas dentro de un radio determinado. Los clusters se forman al conectar estos puntos centrales con sus vecinas. En el contexto

de la solución, el número mínimo de vecinas debe fijarse en 1 ya que de otra forma, individuos aislados de la bandada serían ignorados por la asignación (recordar que se busca expulsar a todos los individuos). La figura 3.21 muestra un ejemplo de agrupación con este método.

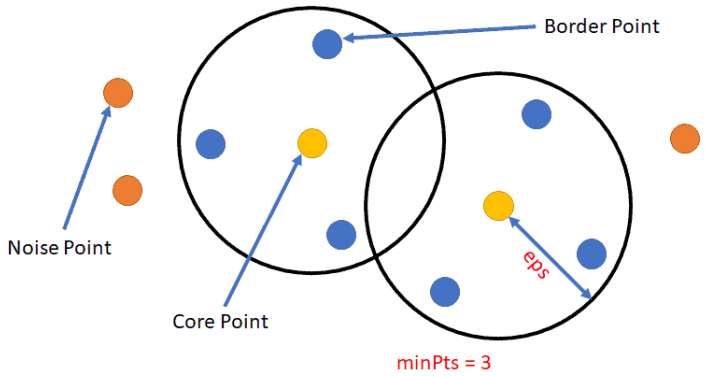

Figura 3.21: Diagrama de funcionamiento del algoritmo DBSCAN, la distancia eps indicada en la imagen corresponde con el radio y minPts con el número mínimo de vecinas.

En cuanto a la eficiencia computacional, DBSCAN tiene una complejidad de $O(n^2)$ en su implementación básica. Esta complejidad hace que sea más adecuado para bandadas de tamaño moderado.

Una de las principales ventajas de DBSCAN es su capacidad para detectar agrupamientos arbitrarios. Esto lo hace especialmente útil para bandadas con una distribución irregular. Sin embargo, la selección de los parámetros adecuados (radio de vecindad y número mínimo de puntos) es crucial, y el algoritmo puede presentar dificultades en bandadas donde coexisten zonas muy densas (como el centro del grupo) con zonas más dispersas (como la periferia), ya que los parámetros fijos del algoritmo no pueden optimizarse simultáneamente para ambos tipos de distribución espacial (Jeong-Hun et al., 2019).

Por último, cabe señalar que una de las particularidades de DBSCAN es que retorna una cantidad no acotada de clusters. Esto significa que pueden existir casos en los que el algoritmo retorna un número de clusters h mayor a la cantidad de agentes disponibles m. Esto representa un problema ya que se debería determinar un método para priorizar m de las sub-bandadas obtenidas ya que no se cuenta con la cantidad suficiente de drones para asignar uno a cada una. Si bien esta puede ser una línea interesante de investigación, en esta instancia se decidió dejar por fuera esta problemática. Si se detecta que $h > m$ se reemplaza la asignación de clusters por la asignación de K-means con $k = m$ al igual que en la solución 1.

Asignación proporcional de agentes

Luego de aplicar alguno de los algoritmos de clustering descritos anteriormente, se cuenta con k sub-bandadas (con $1 \leq k \leq m$) siendo m el tamaño de la flota. El siguiente paso consiste en decidir la cantidad de agentes que se le debe asignar a cada sub-bandada, con la restricción que se debe asignar al menos un agente a cada una. Además, es deseable que la cantidad de agentes asignados sea proporcional al porcentaje de individuos que pertenece a cada grupo.

Para lograr una distribución con estas características se asigna un agente por sub-bandada, luego se utiliza el Método del Resto mayor (Hare, 1859) para asignar los agentes restantes. Dicho método provee una distribución proporcional de una cierta cantidad de recursos m a una cierta cantidad de tareas k con pesos asignados. En el contexto de esta solución, se toma m como la cantidad de agentes, k como la cantidad de sub-bandadas y sus pesos equivalen a la cantidad de individuos en cada una de ellas. Este método devuelve una distribución en tiempo $O(k \times log(k))$.

Aplicación de M-Strömbom

Luego de aplicar los pasos anteriores, se cuenta con una división de la bandada en k grupos y, para cada uno de estos grupos, una cantidad de agentes que se encargarán de cada uno. Finalmente, se aplica el algoritmo de M-Strömbom a cada grupo utilizando la cantidad de agentes mencionada como el parámetro M. Al combinar los puntos obtenidos de cada llamada a M-Strömbom se obtienen las posiciones de pastoreo deseadas (un punto por cada dron de la flota).

Esta solución logra resolver las problemáticas identificadas anteriormente, de esta forma, provee mecanismos para lidiar con bandadas de distinta densidad y/o extensión. También permite tratar con individuos aislados sin sacrificar el desplazamiento al grupo principal y ejerciendo una presión uniforme sobre todas las aves.

La mayor desventaja de esta solución radica en su eficiencia computacional, requiriendo de algoritmos de clustering más complejos y de aplicar el algoritmo de M-Strömbom múltiples veces.

3.4.7. Dispersión de puntos de pastoreo

Luego de obtener los puntos de pastoreo con alguno de los métodos descritos anteriormente, se aplica un método de repulsión por pares sobre el conjunto de puntos con el objetivo de evitar posibles colisiones entre los drones. Para esto, se itera entre todos los pares de puntos de pastoreo (p_i, p_j), si se detecta que $d = \|p_i - p_j\| < d_{min}$ se aplica un desplazamiento opuesto de módulo $d_{min} - d$ a cada uno de estos puntos. Este proceso se repite hasta que el desplazamiento total efectuado en una iteración sea menor a cierto umbral o si se supera un número máximo de iteraciones max_{iter}. El algoritmo se ejecuta en tiempo $O(m^2 \times max_{iter})$ siendo m la cantidad de puntos de pastoreo.

3.4.8. Asignación de objetivos

Una vez obtenidas las posiciones de pastoreo con su respectivo desplazamiento, se debe asignar cada una de ellas a un dron. Para lograr esto se utiliza el Método Húngaro (Kuhn, 1955), un conocido algoritmo de optimización para resolver problemas de asignación. Dicho método opera sobre una matriz de costos para encontrar una asignación que minimice el costo total. En el contexto de las soluciones presentadas, la matriz de costos corresponde a la distancia entre cada agente y los puntos de pastoreo calculados. Minimizar esta distancia equivale a minimizar la cantidad de tiempo de vuelo que debe transcurrir para que todos los drones lleguen a su objetivo.

Este método cuenta con otras cualidades deseables dado el contexto del problema. Por un lado logra una asignación determinista dada una cierta matriz de costos, esto es importante para mantener la coordinación implícita del esquema distribuido. Por otro lado, los mecanismos del algoritmo resuelven las situaciones de empate automáticamente y de forma determinista por lo que, en caso de que dos agentes se encuentren a la misma distancia de un cierto punto de pastoreo, no habrá problemas de coordinación en la asignación de dicho punto.

El método Húngaro resuelve los problemas en tiempo $O(m^3)$ siendo $m \times m$ las dimensiones de la matriz de costo, es requerido que la matriz sea cuadrada para aplicar el método. Por esta razón, los algoritmos de pastoreo descritos anteriormente fueron diseñados para retornar una cantidad de puntos de pastoreo igual al tamaño de la flota.

3.4.9. Cálculo de velocidad objetivo

Finalmente, se calcula la velocidad deseada para cada dron. A la hora de describir el movimiento de un dron se debe especificar su velocidad lineal y su actitud, que se compone por 3 rotaciones en cada uno de los 3 ejes de $\mathbb{R}^3$ como se muestra en la figura 3.22. Para esto, PX4 ofrece un modo de especificar el movimiento conocido como *velocity setpoint*. En este modo, el controlador de vuelo recibe la velocidad lineal y ángulo de guiñada deseados, calculando automáticamente las velocidades angulares de cabeceo y alabeo. En este trabajo se decidió no especificar el ángulo de guiñada, de forma que los drones no presentan rotaciones en este eje respecto a su pose inicial. En instancias futuras puede ser necesario definir dicha orientación, por ejemplo para alinear una cámara frontal con la dirección de vuelo.

Para el cálculo de la velocidad lineal se toman en cuenta dos aspectos: el punto objetivo y la cercanía a otros drones de la flota. Se busca dirigir al dron hacia su objetivo a la vez que esquivar a otros agentes si se detecta peligro de colisiones.

Si el dron d_i se encuentra a una distancia menor a cierto radio d_{segura} de otros drones $d_1, \ldots, d_k$, el vector velocidad se calcula como:

$$\sum_{j \in \{1,\ldots,k\}} ((p_{di} - p_{dj})/\|p_{di} - p_{dj}\|).v_{max}$$

donde v_{max} es la velocidad máxima de movimiento del dron.

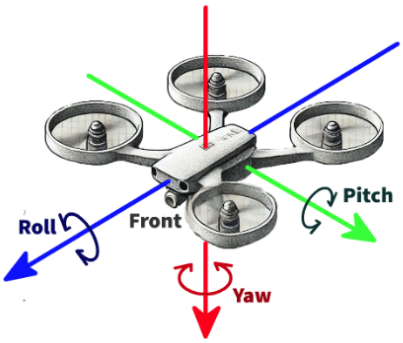

Figura 3.22: Diagrama de las rotaciones que definen la actitud de un dron: guiñada (yaw), cabeceo (pitch) y alabeo (roll).

Si el dron d_i no se encuentra en la cercanía de otros drones y su objetivo asignado es p_i, el vector velocidad se calcula como:

$$M.(p_i - p_{di})/\|p_i - p_{di}\|$$

donde M es el módulo de velocidad. Para determinar este módulo, se tiene en cuenta la distancia $d = \|p_i - p_{di}\|$ (distancia entre el dron y su objetivo) y un cierto módulo de velocidad v de recorrida. Luego, $M = min(v, d.f)$ donde f es un factor que determina a qué distancia del objetivo se empieza a disminuir la velocidad. Por otro lado, si el objetivo se encuentra a una distancia menor a cierto umbral u, el módulo es nulo. De esta manera, el dron se mueve con velocidad v para recorrer grandes distancias y empieza a desacelerar a medida que se acerca a su objetivo.

3.4.10. Observaciones sobre complejidad computacional

Dada la naturaleza del controlador, es importante que sea capaz de ejecutar múltiples veces por segundo el ciclo de operaciones descrito para asegurar su correcto funcionamiento. Esto es vital para que los drones reaccionen correctamente a los cambios en las posiciones de la bandada y se eviten colisiones con otros agentes o el ambiente. Además, PX4 requiere de una frecuencia de actualización de instrucciones de al menos $2Hz$ (Dronecode, 2026), si las instrucciones de vuelo se envían a una frecuencia menor el controlador falla. Se recomienda una frecuencia entre $30Hz$ y $100Hz$ para el correcto funcionamiento de PX4 aunque por lo general $10Hz$ es suficiente, en este proyecto se apuntó a este último valor.

Se debe tener en cuenta entonces la complejidad de los algoritmos utilizados para determinar la escalabilidad de estas soluciones. La etapa de dispersión de puntos de pastoreo tiene complejidad $O(m^2 \times max_{iter})$ y la etapa de asignación de objetivos $O(m^3)$. Estas etapas son comunes a todas las soluciones por lo que se tiene una complejidad base de $O(m^3 + m^2 \times max_{iter})$. Luego, la solución 1

agrega una complejidad de $O(n \times m \times kmeans_i)$ cuando se utiliza K-means y $O(n \times log(n))$ para sectores angulares con asignación fija, la solución 2 agrega una complejidad de $O(n \times log(m))$ y la solución 3 agrega $O(n^2)$ para DBSCAN y $O(n \times m \times kmeans_i \times log(m))$ para X-means. Las figuras 3.23, 3.24 y 3.25 presentan las curvas de nivel de los órdenes de las 3 soluciones para distintos valores de n y m.

En general, se puede ver que la solución más eficiente en cuanto a complejidad es M-Strömbom y los órdenes de las otras soluciones dependen de la configuración que se elija. Las variables que influyen en mayor medida en la eficiencia del sistema son, naturalmente, m (la cantidad de agentes) y n (el tamaño de la bandada). En el marco de este proyecto, no se anticipa el uso de flotas de drones de gran tamaño, por lo que los órdenes cúbicos y cuadráticos de m se ven mitigados por los valores que puede tomar esta variable en implantaciones reales. En cuanto a la cantidad de aves, el mayor riesgo aparece al utilizar DBSCAN, por lo que se deberían considerar los valores típicos de n a los que se enfrenta el sistema antes de optar por esta alternativa.

