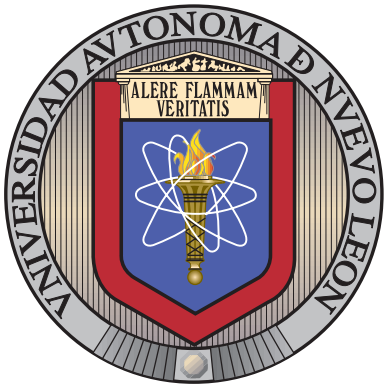
Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica



Reglas de comportamiento para enjambres de robots aéreos con percepción local

por

M. C. Mario Aguilera Ruiz

Como requisito parcial para obtener el Grado de

DOCTORADO EN INGENIERÍA ELÉCTRICA

Marzo 2021

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Subdirección de Estudios de Posgrado

Reglas de comportamiento para enjambres de robots aéreos con percepción local

por

M. C. Mario Aguilera Ruiz

como requisito parcial para obtener el grado de

DOCTORADO EN INGENIERÍA ELÉCTRICA

Marzo 2021

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica
Subdirección de Estudios de Posgrado

Los miembros del Comité de Tesis recomendamos que la Tesis "Reglas de comportamiento para enjambres de robots aéreos con percepción local", realizada por el alumno Mario Aguilera Ruiz, con número de matrícula 1500709, sea aceptada para su defensa como requisito para obtener el grado de Doctorado en Ingeniería Eléctrica.

El Comité de Tesis

Dr. Luis Martín Torres Treviño
Director

Dr. Juan Ángel Rodríguez Liñán
Revisor

Dr. Romeo Sánchez Nigenda
Revisor

Dr. Joaquín Gutiérrez Jagüey
Revisor

Dr. Oscar Salvador Salas Peña
Revisor

Vo. Bo.

Dr. Simón Martínez Martínez
Subdirector de Estudios de Posgrado

SUBDIRECCIÓN DE ESTUDIOS DE POSGRADO FIME

123

San Nicolás de los Garza, Nuevo León, marzo de 2021

Ciudad Universitaria Pedro de Alba s/n, C.P. 66455. A.P. 076 Suc. "F"
San Nicolás de los Garza, Nuevo León, México. Tels: (81) 8332 0903 /
Conm.: 8329 4020 / Fax: (81) 8332 0904

Agradecimientos

Quiero expresar mi más sincero agradecimiento a mi asesor de tesis, el Dr. Luis Martín Torres Treviño, por todo el apoyo y enseñanza durante el desarrollo de esta tesis.

A los miembros del comité de tesis, Dr. Juan Ángel Rodríguez, Dr. Romeo Sánchez, Dr. Joaquín Gutierrez y Dr. Oscar Salvador Salas, por sus valiosos comentarios y observaciones en la revisión de este trabajo.

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACYT) por otorgarme la beca número 434790 para realizar mis estudios.

A la Universidad Autónoma de Nuevo León (UANL) y la Facultad de Ingeniería Mecánica y Eléctrica (FIME) por la oportunidad de estudiar un posgrado, así como al Centro de Innovación, Investigación y Desarrollo en Ingeniería y Tecnología (CIIDIT) por permitirme el uso de sus instalaciones y equipo durante el desarrollo de esta tesis.

A los profesores del Posgrado en Ingeniería Eléctrica por sus enseñanzas durante mis estudios y a todos mis compañeros con los que conviví a lo largo de este posgrado.

Finalmente, quiero agradecer a mi familia por todo su apoyo y motivación.

Índice general

Agradecimientos iv

Resumen xv

1. Introducción 1

1.1. Motivación 1

1.2. Descripción del Problema 3

1.3. Hipótesis 5

1.4. Objetivos 5

1.4.1. Objetivo General 5

1.4.2. Objetivos Particulares 6

1.5. Contribuciones 6

1.6. Organización de la Tesis 7

2. Marco teórico 8

2.1. Enjambre de robots 8

2.1.1. Características 9

2.1.2. Aplicaciones 10

2.1.3. Arquitectura de un enjambre de robots 11

2.1.4. Comportamientos colectivos básicos 12

2.2. Robots Aéreos 14

2.3. Cuadrirrotor 16

2.3.1. Modelo Dinámico 17

2.3.2. Control de Orientación y Altitud 22

3. Esquema de comportamiento propuesto 25

3.1. Modelos inspirados en el comportamiento de animales sociales 25

3.2. Descripción de los miembros del enjambre 27

3.3. Modelo de percepción en 3 dimensiones 29

3.4. Modelo de percepción en 2 dimensiones 30

3.5. Reglas de comportamiento 31

3.5.1. Comportamiento de Repulsión 32

3.5.2. Comportamiento de Orientación 33

3.5.3. Comportamiento de Atracción 34

3.5.4. Comportamiento de Orientación-Atracción 35

3.5.5. Comportamiento de Explorar 36

3.5.6. Evasión de Obstáculos 36

3.5.7. Influencia 37

3.5.8. Dirección Deseada Final 38

4. Resultados **39**

4.1. Plataforma de Simulación 39

4.2. Métricas de Evaluación 42

4.2.1. Unión del enjambre 42

4.2.2. Número de colisiones 43

4.2.3. Polarización 44

4.2.4. Distancia promedio al centro del enjambre 44

4.3. Experimentos en tres dimensiones 45

4.3.1. Movimiento colectivo 46

4.3.2. Movimiento colectivo y obstáculo 50

4.4. Experimentos en dos dimensiones 55

4.4.1. Navegación con obstáculos 55

4.4.2. Agregación y movimiento colectivo 61

5. Conclusiones y Trabajo a Futuro **75**

A. Prototipo de cuadrirrotor **86**

A.1. Diseño 87

A.2. Componentes 88

A.2.1. Controlador de Vuelo 89

A.2.2. Módulo ESP32 89

A.2.3. Sensores de proximidad 90

A.2.4. Sistema de Propulsión 91

A.2.5. Transmisor y Receptor de Radio Control 91

A.3. Identificación de Parámetros 92

B. Código Simulador de Enjambre de Cuadrirrotores 95

Índice de figuras

2.1. Movimientos de alabeo, cabeceo y guiñada. 17

2.2. Sistemas coordenados de referencia. 18

2.3. Simulación del control de posición angular llevando los ángulos roll, pitch y yaw a cero a pesar del ruido agregado. 24

2.4. Simulación del control de altitud para levar el sistema a 1 m. 24

3.1. Comportamientos colectivos del modelo de Couzin. (A) Enjambre. (B) Toroide. (C) Grupo paralelo dinámico. (D) Grupo altamente paralelo. De Couzin et al. [1] 27

3.2. Prototipo de cuadrirrotor Starling 28

3.3. Representación del campo de visión de un sensor 29

3.4. Representación de la distribución de los sensores en tres dimensiones . 29

3.5. Representación de las zonas de percepción alrededor de cada robot. . 31

3.6. Zonas de comportamiento alrededor de cada individuo 32

3.7. Comportamiento de repulsión 33

3.8. Comportamiento de orientación 34

3.9. Comportamiento de atracción 35

3.10. Comportamiento de orientación-atracción 35

3.11. Comportamiento de explorar 36

4.1. Componentes de cada robot en el simulador. 40

4.2. Trayectoria seguida por el punto de influencia 46

4.3. Cantidad de colisiones por robot, polarización de grupo y distancia promedio al centro del grupo en el experimento en tres dimensiones. Zona de Influencia $\Delta_I = 5$ m. Cada punto en la gráfica es el promedio de 5 repeticiones. 48

4.4. Secuencia de comportamiento del experimento en tres dimensiones con N = 5, $\Delta_R = 1$ m, $\Delta_O = 2$ m, $\Delta_A = 1$ m y $\Delta_I = 5$ m 49

4.5. Secuencia de comportamiento del experimento en tres dimensiones con N = 10, $\Delta_R = 1$ m, $\Delta_O = 3$ m, $\Delta_A = 1$ m y $\Delta_I = 5$ m 49

4.6. Secuencia de comportamiento del experimento en tres dimensiones con N = 20, $\Delta_R = 1$ m, $\Delta_O = 3$ m, $\Delta_A = 5$ m y $\Delta_I = 5$ m 49

4.7. Trayectoria seguida por el punto de influencia y ubicación del obstáculo 50

4.8. Cantidad de colisiones por robot, polarización de grupo y distancia promedio al centro del grupo en el experimento en tres dimensiones con obstáculo. Zona de Influencia $\Delta_I = 5$ m. Cada punto en la gráfica es el promedio de 5 repeticiones. 52

4.9. Ejemplo de las trayectorias seguidas por el enjambre para evitar el obstáculo. Enjambre de 5 robots con $\Delta_R = 1.0$, $\Delta_O = 3.0$, $\Delta_A = 3.0$ y $\Delta_I = 5.0$ 53

4.10. Secuencia de comportamiento del experimento en tres dimensiones con N = 5, $\Delta_R = 1$ m, $\Delta_O = 3$ m, $\Delta_A = 1$ m y $\Delta_I = 10$ m 53

4.11. Secuencia de comportamiento del experimento en tres dimensiones con N = 10, $\Delta_R = 1$ m, $\Delta_O = 3$ m, $\Delta_A = 1$ m y $\Delta_I = 10$ m 54

4.12. Secuencia de comportamiento del experimento en tres dimensiones con N = 20, $\Delta_R = 1$ m, $\Delta_O = 5$ m, $\Delta_A = 5$ m y $\Delta_I = 10$ m 54

4.13. Arena para experimento de navegación con obstáculos. 56

4.14. Colisiones por robot para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el experimento de navegación con obstáculos. Cada punto en la gráfica es el promedio de 10 repeticiones. 57

4.15. Tiempo en que todos los robots completan el recorrido para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el experimento de navegación con obstáculos. Cada punto en la gráfica es el promedio de 10 repeticiones. 58

4.16. Ejemplo de simulación de navegación con obstáculos. Enjambre de 5 robots, $\Delta_R = 1$ m, $\Delta_O = 4$ m y $\Delta_A = 5$ m 60

4.17. Ejemplo de simulación de navegación con obstáculos. Enjambre de 10 robots, $\Delta_R = 0.5$ m, $\Delta_O = 5$ m y $\Delta_A = 4$ m 60

4.18. Ejemplo de simulación de navegación con obstáculos. Enjambre de 20 robots, $\Delta_R = 1$ m, $\Delta_O = 5$ m y $\Delta_A = 4$ m 60

4.19. Cantidad de colisiones por robot para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el modelo con 8 sensores. Cada punto en la gráfica es el promedio de 10 repeticiones. . 63

4.20. Polarización promedio durante el tiempo de simulación para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el modelo con 8 sensores. Cada punto en la gráfica es el promedio de 10 repeticiones. 64

4.21. Unión promedio durante el tiempo de simulación para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el modelo con 8 sensores. Cada punto en la gráfica es el promedio de 10 repeticiones. 65

4.22. Ejemplo de simulación de agregación y movimiento colectivo con 5 robots. $\Delta_R = 1$ m, $\Delta_O = 1$ m y $\Delta_A = 1$ m. Se muestra además la polarización del grupo. 67

4.23. Ejemplo de simulación de agregación y movimiento colectivo con 10 robots. $\Delta_R = 1$ m, $\Delta_O = 1$ m y $\Delta_A = 1$ m. Se muestra además la polarización del grupo. 67

4.24. Ejemplo de simulación de agregación y movimiento colectivo con 20 robots. $\Delta_R = 1$ m, $\Delta_O = 1$ m y $\Delta_A = 1$ m. Se muestra además la polarización del grupo. 68

4.25. Ejemplo de simulación de agregación y movimiento colectivo con 5 robots. $\Delta_R = 1.5$ m, $\Delta_O = 5$ m y $\Delta_A = 3$ m. Se muestra además la polarización del grupo. 69

4.26. Ejemplo de simulación de agregación y movimiento colectivo con 10 robots. $\Delta_R = 1.5m$, $\Delta_O = 4m$ y $\Delta_A = 5m$. Se muestra además la polarización del grupo. 70

4.27. Ejemplo de simulación de agregación y movimiento colectivo con 20 robots. $\Delta_R = 1m$, $\Delta_O = 5m$ y $\Delta_A = 5m$. Se muestra además la polarización del grupo. 70

4.28. Ejemplo de simulación de agregación y movimiento colectivo con 50 robots. $\Delta_R = 1m$, $\Delta_O = 5m$ y $\Delta_A = 5m$. 71

4.29. Polarización de grupo del enjambre de 50 robots. $\Delta_R = 1m$, $\Delta_O = 5m$ y $\Delta_A = 5m$. 72

4.30. Ejemplo de simulación de agregación y movimiento colectivo con 50 robots. $\Delta_R = 1m$, $\Delta_O = 15m$ y $\Delta_A = 5m$. 72

4.31. Polarización de grupo del enjambre de 50 robots. $\Delta_R = 1m$, $\Delta_O = 15m$ y $\Delta_A = 5m$. 73

4.32. Distancia promedio al centro del enjambre de 50 robots. $\Delta_R = 1m$, $\Delta_O = 15m$ y $\Delta_A = 5m$. 74

4.33. Número de robots en cada estado de comportamiento en la tarea de agregación y movimiento coordinado con 50 robots, $\Delta_R = 1m$, $\Delta_O = 15m$ y $\Delta_A = 5m$. Se muestra la media y desviación estándar de 10 repeticiones 74

A.1. Prototipo Cuadrirrotor Starling 86

A.2. Dimensiones del cuadrirrotor prototipo 87

A.3. Partes fabricadas con impresión 3D 88

A.4. Componentes del prototipo 88

A.5. Diagrama de conexión de componentes del prototipo 92

A.6. Experimento para determinar coeficiente de empuje 93

A.7. Experimento para determinar coeficiente de arrastre 93

Índice de tablas

2.1. Clasificación de drones propuesta por Hassanalian y Abdelke [2] . . . 15

2.2. Parámetros físicos del prototipo de cuadrirrotor 23

4.1. Parámetros de simulación de movimiento en tres dimensiones 47

4.2. Parámetros de simulación de movimiento en tres dimensiones con obstáculo. 51

4.3. Parámetros de simulación de navegación con obstáculos. 56

4.4. Resultados del experimento de navegación con obstáculos 59

4.5. Parámetros de simulación para agregación y movimiento colectivo . . 61

4.6. Resultados del experimento de agregación y movimiento colectivo . . 66

A.1. Características del controlador de vuelo 89

A.2. Características del módulo ESP32 90

A.3. Características del sensor VL53L0X 91

A.4. Parámetros físicos del prototipo de cuadrirrotor 94

Resumen

M. C. Mario Aguilera Ruiz.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Título del estudio: Reglas de comportamiento para enjambres de robots aéreos con percepción local.

Número de páginas: 114.

Candidato para obtener el grado de Doctorado en Ingeniería Eléctrica.

Objetivos y método de estudio: La robótica de enjambre toma inspiración del comportamiento emergente observado en animales sociales, para buscar la coordinación de grandes cantidades de robots relativamente simples, para completar tareas colectivas. En el caso de los enjambres de robots aéreos, aún existe el problema de desarrollar algoritmos que permitan la aplicación de comportamientos de movimiento colectivo al interior de un edifico, de forma descentralizada, haciendo uso de información sensorial local.

En el presente trabajo, se implementó un modelo de comportamiento bioinspirado, para la coordinación de un enjambre de robots aéreos de tipo cuadrirrotor. Este modelo de comportamiento depende de información local percibida por cada robot y está basado en reglas simples de repulsión, atracción, orientación e influencia, que actúan en zonas alrededor de cada robot. Además, se desarrolló una plataforma de

simulación con el fin de evaluar el modelo de comportamiento con distintos conjuntos de parámetros y en diferentes tareas. Esta plataforma de simulación incorpora el modelo dinámico de un cuadrirrotor, así como controladores PID para regular su vuelo. Paralelamente, se desarrolló un prototipo de robot aéreo cuadrirrotor, para obtener los parámetros físicos que permitan realizar simulaciones más cercanas a la realidad y que sirva como base para trabajos futuros donde se realicen experimentos con robots reales.

CONTRIBUCIONES Y CONCLUSIONES: Los resultados obtenidos en las simulaciones, muestran que es posible lograr un movimiento colectivo de los miembros de un enjambre de robots, con el modelo de comportamiento implementado. El cambio en los parámetros a nivel individual afecta el desempeño global del enjambre. Las simulaciones de enjambres con diferentes cantidades de robots, mostraron que el desempeño global tiende a decaer al incrementar el número de robots, por lo que es necesario incrementar el tamaño de las zonas de comportamiento de cada robot. El uso del parámetro de influencia muestra que es posible guiar al enjambre a una actividad deseada de forma indirecta, al mismo tiempo que los miembros del enjambre interactúan entre sí. Además, las simulaciones de tareas con obstáculos demostraron que el comportamiento es adaptable a cambios en el entorno, por lo que puede emplearse en espacios desconocidos o que no han sido previamente explorados.

Firma del asesor: 

Dr. Luis Martín Torres Treviño

Capítulo 1

Introducción

1.1 Motivación

Se ha observado en los enjambres de la naturaleza un comportamiento colectivo que emerge de las interacciones entre individuos y con el ambiente que los rodea, sin coordinación centralizada. Este comportamiento emergente es robusto, escalable y flexible, lo que permite que el enjambre realice tareas complejas fuera de las capacidades individuales de cada miembro [3, 4].

Tomando inspiración del comportamiento emergente observado en animales sociales (colonias de hormigas y abejas, parvadas de aves, cardúmenes de peces, etc.), la robótica de enjambre busca la coordinación de grandes cantidades de robots relativamente simples, para completar tareas colectivas. Por lo general, los robots en un enjambre tienden a ser descentralizados y con capacidades sensoriales y de procesamiento limitadas, aprovechando las propiedades observadas en la naturaleza para cumplir con su tarea [5].

La robótica de enjambre se ha propuesto como solución para diferentes comportamientos que requieren la coordinación de un grupo de robots. Algunos ejemplos de estos comportamientos son agregación, despliegue, movimiento en grupo, navegación y forrajeo [6]. Estas comportamientos se consideran la base para la realización

de diferentes tareas, principalmente en aplicaciones que involucran grandes espacios y grandes cantidades de tiempo, así como tareas potencialmente peligrosas para los humanos y los mismos robots. Como ejemplos de estas tareas podemos mencionar la búsqueda y recate en zonas de desastre, exploración y mapeo de lugares desconocidos, vigilancia, transportación colectiva, etc. [7, 8, 9, 10, 11].

El uso de robots aéreos, especialmente cuadrirrotores, ha crecido en años recientes. Entre las principales ventajas de los robots aéreos sobre los terrestres se puede mencionar su mayor velocidad para cubrir grandes terrenos en menor tiempo y la facilidad para pasar sobre obstáculos. En el caso de los cuadrirrotores, algunas de sus características más importantes incluyen su maniobrabilidad en espacios reducidos, la capacidad de despegue y aterrizaje vertical y la capacidad de vuelo estacionario sobre algún punto de interés [12].