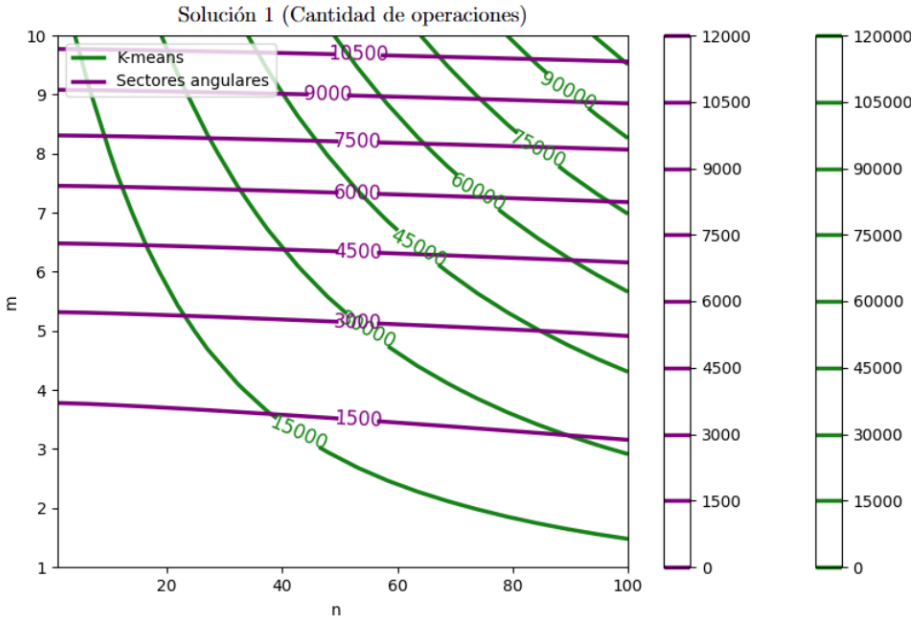

Figura 3.23: Curvas de nivel de complejidad computacional para Clustered Strömbom.

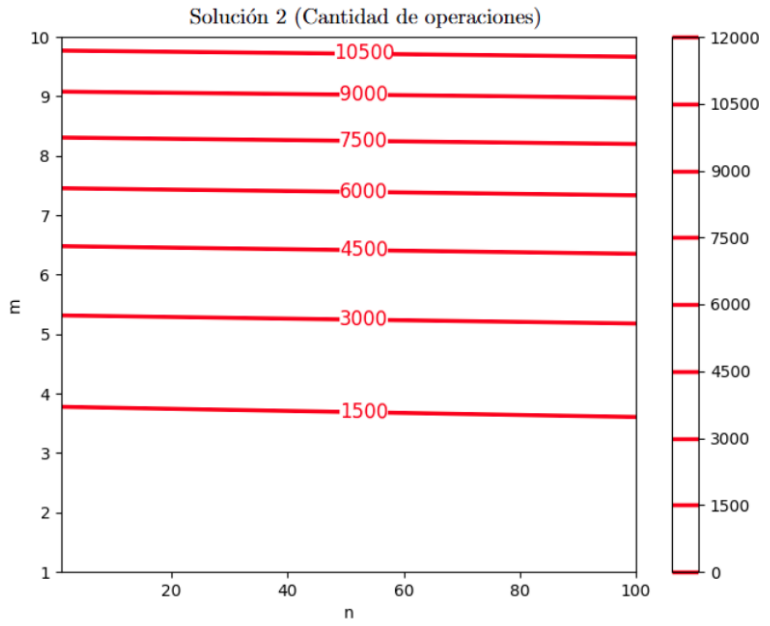

Figura 3.24: Curvas de nivel de complejidad computacional para M-Strömbom.

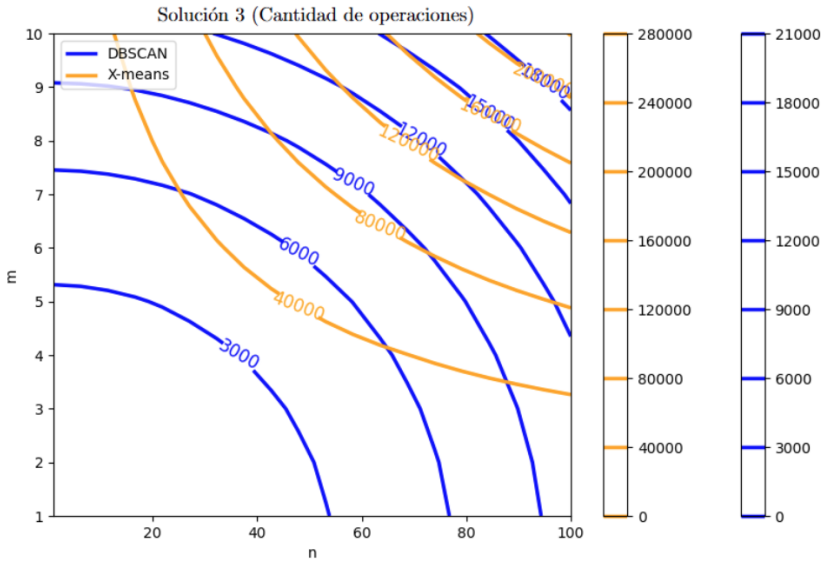

Figura 3.25: Curvas de nivel de complejidad computacional para Clustered M-Strömbom.

Capítulo 4

Experimentación

En esta sección se describe el proceso de experimentación y evaluación del sistema desarrollado. Se diseñaron distintos escenarios para replicar un ambiente típico de cultivos frutales de Uruguay y se utilizaron para simular distintas configuraciones de los parámetros del sistema con el fin de medir su rendimiento en un conjunto de situaciones representativas de la realidad. Para esto, se estandarizaron las condiciones de evaluación para asegurar un criterio uniforme entre las distintas soluciones de pastoreo implementadas.

La evaluación consta de tres etapas: el ajuste paramétrico, la evaluación de las soluciones y el análisis de sensibilidad. En la primera etapa, se realizan múltiples simulaciones del sistema en un escenario reducido para evaluar distintas configuraciones de parámetros que alteran el funcionamiento de los algoritmos de pastoreo, los parámetros estudiados varían en cada solución. La segunda etapa consiste en evaluar el funcionamiento de las soluciones en escenarios completos, utilizando las mejores configuraciones paramétricas obtenidas en la etapa anterior. Finalmente, en la tercera etapa se evalúa el impacto en el desempeño del sistema de variar ciertos parámetros para los cuales no se pudo determinar un valor confiable.

4.1. Entorno de pruebas

Las simulaciones se llevaron a cabo en un entorno controlado para garantizar la repetibilidad de los resultados. Se detallan los aspectos relevantes:

Simulador	Gazebo Garden 7.9.0
Sistema operativo	Ubuntu 22.04 LTS con WSL (Windows Subsystem for Linux)
Framework	ROS 2 Humble
Controlador de vuelo	PX4 Autopilot
Modelo de dron	x500

Tabla 4.1: Características del entorno utilizado para las simulaciones.

Características de Hardware

CPU	AMD Ryzen 5 5600G - 3.90 GHz
RAM	DDR4 - 16 GB
GPU	NVIDIA GeForce RTX 3060 - 12 GB

Tabla 4.2: Características de hardware utilizado para las simulaciones.

4.2. Escenarios y métricas

A continuación se describen los detalles relativos al diseño de los experimentos.

4.2.1. Escenario base

El escenario principal comienza con la bandada ubicada en el centro de la zona de protección, la figura 4.1 contiene una representación de este escenario. En principio se consideró simular la llegada de la bandada desde fuera de los cultivos e incluir este proceso en las evaluaciones, sin embargo esto causaba escenarios triviales en los cuales las aves se mantenían en las afueras de los cultivos (dado que siempre toman los árboles más cercanos como objetivo) y la expulsión se realizaba prácticamente de manera asegurada. Si bien el sistema está diseñado para comenzar el vuelo cuando se detecta la llegada de un ave dentro de la zona, ubicar la bandada en el centro permite evaluar la eficacia del pastoreo bajo condiciones menos favorables y, por tanto, resultados más confiables.

Por otro lado, los árboles se distribuyen de la siguiente manera:

- Escenario 0: 800 árboles, 40 filas con separación $2,5m$ y 20 columnas con separación $5,0m$. Extensión: 1 hectárea. Utilizado para el ajuste paramétrico.
- Escenario 1: 3872 árboles, 88 filas con separación $2,5m$ y 44 columnas con separación $5,0m$. Extensión: 5 hectáreas.
- Escenario 2: 12100 árboles, 220 filas con separación $1,0m$ y 55 columnas con separación $4,0m$. Extensión: 5 hectáreas.

El esquema de distribución de los árboles en la plantación se basa en los datos reportados en (Roel et al., 2013). La extensión del área de los cultivos se basa en (Grosskoff et al., 2003).

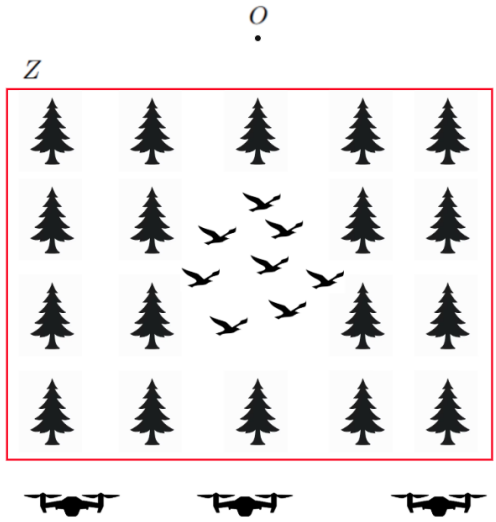

Figura 4.1: Diagrama del escenario base. El recuadro rojo representa Z, la zona de protección.

Finalmente, se posicionan los drones por fuera de la zona de protección, sobre el extremo opuesto al punto objetivo y con una separación de $4{,}0m$ entre ellos. El punto objetivo se define como un punto fijo en cada escenario que se ubica por fuera de la zona de protección, opuesto a la posición inicial de la flota.

4.2.2. Condiciones de parada

Un experimento concluye cuando se cumple alguna de las siguientes condiciones:

- Éxito: Se detecta que no hay aves dentro de Z.
- Fallo: Se alcanza el límite de tiempo definido ($100s$ para el ajuste paramétrico y $180s$ para la evaluación final) sin lograr el objetivo.

4.2.3. Métricas de evaluación

Para cada configuración de parámetros se realizan múltiples ejecuciones para obtener el promedio y la desviación estándar de las siguientes métricas:

- Métrica E: Tasa de éxito, expresada en cantidad de ejecuciones exitosas sobre la cantidad total de ejecuciones. Da una medida de la eficacia del sistema.
- Métrica F: Tiempo final de simulación, expresado en segundos. Se toma en cuenta el tiempo de todas las ejecuciones independientemente de si fueron exitosas o no. Da una medida de la eficiencia del sistema.

- Métrica D: Distancia promedio recorrida por los drones, expresada en metros. Esta métrica es particularmente útil para considerar las capacidades de vuelo y tiempo autónomo que deberían tener los drones a la hora de implantar el sistema.
- Métrica B: Área bajo la curva de la dispersión de la bandada a lo largo del tiempo (Bounding), expresada en metros cuadrados por segundo. Con esta métrica se cuantifica la evolución de la dispersión de la bandada durante el experimento. Esto permite evaluar si la bandada se mantuvo compacta y qué tan eficiente fue el sistema en mantener la cohesión y reducir el área ocupada por las aves a lo largo de toda la trayectoria de pastoreo. Un valor menor en esta métrica indica un control más firme sobre la estructura de la bandada.

En las tablas presentadas en este capítulo se utiliza $\downarrow$ para marcar cuando los valores menores de una métrica indican mejores resultados. En caso contrario se utiliza $\uparrow$.

4.3. Parámetros utilizados

A continuación se detallan los valores de aquellos parámetros del sistema que permanecen fijos durante el proceso de experimentación.

4.3.1. Parámetros del controlador de dron

Parámetro	Descripción	Valor
z_{segura}	Altitud segura de vuelo	6,5m
d_{segura}	Distancia segura de vuelo	4,0m
d_{min}	Distancia mínima entre puntos de pastoreo	4,8m
u	Umbral de parada para cálculo de velocidad	0,1m
f	Factor de distancia para cálculo de velocidad	0,6m
v	Velocidad de recorrida	20,0m/s
$kmeans_u$	Umbral K-means	0,1
$kmeans_i$	Iteraciones K-means	100
$dbscan_n$	Vecinos DBSCAN	1

Tabla 4.3: Parámetros del controlador de dron.

Cabe aclarar que se decidió dejar los parámetros de K-means fijos ya que no se constató una diferencia significativa en el desempeño del sistema al variar sus valores. Por otro lado, el parámetro $dbscan_n$ se deja fijo para asegurar que todos los individuos de la bandada pertenezcan a un cluster.

4.3.2. Parámetros del modelo de bandada

Parámetro	Descripción	Valor
v_{max}	Velocidad lineal máxima	$11{,}25 m/s$
$\omega_{\max}$	Velocidad angular máxima	$3{,}75 rad/s$
$r_{\text{percepcion}}$	Radio de percepción	$10{,}0m$
r_{evasion}	Radio de evasión	$32{,}0m$
w_{ali}	Peso de la fuerza de alineamiento	0,3
w_{coh}	Peso de la fuerza de cohesión	1,3
w_{sep}	Peso de la fuerza de separación	1,1
w_{obj}	Peso de la fuerza de atracción al objetivo	0,2
$arbol_z$	Altura de los árboles	$6{,}0m$
$arbol_{max}$	Cantidad máxima de individuos por árbol	20

Tabla 4.4: Parámetros del modelo de bandada.

Como se mencionó previamente, el relevamiento de valores realistas para estos parámetros resultó difícil debido a la escasez de información respecto al comportamiento de la *Myiopsitta monachus*. Para mitigar este problema, se utilizaron valores de estudios sobre especies con características similares como (Schiffner & Srinivasan, 2016) y de otros trabajos en el área como (Rodriguez Frangias, 2025)

Además, se validó el valor del parámetro r_{evasion} con lo reportado en (Brisson-Curadeau et al., 2025). Los valores obtenidos en este estudio representan cotas superiores para el radio mencionado.

4.4. Ajuste de parámetros

El proceso de calibración del sistema se llevó a cabo utilizando una configuración de 3 drones y 50 aves operando en el Escenario 0, se eligió esta configuración ya que se observó que requería de un buen ajuste para lograr un pastoreo efectivo. Para garantizar la validez estadística de los resultados frente a la naturaleza estocástica de la simulación, se realizaron 10 ejecuciones por cada configuración experimental.

El objetivo primordial de esta fase es determinar los valores óptimos de los parámetros críticos del algoritmo de Strömbom: el radio de recolección R_r y el radio de conducción R_c. Además, el ajuste se extiende a las componentes específicas de cada una de las soluciones propuestas:

- Solución 1 (Clustered Strömbom): Se evalúa el desempeño de los distintos métodos de clustering detallados anteriormente.
- Solución 2 (M-Strömbom): Se busca identificar el factor de separación óptimo (f_s) de la formación en línea para distribuir adecuadamente la presión sobre la bandada.

- Solución 3 (Clustered M-Strömbom): Se analiza la eficacia de los algoritmos de clustering con cantidad de clusters dinámica para adaptarse a la morfología cambiante de la bandada.

Se utilizaron las siguientes configuraciones de valores para los parámetros R_r y R_c:

Config.	R_r	R_c
A	15m	28m
B	15m	27m
C	15m	26m
D	15m	25m
E	20m	28m
F	20m	27m
G	20m	26m
H	20m	25m
I	25m	28m
J	25m	27m
K	25m	26m
L	25m	25m

Tabla 4.5: Configuraciones de los parámetros Radio de conducción (R_c) y Radio de recolección (R_r) utilizadas en el ajuste paramétrico.

El rango de estos valores se estimó mediante observaciones del comportamiento del sistema, determinando los valores mínimos y máximos en los cuales se realizaba un pastoreo efectivo. Notar la mayor granularidad de los valores de R_c, como se mencionó anteriormente, el valor de este parámetro influye enormemente en el desempeño de la solución por lo que se estimó un rango más acotado.

4.4.1. Clustered Strömbom

Se desea determinar el algoritmo de clustering que ofrece mejores resultados para esta solución. De los descritos anteriormente, se decidió dejar por fuera Sectores Angulares (en su variante original) ya que tenía el problema de dejar agentes ociosos en algunas situaciones.

Config.	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
A	6/10	87.1	14.97	286.97	89.74	9.23	4.19
B	4/10	84.8	21.44	375.8	113.9	9.88	4.52
C	7/10	78.1	18.29	374.93	120.92	10.26	4.78
D	4/10	83.6	28.03	495.39	184.76	15.27	8.17
E	5/10	95.2	12.68	329.61	110.74	10.66	5.28
F	**8/10**	**74.2**	**19.4**	**276.1**2	**114.55**	**8.78**	**4.13**
G	6/10	81.1	23.78	386.8	143.47	11.91	5.09
H	3/10	94.0	21.31	537.39	152.18	18.99	7.78
I	5/10	95.6	15.45	404.36	89.35	11.78	3.53
J	4/10	95.5	17.64	386.65	127.45	13.88	7.77
K	6/10	78.6	24.41	355.49	149.97	11.62	6.18
L	7/10	74.9	27.4	362.66	162.86	11.37	5.12

Tabla 4.6: Tasa de éxito (E), Tiempo final de simulación (F), Distancia promedio recorrida por los drones (D) y Bounding (B) para Clustered Strömbom, Sectores angulares con asignación fija.

Config.	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
A	4/10	97.4	14.23	307.0	82.88	8.7	2.62
B	10/10	76.4	8.15	228.2	37.08	7.36	3.59
C	5/10	84.2	24.85	333.97	166.29	10.24	4.31
D	9/10	55.9	18.26	244.34	131.21	7.59	5.04
E	5/10	98.2	13.19	302.07	88.17	7.74	2.77
F†	10/10	74.4	15.35	**212.31**	**56.16**	**6.77**	**2.53**
G	10/10	73.6	15.02	285.49	87.31	8.56	2.93
H	9/10	**58.0**	**17.63**	215.34	100.38	7.74	4.72
I	7/10	95.5	11.38	305.5	50.32	9.33	4.98
J	7/10	81.8	19.16	258.97	108.29	8.08	3.96
K	10/10	63.5	9.22	249.85	78.14	8.77	4.44
L	9/10	66.4	19.5	249.77	103.73	8.69	4.61

Tabla 4.7: Tasa de éxito (E), Tiempo final de simulación (F), Distancia promedio recorrida por los drones (D) y Bounding (B) para Clustered Strömbom, K-means. † Mejor configuración global.

Se puede ver una amplia superioridad en los resultados de K-means por lo que se utilizará este método para la evaluación final. Además, la configuración F presenta resultados mayormente satisfactorios para todas las métricas.

4.4.2. M-Strömbom

Se busca determinar el mejor factor de separación de formación f_s para los valores 2,0 y 2,5.

Config.	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
A	9/10	66.7	20.8	346.79	200.79	5.84	2.81
B	9/10	58.4	16.57	345.7	214.34	5.53	2.88
C	9/10	**50.3**	**17.98**	320.57	214.08	5.54	2.78
D	9/10	54.1	19.73	317.4	154.04	4.84	1.34
E	7/10	83.1	19.19	474.25	238.19	6.46	2.78
F	7/10	68.8	25.91	425.37	208.62	6.53	2.41
G	8/10	58.3	20.16	394.8	216.17	6.01	2.11
H	9/10	52.3	19.46	321.06	242.22	**5.05**	**1.28**
I	9/10	74.5	15.51	361.34	231.3	6.16	3.03
J	9/10	53.2	16.06	**279.07**	**163.72**	5.91	3.09
K	8/10	54.9	24.08	284.64	167.39	5.24	1.77
L	9/10	52.6	18.91	324.94	223.15	6.08	2.88

Tabla 4.8: Tasa de éxito (E), Tiempo final de simulación (F), Distancia promedio recorrida por los drones (D) y Bounding (B) para M-Strömbom, $f_s = 2{,}0$.

Config.	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
A	8/10	77.7	16.76	346.9	172.84	5.11	3.12
B	9/10	68.1	13.19	291.84	131.82	4.64	2.59
C	8/10	55.4	22.18	291.01	175.14	4.83	2.1
D	9/10	61.9	19.61	379.92	166.77	8.49	5.44
E	5/10	82.1	15.04	403.75	183.49	6.78	3.33
F	8/10	68.0	18.82	303.67	176.98	5.95	2.81
G	7/10	61.3	26.78	315.49	191.64	7.54	4.36
H†	**10/10**	**44.2**	**8.9**	**229.32**	**60.56**	**4.58**	**1.11**
I	5/10	83.9	13.82	384.59	159.42	8.17	4.22
J	6/10	76.1	18.17	399.53	183.1	8.57	4.12
K	8/10	66.1	16.98	343.28	115.0	5.84	2.28
L	5/10	71.6	24.26	395.16	178.14	9.93	4.73

Tabla 4.9: Tasa de éxito (E), Tiempo final de simulación (F), Distancia promedio recorrida por los drones (D) y Bounding (B) para M-Strömbom, $f_s = 2{,}5$. † Mejor configuración global.

Se puede ver que los resultados no varían demasiado entre los dos casos. Sin embargo, para $f_s = 2{,}5$ se cuenta con una ligera ventaja en la configuración H.

4.4.3. Clustered M-Strömbom

Nuevamente, se desea determinar el algoritmo de clustering que ofrece mejores resultados. Además, se busca determinar el valor óptimo de $dbscan_r$, el radio dentro del cual dos aves se consideran vecinas, para los valores $10{,}0m, 15{,}0m$ y $20{,}0m$.

Config.	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
A	9/10	69.8	14.94	285.26	61.75	5.98	2.6
B	9/10	70.1	11.88	317.43	86.54	7.05	2.94
C	9/10	54.7	14.94	287.24	127.89	8.44	5.14
D	10/10	47.7	16.99	265.89	117.38	6.6	2.02
E	9/10	69.6	14.88	287.12	77.76	6.75	3.54
F	6/10	66.8	19.48	332.09	139.56	8.45	5.07
G	9/10	51.3	14.8	248.92	98.04	5.89	1.94
H	10/10	**44.1**	**6.44**	**196.65**	**32.88**	**5.28**	**1.53**
I	6/10	70.3	20.45	262.46	88.42	5.39	2.4
J	8/10	66.6	15.75	273.28	79.1	8.94	3.71
K	9/10	60.1	14.71	278.45	98.3	8.68	4.87
L	9/10	47.4	17.51	221.55	129.66	5.64	2.6

Tabla 4.10: Tasa de éxito (E), Tiempo final de simulación (F), Distancia promedio recorrida por los drones (D) y Bounding (B) para Clustered M-Strömbom, X-means.

Config.	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
A	6/10	92.1	13.16	315.73	66.56	7.16	4.53
B	7/10	77.6	19.83	308.66	116.9	6.29	3.76
C	10/10	61.4	6.93	233.69	47.46	5.5	0.99
D	9/10	59.8	18.64	248.16	119.68	8.67	4.04
E	9/10	90.6	10.63	311.68	87.89	5.45	2.25
F	10/10	66.4	8.23	238.9	41.59	5.11	2.26
G	9/10	62.1	16.92	297.09	131.27	6.98	6.04
H†	10/10	**51.0**	**6.64**	**193.2**	**23.78**	**4.92**	**0.83**
I	8/10	91.6	10.47	296.09	74.73	5.5	2.57
J	9/10	75.8	15.98	272.75	74.5	7.16	4.01
K	10/10	64.2	13.64	253.15	71.61	7.23	3.06
L	10/10	52.2	6.73	225.41	51.52	5.99	1.82

Tabla 4.11: Tasa de éxito (E), Tiempo final de simulación (F), Distancia promedio recorrida por los drones (D) y Bounding (B) para Clustered M-Strömbom, DBSCAN $dbscan_r = 10{,}0$. † Mejor configuración global.

Config.	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
A	5/10	97.6	9.95	391.43	92.26	10.12	3.43
B	10/10	73.3	9.69	277.11	48.79	6.65	2.81
C	10/10	71.1	15.33	342.24	123.82	8.92	5.69
D	9/10	58.3	17.1	273.37	130.12	7.69	4.72
E	8/10	92.1	12.05	316.67	63.69	**5.06**	**1.87**
F	9/10	75.4	15.23	295.52	82.91	7.37	4.75
G	8/10	66.4	20.66	287.55	143.46	7.89	6.03
H	10/10	**49.1**	**6.79**	**199.25**	**40.48**	5.77	2.3
I	5/10	92.5	17.75	328.01	104.53	8.67	4.52
J	6/10	88.2	16.68	337.61	124.22	9.55	5.39
K	10/10	64.2	9.94	256.41	60.2	6.68	3.64
L	10/10	56.5	13.43	258.66	110.16	7.14	4.5

Tabla 4.12: Tasa de éxito (E), Tiempo final de simulación (F), Distancia promedio recorrida por los drones (D) y Bounding (B) para Clustered M-Strömbom, DBSCAN $dbscan_r = 15{,}0$.

Config.	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
A	10/10	78.1	12.63	258.02	62.8	4.46	1.33
B	7/10	74.1	20.85	348.31	169.02	8.04	3.93
C	9/10	50.2	17.67	241.92	130.84	5.71	4.92
D	9/10	**50.1**	**16.27**	243.21	117.18	5.77	3.77
E	9/10	82.5	10.76	284.11	82.98	**5.28**	**4.42**
F	9/10	72.9	12.86	277.45	83.53	6.84	3.02
G	8/10	67.0	18.08	281.74	111.49	8.85	4.97
H	10/10	55.1	12.46	271.57	80.95	7.28	3.43
I	4/10	91.1	11.31	336.19	51.1	6.41	2.46
J	10/10	76.5	14.04	317.59	78.51	5.7	2.17
K	9/10	60.6	14.9	255.75	117.78	6.35	3.79
L	9/10	51.4	16.07	**230.84**	**115.03**	6.47	5.66

Tabla 4.13: Tasa de éxito (E), Tiempo final de simulación (F), Distancia promedio recorrida por los drones (D) y Bounding (B) para Clustered M-Strömbom, DBSCAN $dbscan_r = 20{,}0$.

En principio no existe una superioridad clara entre las alternativas de clustering. Sin embargo, al analizar las asignaciones generadas por cada alternativa se observó una tendencia persistente del algoritmo X-means a converger hacia la cantidad máxima de clusters ($K_{max} = 3$).