En tiempos recientes han adquirido popularidad los espectáculos haciendo uso de cientos de drones, equipados con luces para desplegar imágenes en el cielo [13, 14]. Estos enjambres de drones no son autónomos ya que siguen trayectorias pre programadas o son controlados de forma centralizada.

Otras aplicaciones propuestas para los enjambres de drones consideran sus ventajas para cubrir grandes áreas y la exploración paralela de múltiples objetivos [15]. En el campo de la agricultura los enfoques de drones descentralizados tendrían gran importancia para el monitoreo y mapeo de cultivos y maleza [16]. Incluso se ha propuesto el uso de drones para mapear variables atmosféricas, usando mediciones locales para construir un mapa que permita generar trayectorias que maximicen la recolección de la información [17].

Otro de los campos que se puede beneficiar de los enjambres de robots es la logística y distribución de mercancía. Recientemente algunas compañías han comenzado a experimentar con el uso de drones para realizar la entrega de diferentes productos [18, 19, 20]. Existen además algunos estudios que proponen el uso de drones para la entrega de víveres o insumos médicos en áreas de desastres naturales o de

difícil acceso [21, 22]. Todas estas actividades con drones podrían beneficiarse con el uso de enjambres de robots, cuyas características permiten cubrir grandes áreas en menor tiempo, dividir el trabajo entre los miembros del enjambre e inclusive seguir operando a pesar de la pérdida de algunos miembros del enjambre.

1.2 Descripción del Problema

Muchos estudios existentes en el área de enjambres de robots utilizan Sistemas de Posicionamiento Global (GPS, por sus siglas en inglés) junto con sistemas de comunicación inalámbrica para coordinar grupos de robots aéreos en espacios al aire libre. Bürkle et al. [23] presentó un sistema multi-agente con 5 cuadrirrotores pero dependía en gran parte de una estación de control en tierra. Hoffmann et al. [24] implementó un sistema de 3 cuadrirrotores que se comunican entre ellos para generar trayectorias libres de colisiones. Vasarhelyi et al. [25] implementó un enjambre de 10 cuadrirrotores descentralizados con comunicación entre unidades para obtener vuelo coordinado y posteriormente lo mejoran con un sistema de optimización evolutiva logrando realizar experimentos con 30 cuadrirrotores [26]. Este es el enjambre descentralizado de robots aéreos más grande reportado en la literatura, implementado en un entorno al aire libre.

En el caso de enjambres de robots que operan en entornos interiores, no es posible utilizar información de posición de GPS debido a la atenuación de las señales, lo que disminuye considerablemente la precisión y fiabilidad de la información. Uno de los enfoques para la obtención de la posición de los robots en estos entornos interiores es el uso de sistemas de captura de movimiento. Uno de los ejemplos más representativos es el mostrado por Kushleyev et al. [27] con 20 cuadrirrotores miniatura, donde las trayectorias de cada uno son calculadas por una computadora central. La desventaja de este sistema de captura de movimiento es que se requiere una instalación previa de un sistema fijo de cámaras que obtienen la posición de los robots, lo cual no lo hace viable para aplicaciones en ambientes desconocidos;

además mucho del procesamiento se realiza en una computadora central.

Un enfoque más deseable para la implementación de enjambres de robots al interior de edificios consiste en evitar el uso de información global y control centralizado; se prefiere el uso de percepción local, es decir, información obtenida por medio de los sensores a bordo del robot. El uso de este tipo de percepción local permite obtener la posición relativa de los robots vecinos. Un ejemplo de este tipo de percepción local son los sistemas de visión computacional a bordo. Saska et al. [28] presenta un sistema de localización relativa, equipando a los cuadrirrotores con patrones en blanco y negro y una cámara monocular para detectar estos patrones, realizando el procesamiento de imágenes a bordo. Otro ejemplo es el mostrado por Walter et al. [29] donde utilizan un sistema de visión computacional para la detección mutua de marcadores Ultravioleta, logrando obtener la posición relativa y orientación de los drones. Para robots aéreos en entornos interiores, por ejemplo dentro de un edificio, existe una límite en el tamaño del robot, para que pueda desplazarse por pasillos o entrar por puertas. Esta limitante en el tamaño lleva a una restricción en la capacidad de carga del cuadrirrotor. El problema con los sistemas de visión a bordo es el alto costo computacional que conlleva un aumento en el peso del cuadrirrotor.

Una alternativa es el uso de sensores más simples que permitan obtener la posición relativa de los otros miembros del enjambre; algunos ejemplos son sensores ultrasónicos e infrarrojos. Stirling et al. [30] presenta un grupo de tres cuadrirrotores equipados con sensores infrarrojos para obtener la posición y orientación relativa de robots cercanos. Dos de los robots permanecían estáticos sujetos al techo y servían como puntos de referencia para el tercer robot desplazándose. Otro ejemplo es el mostrado por Coppola et al. [31] implementando un sistema de comunicación a bordo para obtener la localización relativa de 3 drones. Los robots se comunican entre ellos para intercambiar información como velocidad, altitud orientación y distancia relativa basada en la intensidad de la señal, con lo cual logran volar en un espacio reducido y evitar colisiones. McGuire et al. [32] presenta una solución de navegación basada en comportamientos simples para realizar una tarea de exploración y regreso

a casa. Los drones implementados utilizan sensores de medición de distancia para detectar obstáculos y comunicación entre robots para obtener una distancia relativa entre ellos y la dirección deseada de cada uno con el fin de no empalmar sus trayectorias en la exploración.

Estos estudios muestran resultados del uso de información relativa de posición para la implementación de robots aéreos autónomos, sin embargo, en algunas ocasiones sólo se considera un pequeño número fijo de miembros en el enjambre. En algunos casos solo se consideran pocos comportamientos como evitar colisiones o navegación de forma independiente entre robots. Por estas razones es de interés la aplicación de comportamientos de movimiento colectivo para robots aéreos de forma descentralizada, haciendo uso de información sensorial local y comunicación local con sus vecinos.

1.3 Hipótesis

Es posible inducir un comportamiento de movimiento colectivo en un enjambre de robots aéreos tipo cuadrirrotor, mediante la propuesta de un conjunto de reglas de repulsión, orientación, atracción e influencia basadas en información de percepción local. Además, este comportamiento emergente debe cumplir con las características de robustez, escalabilidad y flexibilidad.

1.4 Objetivos

1.4.1 Objetivo General

Implementar un modelo de comportamiento con reglas de repulsión, atracción, orientación e influencia, en un enjambre de robots cuadrirrotores que utilicen infor-

mación local.

1.4.2 Objetivos Particulares

- Desarrollar una plataforma para simular un conjunto de robots móviles tipo cuadrirrotor con movimiento en tres dimensiones.

- Desarrollar un modelo de comportamiento local basado en reglas de repulsión, atracción, orientación e influencia, para regular el comportamiento del enjambre de robots.

- Evaluar el modelo de comportamiento simulando diferentes tareas con el enjambre de cuadrirrotores.

- Desarrollar un prototipo de cuadrirrotor capaz de volar al interior de un edificio.

1.5 Contribuciones

Las principales contribuciones de la presente tesis son las siguientes:

- La adaptación del modelo de comportamiento bioinspirado para operar en base a información local proveniente de los sensores del robot.
- El desarrollo de una plataforma para simular enjambres de cuadrirrotores. Cada robot es representado con un modelo dinámico y se consideran los detalles de hardware para la percepción del robot, con el fin de que las simulaciones sean lo más cercanas posibles al comportamiento real.

1.6 Organización de la Tesis

El capítulo 2 define algunos conceptos teóricos relevantes para este trabajo. Se describe que son los enjambres de robots, sus características, aplicaciones y tareas principales. Se describen también los robots aéreos, más específicamente el cuadrirrotor y se presenta su modelo matemático y los esquemas de control de posición angular, altitud y velocidad.

En el capítulo 3 se presenta el esquema de comportamiento inspirado en modelos de movimiento de grupos de animales. Se muestran las consideraciones y limitaciones de percepción de los miembros del enjambre y las reglas de comportamiento de interacción de los robots con el ambiente y con otros robots.

El capítulo 4 muestra las métricas utilizadas para evaluar el desempeño del enjambre y se presentan también los resultados de los experimentos simulados.

Finalmente, en el capítulo 5 se presentan las conclusiones de este trabajo y algunas recomendaciones de trabajo a futuro.

Capítulo 2

Marco teórico

En este capítulo se describen las características y comportamientos principales de los enjambres de robots, así como las aplicaciones potenciales que se han planteado. Se presenta además un resumen de las características de los robots aéreos y el modelo matemático de un cuadrirrotor, en conjunto con sus esquemas de control de posición angular y altitud.

2.1 Enjambre de robots

La inteligencia colectiva es una disciplina de la inteligencia artificial inspirada en la observación del comportamiento emergente en seres sociales como enjambres de abejas, colonias de hormigas, cardúmenes de peces o parvadas de aves, entre otros. Estos seres poseen habilidades individuales muy simples, pero al vivir en grandes grupos logran realizar tareas más complejas como la búsqueda y recolección de comida o escapar de algún depredador.

Los enjambres de robots se basan en la aplicación de la inteligencia colectiva al diseño y construcción de sistemas multi-robot para que exhiban características similares a las observadas en animales sociales. Utilizando grandes cantidades de robots simples, con reglas de comportamiento sencillas y sin el uso de una unidad

central para controlar el enjambre, se espera que un comportamiento deseado emerja de la interacción entre robots y la interacción de robots con el ambiente.