Esta convergencia no surge de una deficiencia en el algoritmo, sino que parte de la naturaleza del problema. Al observar los datos de las simulaciones, se identifican dos motivos principales que explican este comportamiento:

- Dispersión de la bandada: Al tratarse de aves (agentes con alta movilidad y reacciones de huida), la bandada no siempre se comporta como un solo bloque compacto. Para X-means, si existen dos o tres individuos alejados del grupo principal es razón suficiente para crear un clúster nuevo si eso reduce significativamente la varianza local.
- Limitación de la Flota: Como solamente se cuenta con 3 drones, el "techo" operativo es muy bajo. Es natural que un problema complejo de pastoreo demande siempre el máximo de recursos (drones) disponibles.

Por esta razón, se decidió elegir DBSCAN para la evaluación final. Además, en el mejor caso, presenta valores superiores y mayor consistencia en las métricas B y D.

Por otro lado, se observan mejores resultados para valores menores de $dbscan_r$, esto quiere decir que la solución se comporta mejor para agrupamientos de subbandadas con menor extensión.

4.4.4. Resultados del ajuste paramétrico

Finalmente, se eligieron las siguientes configuraciones para cada solución en base a los resultados de 10 ejecuciones por configuración:

Solución	R_r	R_c	Método de clustering	f_s	dbs_r
1	20,0m	27,0m	K-means	N/A	N/A
2	20,0m	25,0m	N/A	2,5	N/A
3	20,0m	25,0m	DBSCAN	N/A	10,0m

Tabla 4.14: Resultados del ajuste paramétrico.

4.5. Evaluación de las soluciones

Para evaluar la efectividad y robustez de las soluciones finales, se realizan variaciones sistemáticas de los siguientes parámetros:

- Cantidad de drones: Se evalúa la escalabilidad del sistema con flotas de distintos tamaños. Se probaron flotas de 1, 2, 3, 4 y 5 drones
- Cantidad de pájaros: Pruebas con bandadas de distinto tamaño para verificar la efectividad en números realistas a la vez que desafiantes. Se probaron bandadas de 20, 40, 60 y 80 individuos.
- Densidad de árboles: Estudio del comportamiento del sistema en escenarios de cultivos con mayor y menor densidad de obstáculos. Se utilizan los escenarios 1 y 2 detallados anteriormente.

El rango de valores del tamaño de la bandada se obtuvo de estudios realizados sobre la *Myiopsitta monachus* como (Hobson et al., 2014; Tinajero & Estrella, 2015). Para cada combinación de tamaño de flota, tamaño de bandada, escenario y solución se realizan nuevamente 10 ejecuciones.

4.5.1. Solución para un agente

Para comenzar, se evalúa el sistema utilizando un solo agente. Notar que en este caso el comportamiento de las 3 soluciones coincide con el algoritmo original de Strömbom. Para las soluciones 1 y 3, se calcula un solo cluster (que coincide con la bandada completa) y se le aplica el algoritmo sin modificaciones. Para la solución 2, en el estado de recolección se persigue a las m aves más lejanas (en este caso $m = 1$ al igual que Strömbom), para el estado de conducción la formación de la flota se compone por un único punto que coincide con la posición del estado de conducción del algoritmo original.

Escenario	Tamaño bandada	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
			μ	σ	μ	σ	μ	σ
Escenario 1	20	9/10	84.8	37.27	264.51	49.62	3.21	0.31
	40	4/10	149.1	55.43	775.43	845.74	5.23	3.49
	60	5/10	122.2	51.59	521.65	448.0	3.6	1.91
	80	3/10	149.8	54.58	711.01	773.21	3.96	1.75
Escenario 2	20	7/10	97.9	31.83	497.88	309.11	5.21	4.33
	40	5/10	86.0	14.28	709.39	467.03	6.82	4.89
	60	4/10	95.6	16.97	607.04	379.5	4.6	2.15
	80	5/10	91.8	7.78	448.35	260.28	3.65	0.96

Tabla 4.15: Tasa de éxito (E), Tiempo final de simulación (F), Distancia promedio recorrida por los drones (D) y Bounding (B) para Strömbom original.

4.5.2. Clustered Strömbom

Se presentan los resultados de la solución Clustered Strömbom.

Tasa de éxito - cantidad de ejecuciones exitosas sobre cantidad total de ejecuciones

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	9/10	9/10	9/10	6/10
	3	10/10	10/10	7/10	5/10
	4	10/10	10/10	9/10	6/10
	5	10/10	10/10	10/10	10/10
Escenario 2	2	9/10	5/10	5/10	6/10
	3	9/10	10/10	8/10	7/10
	4	10/10	10/10	10/10	6/10
	5	10/10	10/10	9/10	10/10

Tabla 4.16: $E \uparrow$ Clustered Strömbom.

Tiempo final de simulación - (μ,σ) segundos

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	(135.7,35.79)	(131.3,30.55)	(120.2,32.5)	(119.3,50.31)
	3	(98.2,11.23)	(102.0,14.73)	(127.6,41.92)	(141.6,51.51)
	4	(104.9,21.58)	(117.3,24.19)	(99.2,36.25)	(91.3,33.81)
	5	(88.6,14.46)	(106.5,25.57)	(86.7,13.07)	(82.6,8.73)
Escenario 2	2	(89.0,32.93)	(129.7,47.37)	(151.3,39.39)	(152.3,35.01)
	3	(127.1,34.85)	(123.3,15.66)	(131.8,33.89)	(150.0,32.27)
	4	(111.2,26.31)	(119.1,14.09)	(127.2,15.39)	(160.1,25.3)
	5	(102.9,15.22)	(107.2,12.7)	(134.2,27.07)	(124.8,20.59)

Tabla 4.17: $F \downarrow$ Clustered Strömbom.

Distancia recorrida - (μ,σ) metros

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	(607.96,170.19)	(773.58,332.69)	(592.18,138.31)	(517.55,326.6)
	3	(491.89,60.29)	(532.29,169.19)	(611.66,268.39)	(681.48,458.97)
	4	(531.52,167.45)	(484.26,126.92)	(429.74,307.84)	(548.73,399.57)
	5	(372.9,44.01)	(405.07,68.31)	(364.75,57.47)	(267.68,17.62)
Escenario 2	2	(393.35,264.72)	(311.18,25.62)	(667.59,275.18)	(644.36,215.23)
	3	(579.29,145.88)	(394.29,58.93)	(512.85,131.68)	(570.97,174.1)
	4	(494.97,113.73)	(383.06,101.79)	(507.7,66.71)	(493.85,115.04)
	5	(491.57,125.43)	(372.33,86.39)	(464.32,166.48)	(390.93,72.74)

Tabla 4.18: $D \downarrow$ Clustered Strömbom.

Bounding - (μ,σ) metros cuadrados por segundo

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	(4.96,3.05)	(5.63,1.71)	(3.51,0.18)	(8.18,6.19)
	3	(3.76,1.58)	(6.07,2.36)	(7.58,4.54)	(13.21,10.35)
	4	(6.83,4.41)	(5.52,1.88)	(3.17,0.64)	(10.32,10.86)
	5	(4.01,2.11)	(4.17,1.43)	(3.25,0.05)	(3.37,0.3)
Escenario 2	2	(3.99,1.11)	(2.62,0.88)	(6.84,3.61)	(5.69,2.74)
	3	(6.79,4.54)	(3.59,0.68)	(4.72,1.76)	(5.66,4.39)
	4	(3.31,0.11)	(3.24,0.1)	(4.16,0.8)	(6.33,4.0)
	5	(3.56,1.07)	(3.17,0.13)	(4.35,2.13)	(3.81,0.72)

Tabla 4.19: $B \downarrow$ Clustered Strömbom.

4.5.3. M-Strömbom

Se presentan los resultados de la solución M-Strömbom.

Tasa de éxito - cantidad de ejecuciones exitosas sobre cantidad total de ejecuciones

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	9/10	9/10	10/10	9/10
	3	9/10	10/10	10/10	10/10
	4	10/10	9/10	10/10	10/10
	5	10/10	10/10	10/10	10/10
Escenario 2	2	8/10	9/10	8/10	6/10
	3	10/10	10/10	10/10	9/10
	4	10/10	9/10	9/10	8/10
	5	10/10	10/10	9/10	9/10

Tabla 4.20: $E \uparrow$ M-Strömbom.

Tiempo final de simulación - (μ,σ) segundos

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	(128.1,31.15)	(130.5,30.45)	(101.7,25.37)	(112.1,37.4)
	3	(117.8,38.45)	(88.5,18.14)	(83.8,15.86)	(90.7,24.34)
	4	(108.9,31.78)	(96.6,33.7)	(75.1,11.03)	(94.8,9.12)
	5	(84.2,14.24)	(72.2,12.76)	(86.6,8.55)	(111.9,17.54)
Escenario 2	2	(126.4,38.95)	(124.3,35.27)	(117.2,41.85)	(138.7,44.07)
	3	(118.1,35.11)	(95.1,18.15)	(86.5,14.69)	(117.2,31.57)
	4	(113.5,36.44)	(111.7,35.9)	(117.6,32.19)	(117.5,46.72)
	5	(100.1,21.28)	(132.7,32.03)	(128.9,20.92)	(138.0,17.94)

Tabla 4.21: $F \downarrow$ M-Strömbom.

Distancia recorrida - (μ,σ) metros

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	(903.96,437.93)	(989.07,516.68)	(600.26,168.27)	(792.47,456.81)
	3	(781.76,309.36)	(595.85,182.23)	(513.82,133.35)	(499.72,252.38)
	4	(706.24,303.78)	(551.47,301.75)	(417.77,75.32)	(430.75,47.1)
	5	(530.96,93.7)	(486.96,59.42)	(406.77,49.86)	(551.6,109.73)
Escenario 2	2	(607.68,323.82)	(820.7,413.41)	(648.95,181.25)	(1054.78,521.33)
	3	(689.13,277.49)	(553.61,109.07)	(472.82,115.79)	(679.2,336.65)
	4	(629.94,311.47)	(555.04,231.7)	(707.39,239.55)	(575.45,262.79)
	5	(465.28,130.98)	(593.83,191.14)	(460.4,117.05)	(436.75,94.9)

Tabla 4.22: $D \downarrow$ M-Strömbom.

Bounding - (μ,σ) metros cuadrados por segundo

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	(6.65,3.39)	(7.48,4.2)	(5.12,1.28)	(4.82,0.91)
	3	(7.86,2.87)	(5.19,1.46)	(5.0,0.9)	(5.09,1.86)
	4	(8.03,4.42)	(5.74,2.05)	(4.13,0.8)	(4.42,0.78)
	5	(4.93,1.78)	(4.32,0.77)	(4.03,0.31)	(5.14,1.18)
Escenario 2	2	(6.61,5.26)	(5.5,1.63)	(5.34,1.13)	(6.29,2.35)
	3	(6.9,2.44)	(5.15,1.11)	(4.54,0.49)	(5.7,2.2)
	4	(6.61,3.3)	(6.17,2.45)	(6.16,1.63)	(5.65,2.36)
	5	(4.31,0.91)	(4.83,1.6)	(3.65,0.6)	(3.33,0.44)

Tabla 4.23: $B \downarrow$ M-Strömbom.

4.5.4. Clustered M-Strömbom

Se presentan los resultados de la solución Clustered M-Strömbom.

Tasa de éxito - cantidad de ejecuciones exitosas sobre cantidad total de ejecuciones

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	10/10	10/10	8/10	9/10
	3	10/10	10/10	10/10	10/10
	4	10/10	10/10	10/10	10/10
	5	10/10	9/10	10/10	10/10
Escenario 2	2	10/10	10/10	9/10	9/10
	3	10/10	9/10	8/10	10/10
	4	10/10	10/10	10/10	10/10
	5	10/10	9/10	10/10	10/10

Tabla 4.24: $E \uparrow$ Clustered M-Strömbom.

Tiempo final de simulación - (μ,σ) segundos

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	(79.5,19.94)	(78.1,26.37)	(88.3,47.52)	(76.5,40.32)
	3	(71.6,16.22)	(62.0,9.63)	(55.4,9.46)	(89.0,44.65)
	4	(61.0,10.83)	(52.8,5.46)	(56.3,5.66)	(82.1,25.24)
	5	(63.0,9.38)	(53.1,6.67)	(50.6,5.08)	(74.7,6.32)
Escenario 2	2	(55.2,10.82)	(62.6,17.97)	(82.9,32.49)	(96.1,28.9)
	3	(57.0,6.5)	(84.7,37.99)	(83.9,41.62)	(82.1,14.01)
	4	(72.2,12.73)	(72.9,12.8)	(86.7,12.79)	(90.7,15.13)
	5	(92.5,29.31)	(75.9,9.65)	(83.2,8.5)	(93.9,14.94)

Tabla 4.25: $F \downarrow$ Clustered M-Strömbom.

Distancia recorrida - (μ,σ) metros

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	(459.52,100.84)	(454.8,191.91)	(384.98,113.47)	(361.79,179.67)
	3	(443.91,81.35)	(367.52,55.75)	(313.46,61.04)	(499.39,291.05)
	4	(353.42,51.23)	(302.21,31.1)	(302.06,39.91)	(418.15,194.49)
	5	(369.71,46.36)	(348.57,23.44)	(316.09,24.89)	(314.54,38.52)
Escenario 2	2	(339.45,62.99)	(393.38,138.7)	(402.06,77.12)	(387.1,100.29)
	3	(334.39,46.08)	(413.21,127.13)	(340.17,67.11)	(357.65,51.95)
	4	(321.32,32.16)	(350.81,54.78)	(390.54,50.71)	(395.68,58.08)
	5	(356.07,35.89)	(345.73,34.69)	(323.03,78.98)	(359.31,33.09)

Tabla 4.26: $D \downarrow$ Clustered M-Strömbom.

Bounding - (μ,σ) metros cuadrados por segundo

		Tamaño bandada			
Escenario	Tamaño flota	20	40	60	80
Escenario 1	2	(2.64,0.18)	(3.4,1.52)	(3.04,0.68)	(3.36,2.29)
	3	(3.97,2.83)	(2.95,0.67)	(2.62,0.53)	(4.21,2.69)
	4	(2.73,0.34)	(2.42,0.66)	(2.31,0.29)	(4.37,3.95)
	5	(3.09,1.2)	(3.91,2.62)	(2.56,0.95)	(2.7,1.1)
Escenario 2	2	(3.03,2.21)	(2.23,0.71)	(2.19,0.38)	(2.27,0.35)
	3	(2.09,0.25)	(3.46,2.32)	(2.36,0.48)	(2.27,0.26)
	4	(2.41,0.32)	(3.28,3.41)	(2.12,0.3)	(2.18,0.24)
	5	(2.6,0.41)	(2.65,0.83)	(2.4,0.24)	(2.29,0.21)

Tabla 4.27: $B \downarrow$ Clustered M-Strömbom.

4.5.5. Comentarios generales

Se detallan algunas observaciones que aplican a todas las soluciones presentadas. En primer lugar, se observan patrones esperables como un mejor rendimiento del sistema a medida que incrementa el tamaño de la flota y disminuye el tamaño de la bandada.

Además, el sistema se desempeña mejor en el primer escenario, donde la densidad de árboles es menor. Una explicación de este comportamiento es que, a mayor densidad, las aves deben trasladarse distancias menores para llegar a su próximo objetivo, por lo que el desplazamiento de la bandada se torna más lento. Recordar que el criterio de falla establece un tiempo máximo de simulación por lo que si el pastoreo demora demasiado, se considera el intento como fallido.

En general se considera que el desempeño de las soluciones 2 y 3 es satisfactorio en cuanto a su eficacia, logrando consistentemente la expulsión de bandadas de distintos tamaños, en un tiempo acotado y con una flota de 3 a 4 drones para el escenario 1. El escenario 2 requiere de al menos 4 drones para lograr una consistencia similar por lo que se debería tener en cuenta la densidad de los cultivos a la hora de implantar estas soluciones en un entorno real.

La solución 1 requiere de 5 drones para adaptarse a bandadas de gran tamaño, por lo cual no es la mejor alternativa para lidiar con este tipo de escenarios. Debería estudiarse qué tan comunes son las bandadas de este tamaño en los lugares que se desee implantar el sistema antes de considerar esta solución como alternativa.

Por otro lado, analizando la métrica B se puede ver que la solución 3 tiene una gran capacidad de contener la bandada y evitar la dispersión. Las otras soluciones presentan valores ligeramente peores pero no por un gran margen, con lo cual se considera que las 3 alternativas logran un desempeño aceptable en este frente.

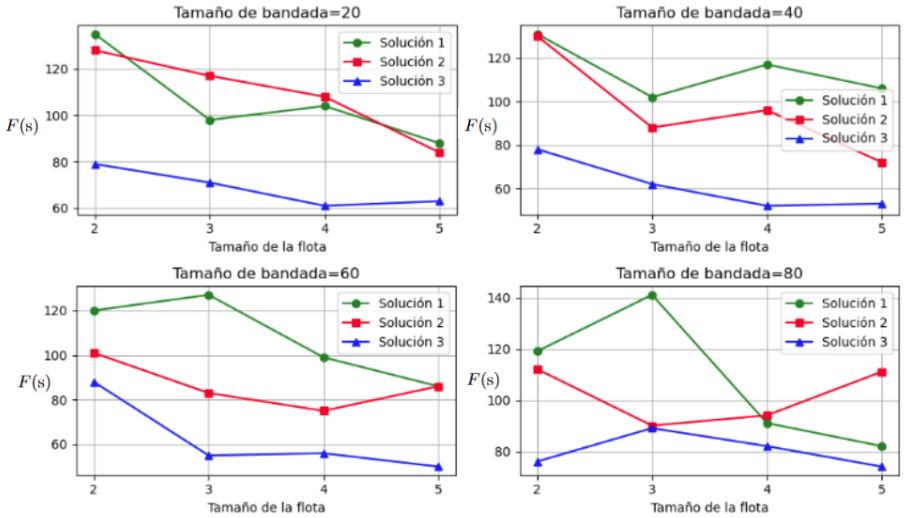

Figura 4.2: μ de F para las distintas soluciones en el escenario 1.

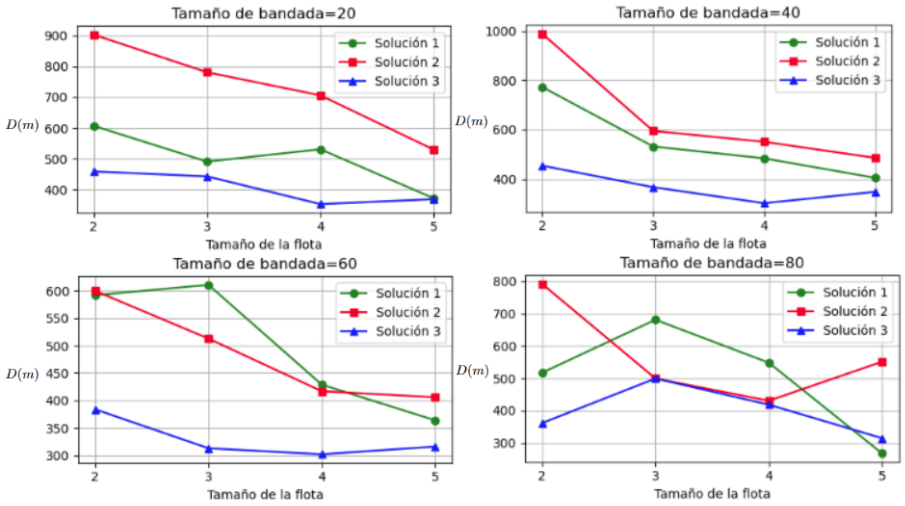

Figura 4.3: μ de D para las distintas soluciones en el escenario 1.

Por último, se considera la eficiencia de las 3 soluciones. En cuanto a la duración del proceso de pastoreo F, se observa una enorme ventaja de la solución 3, requiriendo poco más de un minuto para expulsar a la bandada en la mayoría de los casos. En contraste, las soluciones 1 y 2 requieren por encima del doble de tiempo para lograr la expulsión. De forma similar, la solución 3 presenta mejores distancias de vuelo promedio D que el resto de las soluciones, siendo la solución 2 la que presenta el peor desempeño por un amplio margen. Por estas razones, se puede afirmar que Clustered M-Strömbom es el algoritmo con mayor eficiencia de los tres. Para visualizar lo descrito anteriormente, la figura 4.2 muestra gráficas de la media de F en función del tamaño de la flota para las 3 soluciones en el escenario 1 y la figura 4.3 muestra la misma información para la métrica D.

4.6. Análisis de sensibilidad

Para finalizar la evaluación del sistema, se busca determinar la sensibilidad frente a variaciones en aquellos parámetros para los cuales no se pudo obtener un valor aproximado. Principalmente, esto comprende los parámetros que regulan el comportamiento del modelo de bandada.

Si bien se tomaron valores utilizados en trabajos afines, no se encontraron fuentes que validen estos valores frente a observaciones o medidas de bandadas reales. Por lo tanto, el objetivo de esta sección es evaluar el impacto de variar los valores de estos parámetros dentro de ciertos rangos.

Para esto se utiliza el escenario descrito en el ajuste paramétrico con una flota de 3 drones ejecutando Clustered M-Strömbom. Se elige esta solución ya que presenta mayor consistencia en general, esto permite estimar la variabilidad

introducida por los cambios en los parámetros de forma más precisa. Para cada valor de cada parámetro estudiado se realizan nuevamente 10 ejecuciones.

Se seleccionaron los siguientes parámetros con sus respectivos rangos de valores, se marca en negrita el valor utilizado para el resto de las configuraciones:

w_{ali}	**0,3**	0,5	1,0
w_{coh}	1,0	1,15	**1,3**
w_{sep}	0,4	0,6	**1,1**
w_{obj}	**0,2**	0,5	1,0
$r_{\text{percepcion}}$	6,0m	8,0m	**10,0m**

Tabla 4.28: Valores de los parámetros para el análisis de sensibilidad.

Nuevamente, estos valores fueron elegidos dentro de rangos que preservan los comportamientos propios de aves gregarias. En el caso del peso de la fuerza de cohesión se detectó que, para valores menores a 1,0, el comportamiento grupal desaparecía por completo y cada individuo actuaba por cuenta propia. En algunas investigaciones como (Rodriguez Frangias, 2025) se agrega una componente al modelo de Reynolds conocida como "bounding" que previene esta situación incorporando una fuerza intensa de atracción hacia el centro de masas de la bandada para los individuos que se encuentran en la periferia. Al no contar con esta componente, el modelo desarrollado es más restrictivo en cuanto a los valores de w_{coh}, ya que para valores bajos se obtiene el comportamiento descrito sin posibilidad de controlarlo mediante otro parámetro (como sería el peso de la fuerza de bounding mencionada anteriormente).

Este tipo de consideraciones son particularmente relevantes ya que la especie de ave en la que se enfoca este trabajo, la cotorra, exhibe comportamientos sociales y vive en comunidad (Tinajero & Estrella, 2015). En este sentido, los valores utilizados para los parámetros deben resultar en comportamientos que reflejen este hecho para reducir la brecha entre la simulación y el ambiente de aplicación real del sistema.

4.6.1. Peso de fuerza de alineamiento

Este parámetro controla qué tanto peso le da cada individuo a la dirección en la cual se mueven los otros integrantes de la bandada dentro de su radio de percepción. Intuitivamente, a valores mayores la bandada tiende a ir en una misma dirección conjunta y a valores menores se debería observar mayor disparidad en las direcciones individuales. La figura 4.4 muestra algunos boxplots de las métricas obtenidas para las variaciones de este parámetro.

w_{ali}	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
0,3	10/10	47.3	6.4	**250.71**	**65.07**	5.76	2.3
0,5	10/10	60.1	14.22	288.19	94.99	6.83	2.71
1,0	10/10	**33.1**	**7.33**	262.09	66.46	**3.8**	**2.06**

Tabla 4.29: Sensibilidad del sistema frente a variaciones en w_{ali}.

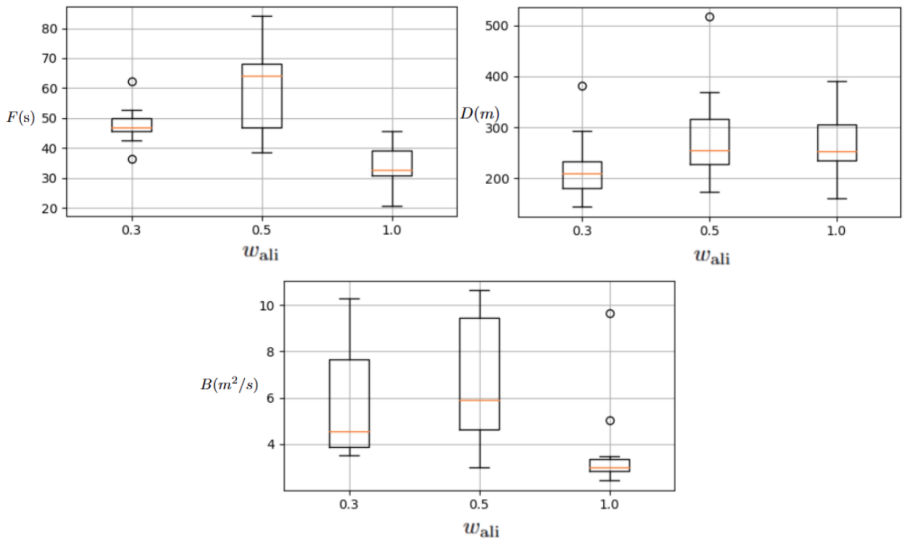

Figura 4.4: Boxplots de $E \uparrow$, F y B del sistema frente a variaciones en w_{ali}.

Si bien no parece existir una gran variación en los resultados se puede ver un impacto positivo en la duración del proceso F para el valor máximo de w_{ali}, implicando casi la mitad de tiempo en promedio, también se observa una mejora para la métrica B en este caso. Resulta razonable que si la tendencia que tienen los individuos de seguir la dirección del resto de la bandada es mayor, más fácil será dirigir y contener al grupo. Aún así, el sistema logra resultados aceptables para valores bajos de este parámetro.