En la naturaleza los enjambres suelen estar formados por cientos o miles de individuos, sin embargo los enjambres de robots reportados en la literatura están formados por pocas unidades. Esto se debe a que un gran obstáculo para poder experimentar con un gran número de robots reales recae en el elevado costo de fabricación y mantenimiento de los robots. Es difícil establecer un límite inferior en el número de robots de un enjambre; en lugar de eso algunos autores señalan que el campo debería estar abierto a trabajos realizados con pocos robots, pero con la visión de escalar el comportamiento a enjambres de mayor tamaño [3]. De manera similar Beni [33] menciona que el uso del término enjambre de robots no debe ser en función del número de unidades en el sistema, en lugar de eso el factor principal debe ser los principios fundamentales de la coordinación del sistema multi-robot. Es decir, que la arquitectura del enjambre debe ser escalable de unas pocas unidades a cientos o miles.

2.1.1 Características

El objetivo de los enjambres de robots es lograr comportamientos robustos, escalables y flexibles como los observados en animales sociales [4, 3]. Algunas ventajas en comparación con otros esquemas de agentes robóticos son:

- Robustez: El enjambre debe poder continuar operando a pesar de fallas en algunos individuos y cambios o disturbios en el ambiente. Esta característica puede ser atribuida a la redundancia del sistema y la coordinación descentralizada.
- Escalable: Debido a que las interacciones entre los miembros son locales, el enjambre es adaptable a un amplio rango en el tamaño del enjambre.

- Económico: Gracias a la simplicidad de diseño y la producción en masa, el costo total de un enjambre de robots sería inferior al de un sólo robot complejo.
- Descentralizado: Las reglas de comportamiento de cada robot le permiten al enjambre completar su tarea sin el uso de un control centralizado. Esto es especialmente útil en ambientes donde las comunicaciones pueden presentar interrupciones o retardos de tiempo.
- Flexibilidad: Un enjambre debe ser capaz de realizar diversas tareas con la misma arquitectura y mínimos cambios de programación.

2.1.2 Aplicaciones

Un enjambre de robots puede ser útil en actividades que involucran grandes cantidades de tiempo y espacio o que pueden resultar peligrosas para los robots [5]. Algunos ejemplos de aplicaciones potenciales se encuentran en:

- Búsqueda de sobrevivientes en desastres [7].
- Detección de derrames de petróleo en el mar [34].
- Búsqueda de objetivos [35].
- Vigilancia y monitoreo de recursos [36].
- Aplicaciones militares [37].
- Exploración espacial [38].
- Transporte colectivo de objetos [11]

2.1.3 Arquitectura de un enjambre de robots

El trabajo presentado en [39] muestra un diagrama general de las etapas necesarias para la producción de movimiento en grupo, tomando como referencia diferentes modelos encontrados en la literatura. Este diagrama consta de 5 etapas que se describen a continuación:

Percepción. Es la forma en que cada miembro del enjambre obtiene información del ambiente que lo rodea. Por lo general esta información es obtenida por medio de sensores que miden diversas magnitudes físicas, sistemas de visión computacional e inclusive sistemas de comunicación entre los miembros del enjambre para intercambiar información. La etapa de percepción es la primera, ya que convierte el mundo real en información que puede ser usada por las demás etapas del modelo del enjambre.

Detección de miembros del enjambre. En esta etapa se utiliza la información de la etapa de percepción para determinar el conjunto de todos los robots en el rango de detección. Este conjunto de individuos detectados puede incluir información como posición y velocidad, entre otros, dependiendo de las necesidades de cada algoritmo.

Selección de vecinos. Tomando el conjunto de individuos detectados anteriormente, ahora se seleccionan sólo aquellos que influyen en el cálculo del movimiento del robot. Por ejemplo, en esta etapa se separarían los vecinos que están en la zona de repulsión o atracción, de acuerdo a las políticas establecidas por el modelo de comportamiento.

Cálculo de movimiento. Esta etapa se encarga de calcular el siguiente movimiento del robot, tomando como entradas los conjuntos de vecinos seleccionados en la etapa previa. Sólo se calcula la dirección que debe tomar el vehículo. El resultado es sólo una representación, como puede ser un vector. El movimiento real del robot se ejecuta en la etapa final.

Movimiento físico. Tomando como entrada la dirección calculada previamente, se traduce para que pueda ser ejecutada dependiendo del sistema, por ejemplo en velocidades de los motores del vehículo.

2.1.4 Comportamientos colectivos básicos

Existen varios comportamientos colectivos considerados como básicos, los cuales son fundamentales para la realización de tareas más complejas. Estos comportamientos se pueden dividir en organización espacial, navegación y toma de decisiones colectivas [4].

2.1.4.1 Comportamientos de organización espacial

Estos comportamientos están orientados hacia la organización y distribución del enjambre en un determinado ambiente.

- Agregación. Es el comportamiento más simple y consiste en agrupar un conjunto de robots muy cerca entre sí. Es de gran importancia al permitir la interacción entre los robots y ser la base para otros comportamientos como el movimiento en grupo.
- Formación. Consiste en que los robots mantengan cierta distancia entre ellos para formar un determinado patrón o figura repetible. Este comportamiento es útil en tareas en las que se puede requerir transportar objetos grandes.
- Auto ensamblaje. Se trata de las acciones que permiten a los robots conectarse entre sí físicamente, con el fin de obtener una estructura de mayor tamaño que permita realizar acciones más complejas. El contar con una estructura de mayor tamaño ayuda al enjambre a completar tareas en terrenos complicados, o para superar obstáculos.

- Agrupación de objetos. Consiste en que los robots muevan objetos distribuidos en el ambiente, para agruparlos en algún punto. Esta es una tarea muy común en los seres vivos como las hormigas y termitas, que puede ser aplicable a la recolección de alimentos o la construcción.

2.1.4.2 Comportamientos de navegación

Los comportamientos de navegación están orientados hacia la coordinación del movimiento de un enjambre de robots.

- Exploración colectiva. El objetivo de este comportamiento es desplegar los robots en un ambiente desconocido con el fin de crear una malla o arreglo para realizar comunicaciones entre los robots y guiar a otros hacia determinado objetivo.
- Movimiento coordinado. Basado en el movimiento de las aves y peces, este comportamiento busca que los robots se muevan en formación, lo cual es de gran importancia para que los robots puedan navegar por un ambiente sin colisiones entre ellos.
- Transportación colectiva. Cuando un objeto es demasiado grande o pesado para ser movido por un sólo robot, la transportación colectiva busca que un grupo de robots trabajen en conjunto para mover ese objeto. Es importante que exista coordinación entre los robots para que se muevan en la misma dirección.

2.1.4.3 Comportamientos de decisión colectiva

Este comportamiento trata el problema de la influencia que tienen los robots entre sí al momento de tomar una decisión. Ejemplos de toma de decisiones pueden ser el logro de consenso o la distribución de tareas.

- Consenso. Cuando hay varias alternativas, el enjambre debe llegar a un consenso sobre la mejor opción para maximizar su desempeño. Esta es una tarea muy compleja ya que la solución óptima puede cambiar o no ser perceptible por las capacidades del robot.

- Distribución de tareas. En este comportamiento los robots se distribuyen para realizar diferentes tareas, con el objetivo de maximizar el desempeño del sistema.

2.2 Robots Aéreos

Un dron es un tipo de robot aéreo generalmente conocido como Vehículo Aéreo No Tripulado (VANT) o UAV (del inglés Unmanned Aerial Vehicle). Es una aeronave que no lleva un piloto a bordo y es operada por un sistema de control de vuelo autónomo a bordo o por control remoto [40].

Las investigaciones realizadas con el fin de mejorar el tiempo de vuelo y capacidad de carga de los robots aéreos, han dado como resultado diferentes configuraciones o plataformas con distintas características. Las principales configuraciones pueden ser clasificadas en las siguientes categorías [41]:

- Dirigibles. Son estructuras más ligeras que el aire, simples, económicas y con la capacidad de mantenerse en vuelo por largos periodos de tiempo. Entre sus limitaciones se encuentra su gran tamaño que lo hace sensible al viento y no le permite viajar a altas velocidades.

- Aeronaves de ala fija. Aeroplanos que pueden viajar a altas velocidades y durante periodos de tiempo prolongados, lo que les permite recorrer distancias relativamente largas. Para despegar o aterrizar requieren de una pista horizontal o un lanzamiento catapultado. Otras de sus desventajas es que no pueden permanecer flotando en un mismo sitio ni moverse hacia atrás.

- Aeronaves de ala rotativa. Los helicópteros tienen la capacidad de despegar y aterrizar verticalmente, mantenerse flotando en un mismo punto y alta maniobrabilidad para moverse lateralmente o hacia atrás. Pueden tener diferentes configuraciones de rotores. La desventaja es que requieren mayor potencia para mantenerse en vuelo.

Existen además otras plataformas que no caen en las configuraciones mencionadas anteriormente y cuyo uso no es tan extendido. Algunos ejemplos son los diseños inspirados en aves o insectos con alas batientes, además de diseños híbridos o convertibles que pueden cambiar su forma en pleno vuelo.

Los drones varían ampliamente en sus características como tamaño, masa y tiempo de vuelo, dependiendo de la aplicación para la que se requieren. Muchos autores han propuesto distintas clasificaciones de acuerdo a estas características. Un ejemplo es la clasificación propuesta por Hassanalian y Abdelke [2] que considera el reciente interés de desarrollar drones muy pequeños. Esta clasificación se puede resumir en la Tabla 2.1.