4.6.2. Peso de fuerza de cohesión

Este parámetro controla qué tanto se ve atraído un individuo hacia aquellos integrantes de la bandada dentro de su radio de percepción. A valores mayores la bandada se mantiene más unida y a valores menores más dispersa. La figura 4.5 muestra algunos boxplots de las métricas obtenidas para las variaciones de este parámetro.

w_{coh}	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
1,0	7/10	72.0	17.7	457.24	177.0	10.17	3.77
1,15	10/10	49.5	6.03	264.43	57.81	6.22	1.9
1,3	10/10	**47.3**	**6.4**	**250.71**	**65.07**	**5.76**	**2.3**

Tabla 4.30: Sensibilidad del sistema frente a variaciones en w_{coh}.

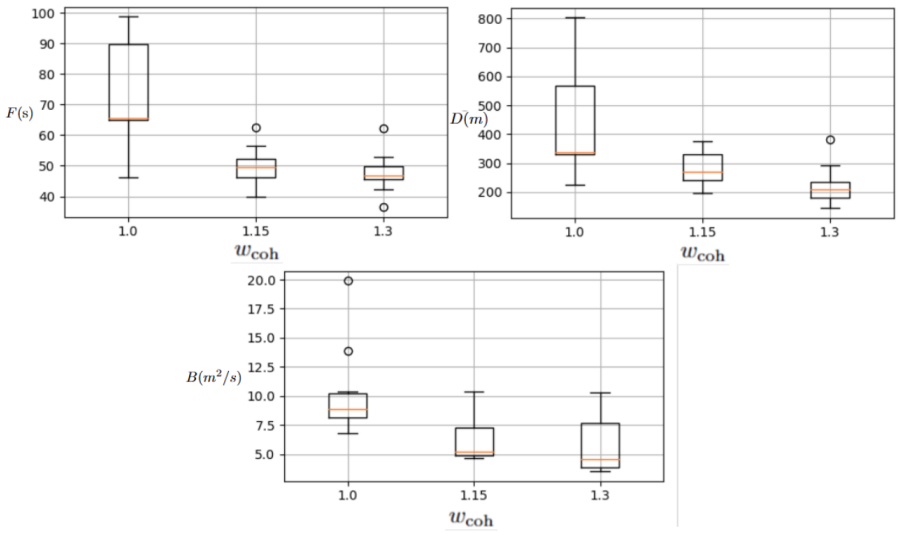

Figura 4.5: Boxplots de E, F y B del sistema frente a variaciones en w_{coh}.

Los resultados indican que el desempeño del sistema empieza a caer a medida que la bandada presenta mayores valores de dispersión. En las simulaciones se puede observar una gran sensibilidad del sistema ante individuos dispersos que quedan por fuera del radio de recolección, una situación muy común en bandadas con baja cohesión.

Una manera de mitigar esta sensibilidad consiste en incrementar R_r de forma que el algoritmo tenga mayor tolerancia ante individuos que se separan ligeramente del grupo principal. De la misma manera, se puede aumentar $dbscan_r$ para que se divida la bandada en agrupaciones más grandes, la siguiente tabla presenta los resultados de ejecutar el sistema con $w_{coh} = 1{,}0$, $R_r = 28{,}0m$ y $dbscan_r = 12{,}0m$.

$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
	μ	σ	μ	σ	μ	σ
9/10	54.8	16.05	348.34	154.21	8.22	1.74

Tabla 4.31: Resultados para $w_{coh} = 1{,}0$, $R_r = 28{,}0m$ y $dbscan_r = 12{,}0m$.

4.6.3. Peso de fuerza de separación

Este parámetro controla qué tanto se separa un individuo de aquellos integrantes de la bandada dentro de su radio de percepción. Variar su valor tiene el efecto contrario a variar el peso de la fuerza de cohesión. La figura 4.6 muestra algunos boxplots de las métricas obtenidas para las variaciones de este parámetro.

w_{sep}	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
0,4	9/10	80.4	9.64	186.22	10.32	**3.21**	**0.05**
0,6	10/10	74.8	9.27	**183.7**	**10.23**	3.23	0.09
1,1	10/10	**47.3**	**6.4**	250.71	65.07	5.76	2.3

Tabla 4.32: Sensibilidad del sistema frente a variaciones en w_{sep}.

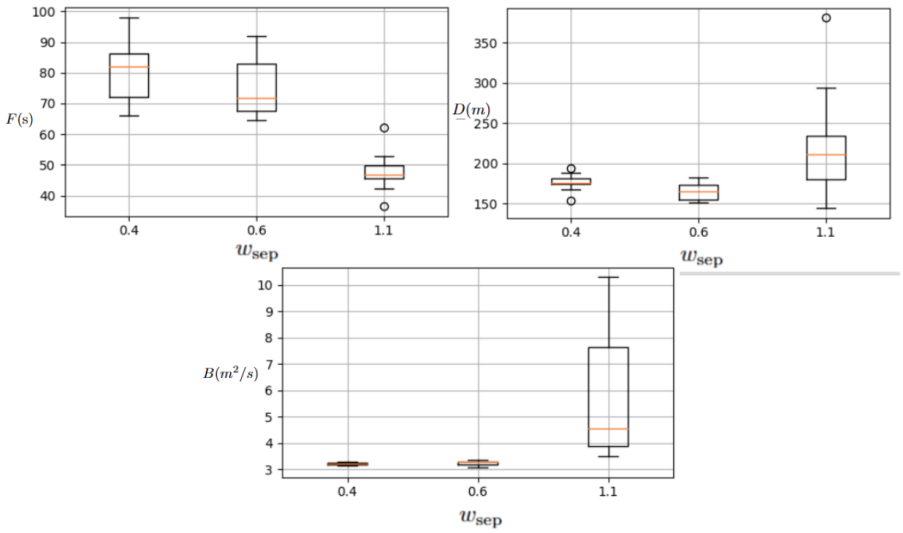

Figura 4.6: Boxplots de E, F y B del sistema frente a variaciones en w_{sep}.

En primer lugar, se pueden notar valores menores de B cuando este parámetro decrementa, lo cual resulta razonable ya que los individuos tienen menor separación entre ellos. Por otro lado, se observa que a medida que disminuye la fuerza de separación, el desempeño del sistema cae ligeramente en cuanto a las métricas E y F. Esto puede parecer contradictorio en principio ya que la bandada se mantiene más compacta en estas situaciones. Sin embargo, en las simulaciones se observa que el desplazamiento de la bandada se enlentece considerablemente debido a que la atracción entre los individuos es más fuerte, esto se comprueba observando los valores de B.

Se puede mitigar este problema disminuyendo el radio de conducción R_c para ejercer una presión mayor sobre la bandada, la siguiente tabla presenta los resultados de ejecutar el sistema con $w_{sep} = 0{,}4$ y $R_c = 20{,}0m$.

$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
	μ	σ	μ	σ	μ	σ
10/10	46.3	5.15	126.21	3.87	2.91	0.17

Tabla 4.33: Resultados para $w_{sep} = 0{,}4$ y $R_c = 20{,}0m$.

4.6.4. Peso de fuerza de atracción a objetivo

Este parámetro controla qué tanto se ve atraído un individuo hacia los árboles de la zona de cultivos. A valores mayores cada individuo buscará con mayor intensidad llegar a los árboles, siempre y cuando no esté siendo perseguido por un dron. La figura 4.7 muestra algunos boxplots de las métricas obtenidas para las variaciones de este parámetro.

w_{obj}	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
0,2	10/10	**47.3**	**6.4**	**250.71**	**65.07**	5.76	2.3
0,5	10/10	52.6	6.52	252.62	59.68	5.58	2.2
1,0	10/10	59.4	14.31	302.96	67.53	**5.17**	**0.71**

Tabla 4.34: Sensibilidad del sistema frente a variaciones en w_{obj}.

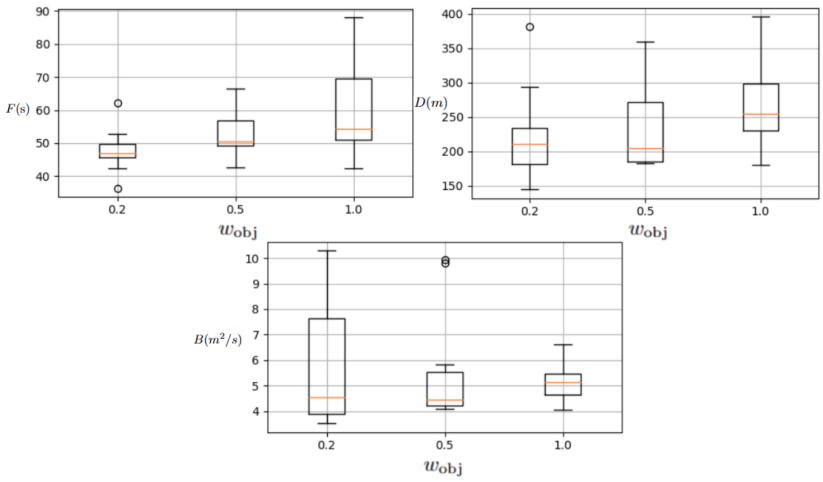

Figura 4.7: Boxplots de E, F y B del sistema frente a variaciones en w_{obj}.

No se observan grandes impactos al variar el peso de atracción hacia el objetivo. El proceso de pastoreo se vuelve ligeramente más lento y requiere de una recorrida mayor por parte de los drones a medida que aumenta esta fuerza. Más allá de esto, el sistema logra un desempeño adecuado para todos los casos.

La explicación de este resultado parte de que el modelo de bandada asume que la fuerza de atracción al objetivo no influye en el movimiento de las aves cuando están siendo perseguidas por los drones. Si la fuerza se mantuviera presente en estos casos el impacto sería mayor ya que contrarrestaría el movimiento de escape.

4.6.5. Radio de percepción

Este parámetro controla a qué distancia un individuo empieza a influir en otros para el cálculo de las fuerzas de cohesión, separación y alineamiento. Intuitivamente, a valores menores se deberían observar divisiones de la bandada en sub-grupos más pequeños. La figura 4.8 muestra algunos boxplots de las métricas obtenidas para las variaciones de este parámetro.

$r_{percepcion}$	$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
		μ	σ	μ	σ	μ	σ
6,0m	8/10	76.8	22.71	541.17	169.1	11.68	3.14
8,0m	8/10	59.9	20.89	363.36	173.28	7.11	4.41
10,0m	**10/10**	**47.3**	**6.4**	**250.71**	**65.07**	**5.76**	**2.3**

Tabla 4.35: Sensibilidad del sistema frente a variaciones en $r_{percepcion}$.

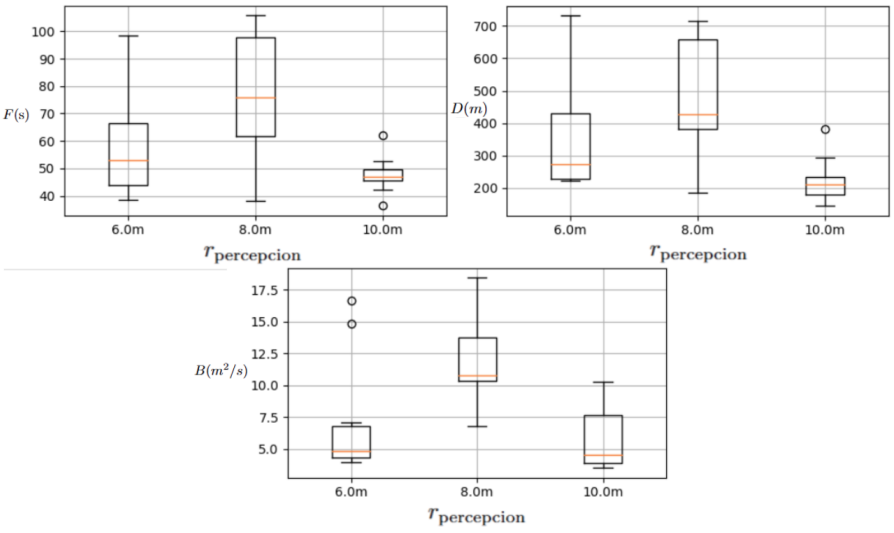

Figura 4.8: Boxplots de E, F y B del sistema frente a variaciones en $r_{percepcion}$.

Se observa que el desempeño del sistema cae para todas las métricas a medida que la bandada tiende a dividirse en grupos más pequeños, lo cual se refleja en una tasa de éxito menor y un peor control de la dispersión de la bandada. Recordar que el ajuste paramétrico de la solución utilizada derivó en un valor de $dbscan_r$ de 10,0m (el menor valor considerado) lo cual parece no ser compatible con esta situación en la cual se forman más sub-bandadas de menor tamaño. Disminuir este radio ayuda a mitigar el problema descrito, la siguiente tabla presenta los resultados de ejecutar el sistema con $r_{percepcion} = 6{,}0m$ y $dbscan_r = 8{,}0m$.

$E \uparrow$	$F \downarrow$		$D \downarrow$		$B \downarrow$	
	μ	σ	μ	σ	μ	σ
9/10	80.2	20.35	534.21	191.19	10.89	3.87

Tabla 4.36: Resultados para $r_{percepcion} = 6{,}0m$ y $dbscan_r = 8{,}0m$.

4.6.6. Resultados del análisis de sensibilidad

El análisis de sensibilidad realizado reveló que el sistema mantiene un desempeño adecuado para la mayoría de los casos. Si bien se producen caídas de las métricas para ciertos escenarios, no se llega a escenarios en los que el sistema falle completamente sino que los resultados muestran menor consistencia.

En este sentido, el análisis reveló una de las principales debilidades de las soluciones presentadas: una fuerte dependencia en el ajuste de ciertos parámetros que impactan fuertemente en el desempeño de los pastores. Como el sistema no cuenta con mecanismos para adaptar estos parámetros ante ciertas distribuciones de bandada, se observa una consistencia menor a la hora de lidiar con grupos cuyo comportamiento difiere del que se utilizó para el ajuste paramétrico.

Sería importante contar con dichos mecanismos si se busca mejorar la robustez del sistema. En particular, diseñar lógicas para estimar los radios R_c, R_r y $dbscan_r$ dinámicamente, a partir del comportamiento de la bandada detectada.

Por otro lado, en base a los resultados de la sección 4.5 se puede afirmar que estos escenarios problemáticos pueden ser mitigados incrementando el tamaño de la flota, ya que esto resulta en un mejor desempeño general de las soluciones. Sin embargo, esto solo se puede lograr a través de una mayor inversión en infraestructura por lo que en principio deberían considerarse alternativas como las presentadas anteriormente. Más allá de esto los resultados presentados en estas secciones permiten estimar los límites funcionales de las soluciones en relación al tamaño de la flota y la bandada.

4.7. Conclusiones preliminares

Finalizada la etapa de experimentación se concluye que las soluciones evaluadas presentan un buen desempeño para los escenarios descritos, con posibilidades de mejora ante variaciones en el comportamiento de las aves.

En particular se destaca la capacidad de las 3 soluciones para evitar la dispersión de la bandada (evaluado con la métrica B), la eficiencia de la solución 3 para completar el proceso de pastoreo y el desempeño de la solución 2 que, a través de una lógica con menor complejidad computacional, presenta resultados comparables con los de sus contrapartes más complejas. También se destaca la mejora de desempeño presentada por las 3 soluciones respecto al algoritmo original para un solo agente.

En general, las 3 soluciones resultaron eficaces para el pastoreo de las bandadas simuladas. La principal diferencia entre dichas soluciones radica en los recursos requeridos (cantidad de agentes y tiempo de cómputo) para lograr un cierto grado de consistencia. También se observan diferencias sustanciales en la cantidad de tiempo F y distancia recorrida por los drones D que se necesita para completar el proceso con cada solución.

Capítulo 5

Conclusiones

El presente trabajo permitió validar la factibilidad técnica de utilizar una flota coordinada de drones para el pastoreo de aves plaga en un entorno de simulación. Se analizaron varias estrategias, contemplando sus fortalezas y debilidades y se evaluó el desempeño de las mismas en diversos escenarios. A partir del trabajo desarrollado a lo largo de este informe, se extraen las siguientes conclusiones:

Evaluación de las estrategias multi-agente

Las extensiones desarrolladas sobre el algoritmo de Strömbom demostraron que la cooperación entre drones permite mejorar la eficacia del pastoreo. Las soluciones presentan una gran escalabilidad y realizan un buen aprovechamiento de los recursos disponibles, minimizando el costo de las tareas asignadas a cada agente mediante mecanismos deterministas que aseguran la coordinación.

Se considera que todas los algoritmos evaluados resultan eficaces pero difieren en sus requerimientos en cantidad de agentes y capacidad de cómputo para asegurar la consistencia de su desempeño. Esto resulta particularmente relevante ya que el costo computacional de las soluciones y el costo en infraestructura de comunicación requerida crecen rápidamente a medida que aumenta el tamaño de la flota. De más está decir que esto también implicaría una mayor inversión económica por parte de los productores. En este sentido, las soluciones 2 y 3 son preferibles dado que logran mejores resultados con una cantidad menor de agentes.

Por otro lado, algunas soluciones presentan una mayor eficiencia en cuanto a la duración del proceso y el esfuerzo que debe realizar cada agente. En particular, la solución *Clustered M-Strömbom* probó ser la más robusta al adaptarse a la fragmentación natural de la bandada, logrando reducir la métrica B y manteniendo la cohesión grupal durante la expulsión.

Robustez del diseño del sistema

La utilización de un esquema basado en coordinación implícita y comunicación mediante ROS 2 permitió crear un sistema distribuido que evita puntos únicos de falla y otros problemas de los sistemas centralizados. Sin embargo, este diseño impone una dependencia crítica sobre la disponibilidad y frescura

de los datos de posicionamiento de la flota y las aves.

En general, uno de los principales desafíos en la etapa de diseño fue la falta de detalles respecto a las características de un sistema de detección de aves. Aún así, se tuvieron consideraciones de diseño e implementación de manera de introducir la menor cantidad de restricciones posibles para el futuro desarrollo de dicho sistema.

Más allá de esto, los algoritmos presentados requieren de ciertas certezas para su correcto funcionamiento en una implementación real. En particular, la red DDS debe garantizar latencias mínimas y consistencia en la información obtenida por cada dron para evitar divergencias en los cálculos realizados.

Complejidad del comportamiento animal

El modelado de la *Myiopsitta monachus* reveló que la efectividad del pastoreo depende significativamente de ciertos parámetros. La escasez de datos biológicos específicos sobre esta especie representó un desafío, cuya incertidumbre se mitigó mediante un estudio de sensibilidad de los parámetros que regulan el comportamiento del modelo desarrollado.

El estudio reveló una de las principales debilidades de las soluciones desarrolladas: el desempeño del sistema depende fuertemente de un adecuado ajuste de ciertos parámetros críticos para la lógica de pastoreo. Al introducir variaciones en el comportamiento de la bandada, los valores ajustados anteriormente dejan de ser óptimos lo cual provoca una caída en el desempeño del sistema.

Se considera importante que las siguientes etapas de este proyecto se centren en el estudio del comportamiento de esta especie con el fin de reducir la brecha entre el modelo computacional utilizado y el comportamiento de las aves en el entorno real.

De esta forma, el trabajo presentado se incorpora al área de investigación de técnicas de pastoreo mediante sistemas robóticos, aportando diversas soluciones que podrían ser elegidas para implantarse en cultivos reales. Dicha elección dependerá fuertemente de las restricciones y problemas que se encuentren en el escenario particular.

5.1. Trabajo a Futuro

La presente investigación establece una arquitectura funcional para el pastoreo, pero abre diversas líneas de desarrollo para aumentar la eficiencia operativa y la sostenibilidad del sistema:

5.1.1. Optimización algorítmica

Para comenzar, se recomienda evaluar la posibilidad de realizar un ajuste automático de los radios R_c, R_r y $dbscan_r$. Estos parámetros resultan críticos para la eficacia del pastoreo por lo cual sería interesante encontrar formas de estimarlos en tiempo de vuelo en lugar de depender de un valor estático. Una alternativa para el ajuste de R_c podría ser estimarlo en función de la velocidad de la bandada, detectando el momento en el cual comienza la huida y ajustando

R_c a la distancia que se encuentren la flota del centro de masas de la bandada en dicho momento. Se deberían tener cuidados para asegurar que todos los drones calculen la misma distancia si se busca mantener la coordinación implícita.

Complementariamente, se propone el uso de maniobras de aproximación por flancos para mejorar la directividad hacia el objetivo de pastoreo. De esta forma, los drones podrían efectuar el pastoreo hacia distintos puntos objetivos sin importar su posición inicial.

Por otro lado, sería interesante evaluar la capacidad de expulsar diferentes bandadas hacia puntos de salida distintos de forma simultánea, calculando objetivos de expulsión dinámicos en lugar de forzar un agrupamiento único. En este proyecto se trabajó sobre el supuesto de que solo puede existir una bandada o grupo consolidado en la zona de protección en un momento dado. Si este no fuera el caso, se debería introducir un nuevo concepto de división para diferenciar las bandadas y considerar múltiples puntos objetivo.

De forma similar, puede ser deseable contar con la posibilidad de definir ciertas áreas adyacentes a Z que contengan posiciones válidas para un punto objetivo. En las soluciones actuales se elige este punto arbitrariamente por fuera de Z pero si se quisiera calcular dinámicamente es necesario poder restringir las zonas hacia las cuales se expulsa la bandada. Esto puede ser útil para evitar dirigir a la bandada hacia un cultivo o propiedad vecina, lo cual simplemente traslada el problema de un lugar a otro. De forma inversa, también se podría utilizar para guiar a la bandada hacia su nido (si se conociera la ubicación) o hacia un cultivo mártir (zona reducida de cultivos que se sacrifican en pos de mantener el cultivo principal seguro).

En cuanto a las soluciones particulares, se podrían probar algunas alternativas:

- Evolucionar el algoritmo Clustered M-Strömbom con un sistema de asignación inteligente que considere tanto la dispersión espacial como la priorización dinámica de clusters. Los pesos para la asignación de agentes se basarían en métricas de inestabilidad (dispersión, velocidad relativa, cohesión) y cuando el número de clusters supere la capacidad de la flota, se implementaría un sistema de scheduling que alterne objetivos según criterios de riesgo de dispersión.

- Probar distintas formaciones para el estado de conducción de M-Strömbom. Si bien la formación en línea presentó resultados aceptables, otros estudios como (Shi & Wang, 2022) proponen el uso de formaciones en arco para el pastoreo de aves.

5.1.2. Consideraciones para la implantación del sistema

A la hora de abandonar el entorno de simulaciones y trasladar el sistema a un entorno real se deben tener varios cuidados. Para empezar, se deberían integrar mecanismos más sofisticados de evasión de obstáculos y prevención de colisiones entre agentes, que permitan vuelos en formaciones cerradas sin comprometer la

seguridad de los drones. Si bien se implementaron algunos controles, es importante contar con una lógica robusta que detecte intersección de trayectorias con otros drones u obstáculos del lugar.

Por otro lado, dado el diseño distribuido de las soluciones, se podría desarrollar un sistema de detección de fallas en tiempo de vuelo. Si un dron presenta defectos en su funcionamiento, el sistema podría ser capaz de reconfigurar el tamaño de la flota y redistribuir las tareas de pastoreo dinámicamente entre los agentes operativos restantes. De esta forma se maximiza el tiempo operativo del sistema. De forma similar, se podría estudiar la posibilidad de iniciar el proceso de pastoreo utilizando solamente una parte de la flota total de drones y dejando al resto en reserva. Esto permitiría disminuir el desgaste de los drones en escenarios en los que no sea necesario hacer uso de la flota completa, por ejemplo para bandadas de pocos individuos.

En cuanto a las consideraciones de complejidad computacional detalladas en el capítulo 3, si se detecta que no es posible ejecutar los cálculos a una frecuencia adecuada (ya sea por las capacidades de cómputo del dron o un escalado del tamaño de la flota o de la bandada), se podría considerar implementar la ejecución de la asignación de clusters de manera asíncrona cada n ciclos en lugar de en cada paso temporal. Sin embargo esto introduciría una nueva restricción: el sistema de detección de aves debería asegurar consistencia de datos entre las detecciones en t_i y t_{i+1} para mantener el seguimiento de los individuos. Esto se debe a que la asignación de clusters calculada en un tiempo anterior solo mantiene validez si la lista de posiciones preserva el indexado de los individuos.

Finalmente, es importante evaluar y monitorear el impacto ambiental de un sistema de estas características. Esto abarca investigar el impacto del estrés crónico en la *Myiopsitta monachus* y otras especies. También resulta fundamental diseñar rutas de pastoreo hacia corredores biológicos para mitigar el "efecto vecindario", garantizando que el éxito en un cultivo no signifique simplemente transferir la presencia de la bandada a otras áreas productivas o ecosistemas vulnerables, independientemente de su ubicación geográfica. De forma similar, se debería evaluar la pérdida de eficacia del sistema a largo plazo debido a la inteligencia de las aves y su capacidad de adaptación y proponer estrategias de mitigación de ser necesario.

Referencias

Bajec, I. L., Zimic, N., & Mraz, M. (2007). The computational beauty of flocking: Boids revisited. *Mathematical and Computer Modelling of Dynamical Systems*, *13*(4), 331-347.

Brisson-Curadeau, É., Lacombe, R., Gousy-Leblanc, M., Poirier, V., Jackson, L., Petalas, C., Miranda, E., Eby, A., Baak, J., Léandri-Breton, D.-J., Choy, E., Legros, J., Tranze-Drabinia, E., & Elliott, K. H. (2025). A meta-analysis of the impact of drones on birds. *Frontiers in Ecol Environ*, *23*(2), e2809.

Dronecode. (2026). PX4 Autopilot User Guide [Accedido: 2026-01-10]. `https://docs.px4.io/main/en/`

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 226-231.