Tabla 2.1: Clasificación de drones propuesta por Hassanalian y Abdelke [2]

Clasificación	Rango de Envergadura	Rango de Peso
UAV	$2\text{ m} < \text{R} \leq 61\text{ m}$	$5\text{ Kg} < \text{W} \leq 15000\text{ Kg}$
μUAV	$1\text{ m} < \text{R} \leq 2\text{ m}$	$2\text{ Kg} < \text{W} \leq 5\text{ Kg}$
MAV	$15\text{ cm} < \text{R} \leq 1\text{ m}$	$50\text{ g} < \text{W} \leq 2\text{ Kg}$
NAV	$2.5\text{ cm} < \text{R} \leq 15\text{ cm}$	$3\text{ g} < \text{W} \leq 50\text{ g}$
PAV	$0.25\text{ cm} < \text{R} \leq 2.5\text{ cm}$	$0.5\text{ g} < \text{W} \leq 3\text{ g}$
SD	$1\text{ mm} \leq \text{R} \leq 0.25\text{ cm}$	$0.005\text{ g} \leq \text{W} \leq 0.5\text{ g}$

Dejando de lado las aplicaciones militares, el uso de vehículos aéreos no tripulados ha crecido rápidamente en aplicaciones civiles. Algunos ejemplos de estas aplicaciones son:

- Búsqueda y rescate.
- Agricultura de precisión.
- Entrega de bienes.
- Vigilancia.
- Inspección de infraestructura.

2.3 Cuadrirrotor

Considerando las características de las diferentes plataformas de robots aéreos, los Micro Vehículos Aéreos (MAVs, del inglés Micro Aerial Vehicles) de ala rotativa presentan ventajas para volar en espacios reducidos por sus pequeñas dimensiones. Además pueden volar en cualquier dirección o permanecer flotando sobre un punto en específico. La configuración estándar del helicóptero con un rotor principal y un rotor de cola es mecánicamente complejo para producir, razón por la cual se ha optado por configuraciones multi-rotor en distintos proyectos robóticos.

El cuadrirrotor es un vehículo aéreo con cuatro rotores horizontales en configuración de cruz. El movimiento del cuadrirrotor se crea al variar la velocidad de los rotores. Al aumentar la velocidad de los cuatro rotores se genera un movimiento vertical. En la Figura 2.1 se pueden apreciar las rotaciones en los tres ejes del cuadrirrotor. El alabeo (roll) o rotación alrededor del eje longitudinal se consigue con una diferencia en el empuje de los rotores 2 y 4. Es decir, al aumentar la velocidad de un rotor y disminuir la del otro se consigue un giro alrededor del eje x, lo que genera un movimiento traslacional en el eje y. La rotación alrededor del eje y o eje transversal se conoce como cabeceo (pitch) y se consigue con una diferencia en el empuje de los rotores 1 y 2. Para obtener una rotación en el eje z o eje vertical es necesario aumentar la velocidad del par de rotores con el mismo sentido de giro y disminuir la velocidad de los otros dos rotores. Esta rotación se conoce como guiñada (yaw).

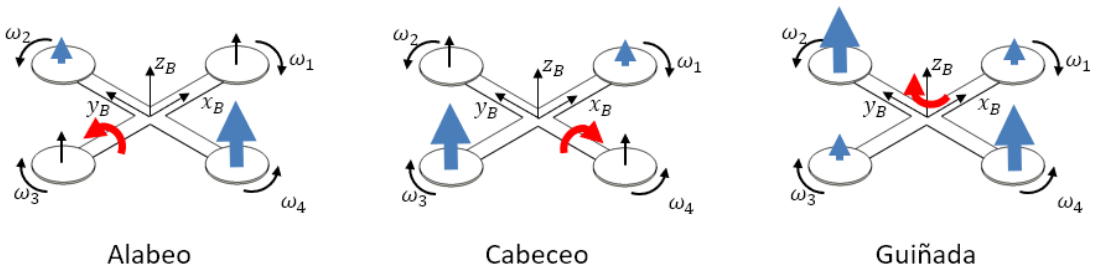


Figura 2.1: Movimientos de alabeo, cabeceo y guiñada.

2.3.1 Modelo Dinámico

Para comprobar las reglas de comportamiento del enjambre, es necesario simular un enjambre de robots tan cercano a la realidad como sea posible. Para este caso, cada miembro del enjambre será un robot cuadrirrotor, por lo que un modelo matemático representa la cinemática y dinámica de esta configuración.

Para simplificar el modelado del cuadrirrotor y aún obtener un modelo lo suficientemente preciso, se hacen las siguientes suposiciones [42]:

- La estructura del cuadrirrotor es rígida y simétrica.
- El centro de gravedad del cuadrirrotor coincide con el origen del sistema de coordenadas de referencia unido al cuerpo.
- Las constantes de empuje y arrastre son proporcionales al cuadrado de la velocidad del rotor.
- No se consideran algunos efectos aerodinámicos como la fuerza de arrastre rotacional, deformación de las hélices o velocidad del viento alrededor.

El primer paso para el modelado matemático es establecer un marco de referencia inercial $\{A\}$ y un marco de referencia unido al cuerpo del cuadrirrotor $\{B\}$ [43, 44, 45], como se muestra en la Figura 3.4. Los rotores están numerados i=1,2,3,4

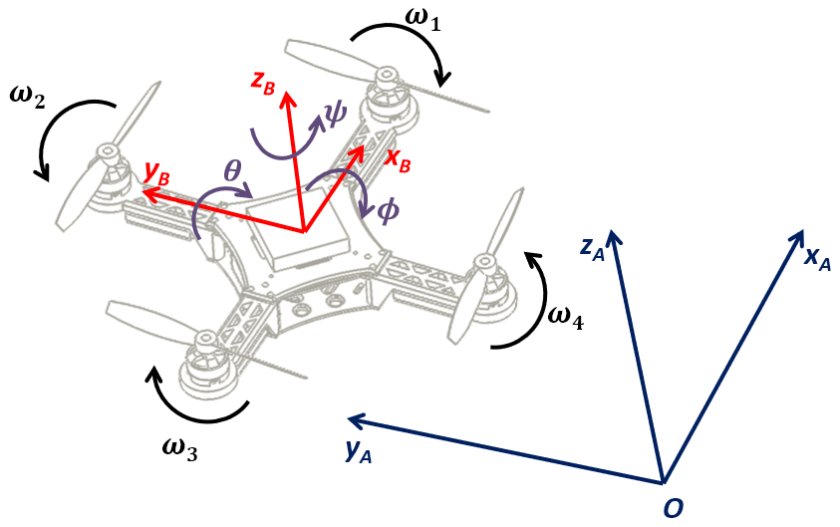


Figura 2.2: Sistemas coordenados de referencia.

y la velocidad angular de los rotores está dada por ω_i. En el marco de referencia inercial, la posición cartesiana del cuadrirrotor está dada por ξ y la orientación η por los ángulos Roll-Pitch-Yaw:

$$\xi = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \eta = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{2.1}$$

La matriz de rotación que transforma las coordenadas de posición del cuadrirrotor del marco de referencia $\{B\}$ al marco de referencia inercial $\{A\}$ está dada por:

$$R = \begin{bmatrix} C_\theta C_\psi & S_\phi S_\theta C_\psi - C_\phi S_\psi & C_\phi S_\theta C_\psi + S_\phi S_\psi \\ C_\theta S_\psi & S_\phi S_\theta S_\psi + C_\phi C_\psi & C_\phi S_\theta S_\psi - S_\phi C_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \tag{2.2}$$

donde C_x representa $\cos x$ y S_x representa $\operatorname{sen} x$.

En el marco de referencia $\{B\}$ la velocidad angular del cuadrirrotor se define como $\nu = [p \quad q \quad r]^T$ y su relación con el cambio de orientación $\dot{\eta}$ en el marco de

referencia inercial está dada por:

$$\nu = M\dot{\eta} \tag{2.3}$$

con

$$M = \begin{bmatrix} 1 & 0 & -S_\theta \\ 0 & C_\phi & C_\theta S_\phi \\ 0 & -S_\phi & C_\phi C_\theta \end{bmatrix} \tag{2.4}$$

Si las perturbaciones en vuelo estático son pequeñas, la matriz de transformación entre la tasa de cambio de los ángulos de orientación $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ y la velocidad angular del cuadrirrotor (p, q, r) se puede considerar una matriz identidad [46].

2.3.1.1 Dinámica traslacional

Para que el cuadrirrotor flote en el aire de manera estable se requiere que la suma del empuje de los cuatro rotores sea igual al peso del dron. La velocidad angular del rotor ω_i crea una fuerza de empuje en la dirección del eje del rotor, aproximada por:

$$f_i = k_T \omega_i^2 \tag{2.5}$$

donde k_T es una constante de sustentación o empuje. La fuerza total de empuje T en el marco de referencia del cuadrirrotor está dada por:

$$T = \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} f_i \end{bmatrix} \tag{2.6}$$

La fuerza de arrastre generada por la resistencia del aire [47] es incluida de forma simplificada como una fuerza proporcional a la velocidad lineal del cuadrirrotor:

$$F_D = \begin{bmatrix} -A_{dx} & 0 & 0 \\ 0 & -A_{dy} & 0 \\ 0 & 0 & -A_{dz} \end{bmatrix} \dot{\xi} \tag{2.7}$$

Por último la fuerza de gravedad actuando sobre el cuadrirrotor está representada por:

$$F_G = m \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{2.8}$$

siendo m la masa del cuadrirrotor y g la aceleración debida a la gravedad.

Utilizando las ecuaciones Newton-Euler sobre movimiento traslacional obtenemos:

$$m\ddot{\xi} = RT - F_G - F_D \tag{2.9}$$

2.3.1.2 Dinámica rotacional

El par alrededor de los ejes x y y está dado por la diferencia entre los pares de los rotores en ejes contrarios, mientras que el par alrededor del eje z está dado por la suma de pares de cada rotor:

$$\begin{aligned} \tau_\phi &= l\ k_T\ (\omega_2^2 - \omega_4^2) \\ \tau_\theta &= l\ k_T\ (\omega_3^2 - \omega_1^2) \end{aligned} \tag{2.10}$$

$$\tau_\psi = d\left(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2\right)$$

donde d es una constante de arrastre y l es la distancia entre el centro del cuadrirrotor y uno de los rotores.

Usando las ecuaciones de Newton-Euler, la dinámica rotacional se obtiene como:

$$I\dot{\nu} + \nu \times (I\nu) = \tau \tag{2.11}$$

Considerando que el cuadrirrotor tiene una estructura simétrica, la matriz de inercia I resulta en una matriz diagonal de la forma:

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{2.12}$$

Utilizando las ecuaciones 2.9 y 2.11, el modelo dinámico del cuadrirrotor es:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} (C_\phi S_\theta C_\psi + S_\phi S_\psi)\frac{1}{m}T - A_{dx}\dot{x} \\ (C_\phi S_\theta S_\psi - S_\phi C_\psi)\frac{1}{m}T - A_{dy}\dot{y} \\ -g + (C_\phi C_\theta)\frac{1}{m}T - A_{dz}\dot{z} \\ \frac{1}{I_{xx}}\tau_\phi + \frac{I_{yy}-I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} \\ \frac{1}{I_{yy}}\tau_\theta + \frac{I_{zz}-I_{xx}}{I_{yy}}\dot{\phi}\dot{\psi} \\ \frac{1}{I_{zz}}\tau_\psi + \frac{I_{xx}-I_{yy}}{I_{zz}}\dot{\phi}\dot{\theta} \end{bmatrix} \tag{2.13}$$

2.3.2 Control de Orientación y Altitud

El cuadrirrotor tiene 6 grados de libertad, sin embargo, sólo cuatro pueden ser controlados de forma directa: la altitud y los tres ángulos de orientación. Para llevar al cuadrirrotor a la orientación deseada y a la altitud deseada se utilizan controladores Proporcional - Integral - Derivativo (PID) [48]. Las variables de control del modelo son las velocidades de los rotores ω_i. A partir de las ecuaciones (2.6) y (2.10) las velocidades de los rotores se calculan como:

$$\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{2k_T l} & -\frac{1}{4d} & \frac{1}{4k_T} \\ \frac{1}{2k_T l} & 0 & \frac{1}{4d} & \frac{1}{4k_T} \\ 0 & \frac{1}{2k_T l} & -\frac{1}{4d} & \frac{1}{4k_T} \\ -\frac{1}{2k_T l} & 0 & \frac{1}{4d} & \frac{1}{4k_T} \end{bmatrix} \begin{bmatrix} u_\phi \\ u_\theta \\ u_\psi \\ u_T \end{bmatrix} \tag{2.14}$$

Definiendo la altitud y orientación deseada como z_d y $[\phi_d \ \theta_d \ \psi_d]$ respectivamente, los errores entre estas variables deseadas y las variables reales se definen por:

$$\begin{aligned} e_z &= z_d - z \\ e_\phi &= \phi_d - \phi \\ e_\theta &= \theta_d - \theta \\ e_\psi &= \psi_d - \psi \end{aligned}$$

con las leyes de control PID:

$$\begin{aligned} u_\phi &= K_{p\phi}\, e_\phi + K_{i\phi} \int_0^T e_\phi dt + K_{d\phi}\, \dot{e}_\phi \\ u_\theta &= K_{p\theta}\, e_\theta + K_{i\theta} \int_0^T e_\theta dt + K_{d\theta}\, \dot{e}_\theta \end{aligned} \tag{2.15}$$

$$u_\psi = K_{p\psi}\, e_\psi + K_{i\psi} \int_0^T e_\psi dt + K_{d\psi}\, \dot{e}_\psi$$
$$u_T = \frac{m}{C_\phi C_\theta} \left(g + K_{pz}\, e_z + K_{iz} \int_0^T e_z dt + K_{dz}\, \dot{e}_z \right)$$

Los coeficientes K_p, K_i y K_d son las ganancias proporcional, integral y derivativa del controlador, respectivamente.

Para simular el cuadrirrootor y probar los controladores de orientación y altitud, los parámetros físicos del modelo dinámico se muestran en la Tabla 2.2 y fueron obtenidos empíricamente a partir de un prototipo físico de cuadrirrotor (ver Apéndice A).

Tabla 2.2: Parámetros físicos del prototipo de cuadrirrotor

Parámetro	Valor	Unidades
m	0.42	Kg
l	0.11	m
I_{xx}	7.92×10^{-6}	Kg m^2
I_{yy}	7.92×10^{-6}	Kg m^2
I_{zz}	1.56×10^{-5}	Kg m^2
k_T	8.50×10^{-7}	N/(rad/s)^2
d	1.46×10^{-8}	N m /(rad/s)^2

Para una prueba más realista, se simulan errores en los sensores agregando ruido gaussiano a la velocidad angular y altitud del cuadrirrotor, con una desviación estándar de 0.1 rad/s para la velocidad angular y 0.1 m para la altitud. En la Figura 2.3 la tarea del controlador es mantener el cuadrirrotor suspendido en el aire, manteniendo los ángulos roll, pitch y yaw a cero. Los ángulos iniciales del cuadrirrotor se toman al azar entre -15 y 15 grados. Se puede observar como los tres ángulos alanzan un valor cercano a cero después de un segundo de simulación, con algunas oscilaciones debidas al ruido agregado. Para el control de altitud la tarea es elevar el cuadrirrotor a una altura de 1 m y mantener esta altitud como se muestra

en la Figura 2.4.

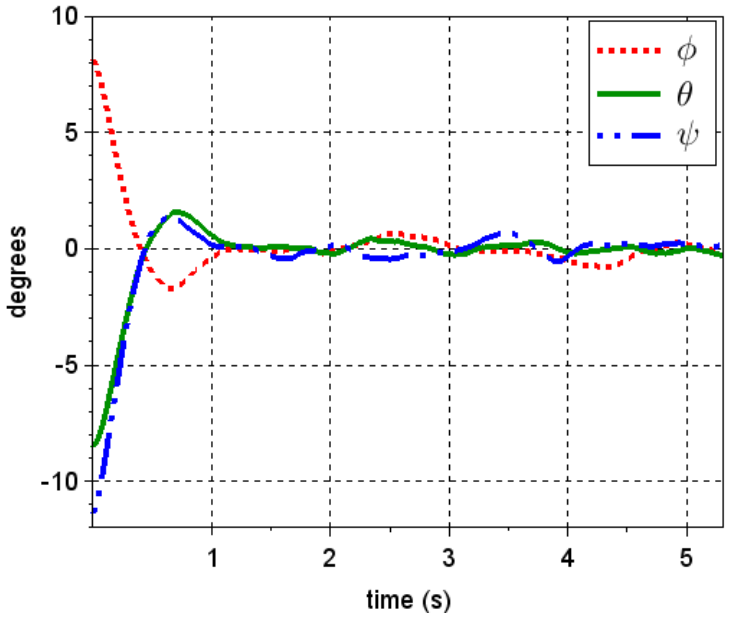


Figura 2.3: Simulación del control de posición angular llevando los ángulos roll, pitch y yaw a cero a pesar del ruido agregado.

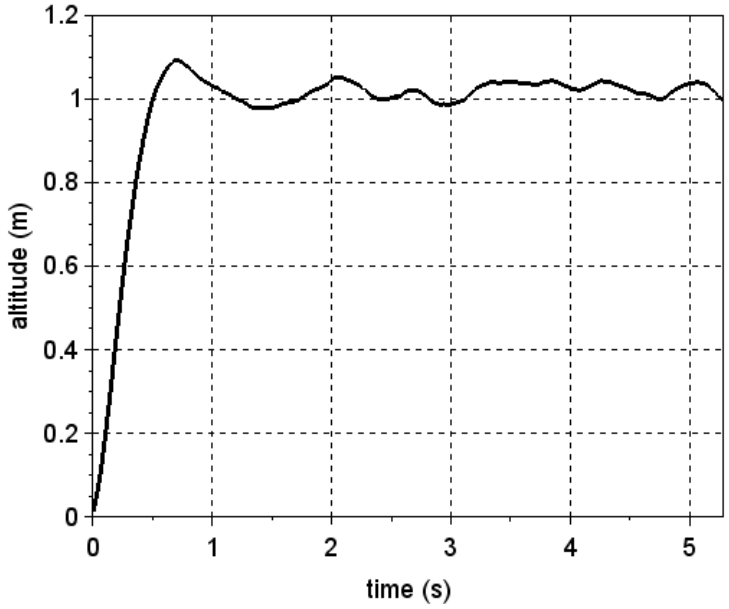


Figura 2.4: Simulación del control de altitud para llevar el sistema a 1 m.