Grosskoff, R., Saavedra, C., Hernández, A., Bregante, A., Arenare, L., Casares, I., Méndez, I., Méndez, J. A., Ramos, A. R. C., Rincón, F., & Grasso, A. (2003). *ENCUESTA FRUTICOLA Zafra 2002/03 - Serie Encuestas N°216* (inf. téc.). DIEA.

Hare, T. (1859). *A Treatise on the Election of Representatives, Parliamentary and Municipal*. Longman, Brown, Green, Longmans & Roberts.

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A K-Means Clustering Algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *28*(1), 100-108.

Hasan, Y., Baxter, J. E. G., Salcedo, C. A., & Delgado, E. (2022). Flock Navigation by Coordinated Shepherds via Reinforcement Learning. *International Workshop on the Algorithmic Foundations of Robotics*, 454-469.

Hobson, E. A., Avery, M. L., & Wright, T. F. (2014). The socioecology of Monk Parakeets: Insights into parrot social complexity. *The Auk: Ornithological Advances*, *131*(4), 756-775.

Hoshi, H., Iimura, I., Nakayama, S., Moriyama, Y., & Ishibashi, K. (2018). Computer Simulation Based Robustness Comparison Regarding Agents' Moving-Speeds in Two- and Three-Dimensional Herding Algorithms. *Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*, 1307-1314.

INIA. (2025). Método contraceptivo para el manejo del daño de cotorras [Accedido: 2026-01-10]. https://inia.uy/proyectos/metodo-contraceptivo-para-el-manejo-del-dano-de-cotorras

Jawhar, I., Mohamed, N., Wu, J., & Al-Jaroodi, J. (2018). Networking of Multi-Robot Systems: Architectures and Requirements. *Sensor and Actuator Networks: Feature Papers*, *7*(4), 52.

Jeong-Hun, K., Jong-Hyeok, C., Kwan-Hee, Y., & Aziz, N. (2019). AA-DBSCAN: an approximate adaptive DBSCAN for finding clusters with varying densities. *The Journal of Supercomputing*, *201*(75), 142-169.

King, A. J., Portugal, S. J., Strömbom, D., Mann, R. P., Carrillo, J. A., Kalise, D., de Croon, G., Barnett, H., Scerri, P., Groß, R., Chadwick, D. R., & Papadopoulou, M. (2023). Biologically inspired herding of animal groups by robots. *Methods in Ecology and Evolution*, *14*(2), 478-486.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, *2*(1-2), 83-97.

Lee, W., & Kim, D. (2017). Autonomous Shepherding Behaviors of Multiple Target Steering Robots. *Sensors, Wireless Connectivity and Systems for Autonomous Vehicles and Smart Mobility*, *17*(12), 2729.

Long, N. K., Sammut, K., Sgarioto, D., Garratt, M., & Abbass, H. A. (2020). A Comprehensive Review of Shepherding as a Bio-Inspired Swarm-Robotics Guidance Approach. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *4*(4), 523-537.

MercoPress. (2025). Uruguay aprueba el sacrificio masivo de cotorras para salvar cultivos [Accedido: 2026-01-10]. https://es.mercopress.com/2025/11/29/uruguay-aprueba-el-sacrificio-masivo-de-cotorras-para-salvar-cultivos#:~:text=Estos%20compuestos%20causan%20una%20muerte,Medio%20ambiente%2C%20Pol%C3%ADtica%2C%20Uruguay.

MGAP. (2021). Manejo de plagas - Cotorra [Accedido: 2025-12-10]. https://www.gub.uy/ministerio-ganaderia-agricultura-pesca/politicas-y-gestion/cotorra

MGAP. (2024). Valor Bruto de Producción a precio final del complejo hortifrutícola y su aporte al empleo en Uruguay. *Observatorio Granjero*.

Okoli, C. (2015). A guide to conducting a standalone systematic literature review. *Communications of the association for information systems*, *37*(43).

Olivera, L., & Rodriguez, E. (2018). GUÍA DE BUENAS PRÁCTICAS PARA EL MANEJO DE COTORRAS EN CULTIVOS. *Área Vertebrados Plagas Acuerdo INIA-DGSA*.

Paranjape, A. A., Chung, S.-J., Kim, K., & Shim, D. H. (2018). Robotic Herding of a Flock of Birds Using an Unmanned Aerial Vehicle. *IEEE Transactions on Robotics*, *34*(4), 901-915.

Pelleg, D., & Moore, A. (2000). X-means: Extending K-means with efficient estimation of the number of clusters. *Proceedings of the Seventeenth International Conference on Machine Learning*, 727-734.

Radoglou-Grammatikis, P., Sarigiannidis, P., Lagkas, T., & Moscholios, I. (2020). A compilation of UAV applications for precision agriculture. *Computer Networks*, *172*, 107148.

Reynolds, C. W. (1987). Flocks, Herds, and Schools: A Distributed Behavioral Model. *ACM SIGGRAPH Computer Graphics*, *21*(4), 25-34.

Reynolds, C. W., et al. (1999). Steering behaviors for autonomous characters. *Game developers conference*, *1999*, 763-782.

Rodriguez Frangias, G. (2025). *Disuasión de aves mediante vehículos aéreos autónomos* [Tesis de maestría, Udelar. FI. IIE].

Roel, Á., Repetto, J. L., Bentancur, Á., Zerbino, P., Mangado, J., & Gorriti, P. (2013). *Jornada de Divulgación - Plantación y prácticas de manejo en Fruticultura* (inf. téc.). INIA.

Schiffner, I., & Srinivasan, M. V. (2016). Budgerigar flight in a varying environment: flight at distinct speeds? *Biology Letters*, *12*(6), 20160221.

Shi, H., & Wang, Z. (2022). Research on Strategies of Herding a Flock of Birds by Multiple UAVs. *34th Chinese Control and Decision Conference (CCDC)*, 3030-3035.

Strömbom, D., Mann, R. P., Wilson, A. M., Hailes, S., Morton, A. J., Sumpter, D. J. T., & King, A. J. (2014). Solving the shepherding problem: heuristics for herding autonomous, interacting agents. *Journal of The Royal Society Interface*, *11*(100), 20140719.

ThinkRobotics. (2025). PX4 vs ArduPilot: Complete Comparison Guide for Drone Developers [Accedido: 2026-01-10]. `https://thinkrobotics.com/blogs/learn/px4-vs-ardupilot-complete-comparison-guide-for-drone-developers`

Tinajero, R., & Estrella, R. R. (2015). Cotorra argentina (Myiopsitta monachus), especie anidando con éxito en el sur de la Península de Baja California. *Acta zoológica mexicana*, *31*(2).

Warrick, D. R., & Dial, K. P. (1998). Kinematic, aerodynamic and anatomical mechanisms in the slow, maneuvering flight of pigeons. *Journal of Experimental Biology*, *201*(5), 655-672.

Anexo A

A. Glosario

A.1. Paquete

Unidad organizacional fundamental de ROS 2 que encapsula código relacionado, mensajes, servicios y dependencias. El proyecto contiene cinco paquetes principales: boids para simulación de bandadas, drone_controller con algoritmos de pastoreo, gz_interface para gestionar Gazebo, y px4_msgs/px4_ros_com para comunicación con PX4.

A.2. Nodos

Procesos ejecutables independientes que realizan tareas específicas y se comunican mediante mensajería. Los nodos principales incluyen boids_node que simula la bandada de aves, múltiples instancias de shepherding_mode (una por dron pastor), gz_interface_node para gestión de simulación 3D, y nodos PX4 para control de bajo nivel.

A.3. Tópicos

Canales de comunicación unidireccional que implementan el patrón publicador-suscriptor para intercambio asíncrono de mensajes. Los tópicos críticos incluyen /boids_positions para posiciones de la bandada, /fleet_status para coordinación de drones, y tópicos /fmu/in/ y /fmu/out/ para comunicación con PX4.

A.4. Agentes

Entidades autónomas que ejecutan comportamientos inteligentes y toman decisiones basadas en su entorno y objetivos. En este proyecto, los agentes principales son los drones.

A.5. Clusters

Grupos o agrupaciones de datos similares identificados mediante algoritmos de clustering. En este proyecto, los clusters representan subgrupos de boids

(aves) que se forman dinámicamente según su proximidad espacial. El sistema implementa múltiples algoritmos de clustering (K-means, DBSCAN, X-means, Angular Sorted) para dividir la bandada en grupos manejables.

A.6. Centroides

Puntos representativos que definen el centro geométrico de un cluster.

A.7. PCA (Principal Component Analysis)

Técnica de reducción de dimensionalidad que identifica las direcciones de mayor varianza en un conjunto de datos. En el sistema, PCA se utiliza en el algoritmo X-means para dividir clusters de forma inteligente, calculando la matriz de covarianza de las posiciones de los boids y encontrando la dirección principal de dispersión para crear dos sub-clusters óptimos cuando es necesario subdividir un grupo.

A.8. BIC (Bayesian Information Criterion)

Criterio estadístico para selección de modelos que balancea la calidad del ajuste con la complejidad del modelo, penalizando configuraciones con excesivos parámetros. En el proyecto, BIC se implementa en el algoritmo X-means para determinar automáticamente el número óptimo de clusters al evaluar si dividir un grupo de boids mejora significativamente la representación de los datos.

A.9. Sistema distribuido

Arquitectura donde múltiples componentes autónomos operan de forma independiente, coordinando para lograr objetivos comunes. Los componentes toman decisiones locales basadas en su información disponible y se comunican entre sí para mantener coherencia global, ofreciendo ventajas como escalabilidad, tolerancia a fallos y procesamiento paralelo.

A.10. Sistema centralizado

Arquitectura donde un componente central coordina y controla las operaciones de todo el sistema, concentrando la toma de decisiones y el procesamiento de información.

A.11. DDS (Data Distribution Service)

Estándar de comunicación descentralizado de tipo publicación-suscripción, ideal para sistemas robóticos distribuidos.

A.12. Firmware

Software de bajo nivel, esencial y permanente, que reside en un dispositivo de hardware y proporciona instrucciones fundamentales para su funcionamiento y control, actuando como el sistema operativo básico del componente

A.13. Framework

Conjunto estandarizado de herramientas, librerías, convenciones y buenas prácticas que proporciona una estructura base para el desarrollo rápido y eficiente de aplicaciones.

A.14. Middleware

Software que actúa como una capa intermedia entre el sistema operativo y las aplicaciones, facilitando la comunicación, el intercambio de datos y la gestión entre sistemas dispares, permitiendo que funcionen juntos sin problemas.

Anexo B

B. Tecnologías utilizadas

B.1. ROS 2 (Robot Operating System 2)

Framework de middleware de código abierto para el desarrollo de sistemas robóticos distribuidos. En este proyecto, ROS 2 actúa como la infraestructura central que integra la simulación de boids, el control de múltiples drones PX4 y la interfaz con Gazebo.

B.2. Gazebo

Simulador robótico 3D de código abierto que proporciona entornos físicos realistas para pruebas y desarrollo de sistemas robóticos. En este proyecto, Gazebo Garden renderiza el mundo virtual donde se ejecuta la simulación, incluyendo modelos 3D de boids, drones y elementos del entorno como árboles.

B.3. C++

Lenguaje de programación compilado de alto rendimiento utilizado para implementar componentes críticos que requieren procesamiento computacional eficiente. En este proyecto, C++ se emplea para desarrollar los nodos de control de drones, los algoritmos de clustering y las estrategias de pastoreo, garantizando baja latencia en los cálculos realizados.

B.4. Python

Lenguaje de programación interpretado y de alto nivel que facilita el desarrollo rápido. En este proyecto, Python se utiliza para implementar el nodo de simulación de boids, el sistema de generación de escenarios y el cálculo de métricas de rendimiento.

B.5. PX4

Plataforma de software de autopiloto de código abierto para vehículos autónomos que proporciona control de vuelo, navegación y gestión de misiones. En este

proyecto, PX4 Autopilot gestiona el control de bajo nivel de cada dron en la simulación.

B.6. rqt_graph

Herramienta gráfica de ROS 2 que permite visualizar la arquitectura de comunicación entre nodos mediante un grafo de tópicos, servicios y acciones.

B.7. MAVSDK

Biblioteca de desarrollo de software que proporciona una interfaz de alto nivel para controlar sistemas MAVLink, simplificando la comunicación con autopilotos. Se utilizó para configurar ciertos parámetros de PX4.

B.8. MicroXRCEAgent

Agente del protocolo Micro XRCE-DDS que actúa como puente entre clientes DDS embebidos y el espacio de nombres DDS global. En este proyecto, MicroXRCEAgent permite la comunicación bidireccional entre las instancias PX4 (que ejecutan clientes Micro XRCE-DDS) y los nodos ROS 2.

B.9. QGroundControl

Estación de control terrestre de código abierto que proporciona una interfaz gráfica para planificación de misiones, configuración y monitoreo de vehículos autónomos. En este proyecto, QGroundControl puede utilizarse opcionalmente para visualizar el estado de los drones en tiempo real y realizar configuraciones manuales durante las pruebas de desarrollo.