Capítulo 3

Esquema de comportamiento propuesto

En este capítulo se presenta el modelo de comportamiento planteado para el enjambre de robots, el cual está basado en modelos de movimiento de grupos de animales. Se muestran las limitaciones de percepción de cada robot, con el fin de considerarlas al momento de desarrollar las reglas de comportamiento de interacción de los robots con el ambiente y con otros robots.

3.1 Modelos inspirados en el comportamiento de animales sociales

El estudio del comportamiento de grupos de animales se ha llevado a cabo desde hace muchos años por parte de científicos en el campo de la biología [49, 50]. Diferentes modelos matemáticos han sido propuestos para imitar o replicar el movimiento colectivo de los seres sociales [39].

En 1987 Reynolds propone un modelo distribuido de comportamiento para simular de manera realista el movimiento de una bandada de aves o peces en animaciones gráficas [51]. En lugar de programar la trayectoria de cada ave o pez, cada

individuo simulado contaba con un conjunto de comportamientos simples y el movimiento grupal es el resultado de las interacciones entre los individuos. Establece tres comportamientos que permiten simular un movimiento grupal:

1. Evitar colisiones con compañeros cercanos.
2. Tratar de igualar la velocidad de compañeros cercanos.
3. Tratar de permanecer cerca de compañeros cercanos.

Siguiendo el enfoque de Reynolds, un modelo biológicamente más realista fue propuesto por Couzin et al. para estudiar la dinámica espacial de bancos de peces y bandadas de aves [1, 52]. El modelo de Couzin se basa en tres simples reglas de comportamiento: (1) Los individuos tratan de mantener una distancia mínima con otros en todo momento, (2) si los individuos no están evadiendo, entonces tienden a ser atraídos a otros y (3) alinear su dirección con la de sus vecinos. El rango de percepción de cada individuo se divide en tres zonas esféricas que no se traslapan y que determinan la aplicación de las reglas del modelo:

- Zona de repulsión (zor).
- Zona de orientación (zoo).
- Zona de atracción (zoa).

Modificando el ancho de estas tres zonas, el comportamiento colectivo del sistema cambia entre 4 diferentes movimientos colectivos que se observan en la Figura 3.1:

- Enjambre: un conjunto con cohesión pero con un bajo nivel de polarización.
- Toroide: los individuos se mantienen rotando alrededor de un núcleo vacío.

- Grupo paralelo dinámico: el grupo presenta alta polarización y es mucho más móvil.
- Grupo altamente paralelo: el grupo se organiza en un arreglo altamente alineado con movimiento rectilíneo.

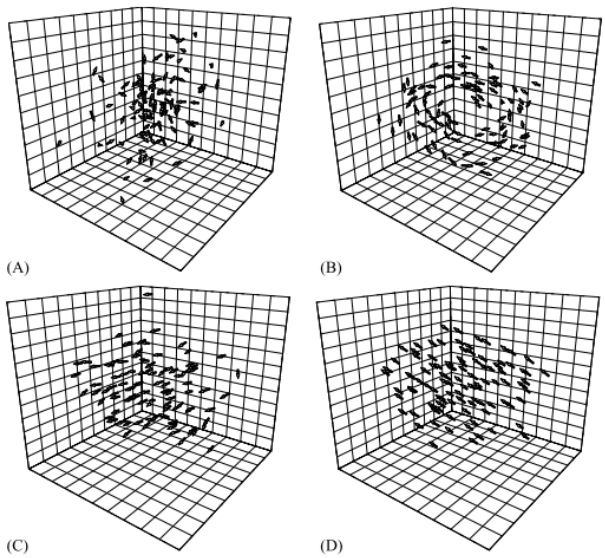


Figura 3.1: Comportamientos colectivos del modelo de Couzin. (A) Enjambre. (B) Toroide. (C) Grupo paralelo dinámico. (D) Grupo altamente paralelo. De Couzin et al. [1]

3.2 Descripción de los miembros del enjambre

Es importante considerar el tipo de plataforma robótica a utilizar al momento de diseñar las reglas de comportamiento para un enjambre de robots. Las reglas de comportamiento deben adaptarse para operar con la información obtenida de los sensores disponibles y calcular movimientos que puedan ser ejecutados por los actuadores del robot.

Se propone el uso de robots aéreos tipo cuadrirrotor como el presentado en la Figura A.1. Este prototipo es un cuadrirrotor pequeño de 36 cm de diámetro,

diseñado con la finalidad de volar al interior de edificios u otras estructuras. Más detalles pueden consultarse en el Apéndice A.

Figura 3.2: Prototipo de cuadrirrotor Starling

Considerando las limitaciones de procesamiento computacional, tamaño y capacidad de carga del cuadrirrotor y que debe operar al interior de edificios, cada robot estará equipado con sensores para medir la distancia hacia otros robots u obstáculos y un módulo de comunicación inalámbrica de corto alcance para transmitir su orientación a los robots cercanos. Esto permitirá la localización relativa de otros robots basada en información local.

Este tipo de sensores cuentan con un ángulo de visión de aproximadamente 25 grados, lo que ocasiona que la capacidad de percepción quede limitada en función de la cantidad de sensores que se monten en el robot.

Otra limitación es la obstrucción de los sensores. Cuando un sensor detecta algo frente a el, no será capaz de detectar más objetos porque su visión queda obstruida por el objeto más cercano. Por eso, este modelo sólo considera el robot más cercano en cada sector de visión de los sensores como en la Figura 3.3.

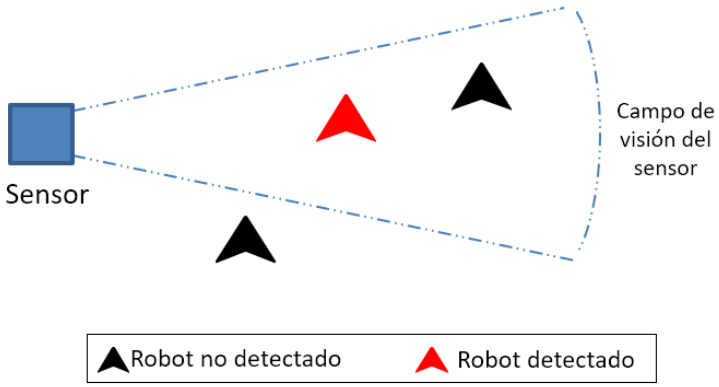


Figura 3.3: Representación del campo de visión de un sensor

3.3 Modelo de percepción en 3 dimensiones

Se considera un arreglo de sensores como el presentado por Roberts [53] para que cada robot pueda detectar la distancia relativa a otros robots en un espacio tridimensional. En la Figura 3.4 se muestra un ejemplo de la distribución de los sensores representando al robot como una esfera. El espacio angular entre sensores es de 45 grados, por lo que en cada eje tenemos 8 sensores y un total de sensores $K = 26$.

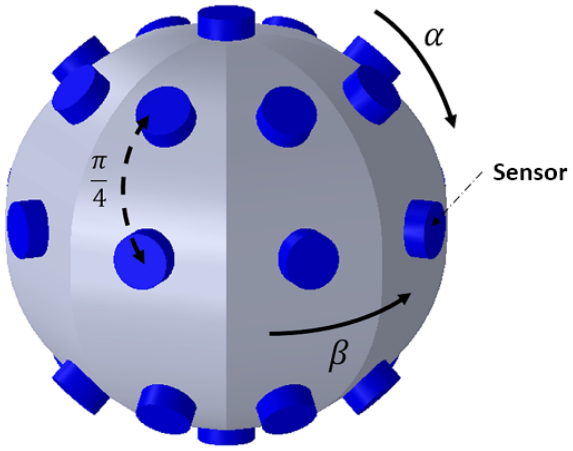


Figura 3.4: Representación de la distribución de los sensores en tres dimensiones

Cada uno de estos sensores apunta en una dirección, la cual es representada

por un vector unitario $\hat{s}_k$ dado por:

$$\hat{s}_k = [\text{sen}(\alpha)\cos(\beta) \quad \text{sen}(\alpha)\,\text{sen}(\beta) \quad \cos(\alpha)] \tag{3.1}$$

con $\alpha = \{0, \quad \frac{\pi}{4}, \ldots, \pi\}$ y $\beta = \{\frac{\pi}{8}, \quad \frac{3\pi}{8}, \ldots, \frac{15\pi}{8}\}$.

La matriz S_i contiene los vectores unitarios de dirección de todos los sensores:

$$S_i = \begin{bmatrix} \hat{s}_1 \\ \hat{s}_2 \\ \vdots \\ \hat{s}_K \end{bmatrix} \tag{3.2}$$

3.4 Modelo de percepción en 2 dimensiones

El modelo de percepción en 3 dimensiones es la forma más cercana al modelo original de Couzin, donde los miembros del enjambre pueden percibir y moverse en 3 dimensiones. Sin embargo, puede ser difícil la colocación de los sensores para la detección de otros robots en 3 dimensiones por la cantidad de sensores necesarios. Además, pueden existir perturbaciones generadas por el empuje de los rotores al volar uno sobre otro. Por estas razones en muchas de las ocasiones se prefiere que un enjambre de robots vuele a la misma altura para evitar problemas en la detección de vecinos. A continuación se modifica el modelo para considerar que los robots vuelen a la misma altura, con lo que se restringe el modelo a 2 dimensiones.

Para este caso consideramos la colocación de los sensores en un plano horizontal alrededor del robot. Por lo tanto, dividimos el área alrededor de cada robot en k sectores correspondientes con el número de sensores. En este caso se consideran 8 sensores como se observa en la Figura 3.5.

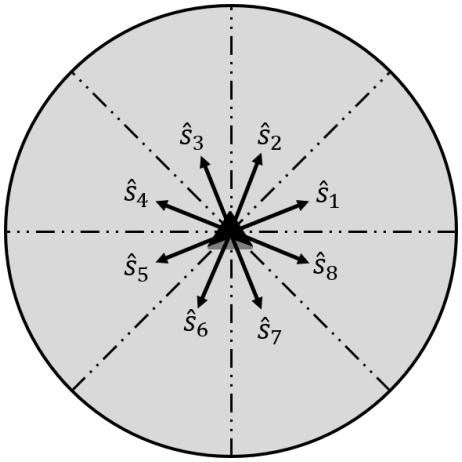


Figura 3.5: Representación de las zonas de percepción alrededor de cada robot.

Además se utiliza un vector $S_{i(t)} = [\hat{s}_1, \hat{s}_2, ..., \hat{s}_k]^T$ que contiene los vectores unitarios de dirección de los k sensores, con $\hat{s}_k = (\cos((2k-1)\pi/8),\ \text{sen}((2k-1)\pi/8))$

3.5 Reglas de comportamiento

El modelo de comportamiento de la presente tesis está basado en el modelo de Couzin. Se establecen tres zonas de comportamiento alrededor de cada individuo determinadas por el ancho Δ. Utilizamos una cuarta zona esférica denominada zona de influencia (zoi), la cual representa una forma de guiar al enjambre hacia un objetivo deseado [54]. Esta zona de influencia se puede superponer con las otras zonas de comportamiento como se muestra en la Figura 3.6.

Para darle un sentido más realista, se considera que cada sensor solo puede detectar al robot vecino más cercano, ya que este bloquea el campo de visión del sensor impidiendole detectar más robots. Esta es una de las principales diferencias con el modelo propuesto por Couzin donde todos los vecinos dentro de determinada zona de comportamiento son considerados.

Nuestro modelo considera el uso de las zonas de comortamiento (ZOR, ZOO, ZOA y ZOI) sobrepuestas con los sectores de los sensores de distancia, para discre-

tizar el espacio alrededor de cada robot.

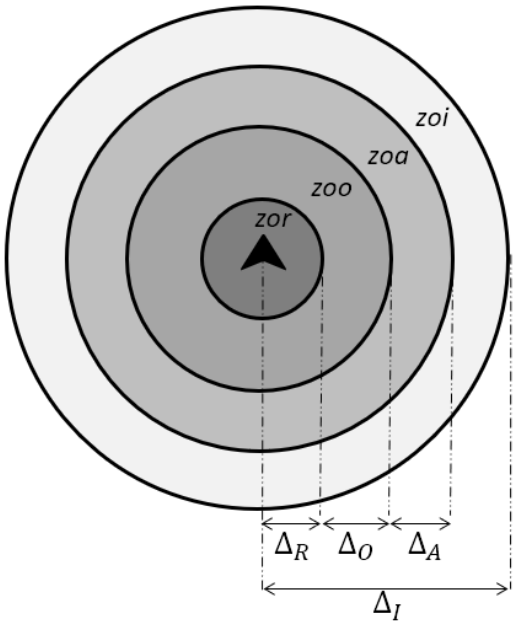


Figura 3.6: Zonas de comportamiento alrededor de cada individuo

Las reglas de comportamiento actúan en función de la cantidad de vecinos detectados en las zonas de comportamiento alrededor de cada robot. Podemos dividir los comportamientos en dos tipos: los originados por las interacciones entre robots y los originados por la interacción con el entorno.

Cada miembro del enjambre puede ejecutar diferentes comportamientos individuales al interactuar con sus vecinos. Estos comportamientos son repulsión, orientación y atracción. Además, al interactuar con su entorno puede ejecutar los comportamientos de evitar obstáculos, ser atraídos hacia objetivos y explorar.

3.5.1 Comportamiento de Repulsión

Si un robot i detecta al menos un vecino j dentro de su zona de repulsión ($n_R \geq 1$) como se ve en la Figura 3.7, entonces calculará un vector con dirección opuesta a la posición del robot j para alejarse y mantener una distancia mínima

entre ellos. Esta dirección deseada de repulsión se calcula con:

$$ddR_{(t+\tau)} = \frac{-QR_{i(t)} \cdot S_{i(t)}}{\|QR_{i(t)} \cdot S_{i(t)}\|} \tag{3.3}$$

donde $QR_i = \begin{bmatrix} qR_1 & qR_2 & \dots & qR_K \end{bmatrix}$ es el vector que guarda la distancia al robot más cercano d_{ij} en la zona de repulsión de cada sector de visión k. El total de vecinos detectados en el sector k se denota por n_k y el valor de qR_k está dado por:

$$qR_k = \begin{cases} 0, & \text{si } n_k = 0 \\ \min\limits_{j \in [1, n_k]} d_{ij}, & \text{si } n_k \geq 1 \end{cases} \tag{3.4}$$

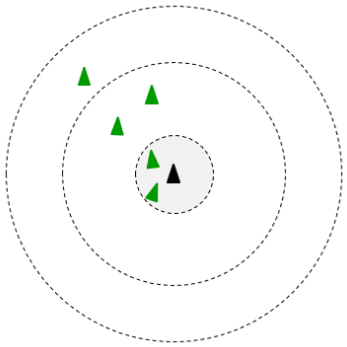


Figura 3.7: Comportamiento de repulsión

3.5.2 Comportamiento de Orientación

Si no se detecta a ningún robot vecino en la zona de repulsión y se detectan vecinos en la zona de orientación ($n_O \geq 1$) como en la Figura 3.8, entonces se forma un vector unitario que representa la dirección del robot vecino j:

$$\hat{o}_j = \begin{bmatrix} \cos(\psi_j) & \sin(\psi_j) \end{bmatrix} \tag{3.5}$$

donde ψ_j es el ángulo de la dirección del vecino j

El robot tratará de cambiar su dirección para que corresponda con la dirección promedio de los vecinos detectados. La dirección deseada de orientación se calcula con:

$$ddO_{(t+\tau)} = \sum_{j=1}^{n_O} \frac{\hat{o}_j}{\|\hat{o}_j\|} \tag{3.6}$$

donde n_O es el número de robots vecinos en la zona de orientación.

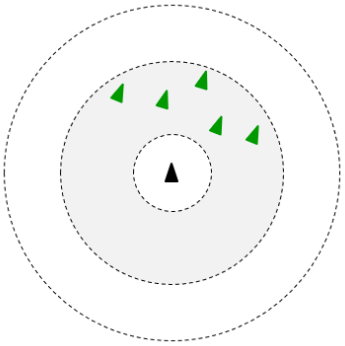


Figura 3.8: Comportamiento de orientación

3.5.3 Comportamiento de Atracción

En la Figura 3.9 cuando sólo se detectan robots vecinos en la zona de atracción ($n_A \geq 1$), se crea un vector donde $QA_i = \begin{bmatrix} qA_1 & qA_2 & \dots & qA_K \end{bmatrix}$ que guarda la distancia al robot más cercano d_{ij} en la zona de atracción de cada sector de visión k. El total de vecinos detectados en el sector k se denota por n_k y el valor de qA_k está dado por:

$$qA_k = \begin{cases} 0, & \text{si } n_k = 0 \\ \min\limits_{j \in [1, n_k]} d_{ij}, & \text{si } n_k \geq 1 \end{cases} \tag{3.7}$$

Entonces, la dirección deseada de atracción se calcula con:

$$ddA_{(t+\tau)} = \frac{QA_{i(t)} \cdot S_{i(t)}}{\|QA_{i(t)} \cdot S_{i(t)}\|} \tag{3.8}$$

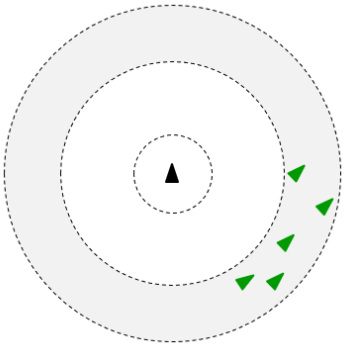


Figura 3.9: Comportamiento de atracción

3.5.4 Comportamiento de Orientación-Atracción

Los comportamientos de orientación y atracción pueden ocurrir al mismo tiempo cuenado se detectan vecinos en ambas zonas, como se observa en la Figura 3.10

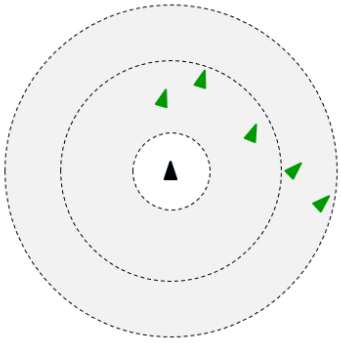


Figura 3.10: Comportamiento de orientación-atracción

3.5.5 Comportamiento de Explorar

Cuando no se detecta ningún robot vecino en las zonas de repulsión, orientación y atracción (Figura 3.11), se asigna una dirección aleatoria al robot, con el fin de que explore el área en que se encuentra. Esta dirección deseada de exploración está dada por:

$$ddE_{(t+\tau)} = \begin{bmatrix} \cos(\psi_{rand}) & \sin(\psi_{rand}) \end{bmatrix} \tag{3.9}$$

donde ψ_{rand} es un ángulo aleatorio entre 0 y 2π. El robot mantendrá esta dirección hasta que se encuentre con algún obstáculo, en cuyo caso se asignará otro ángulo aleatorio.

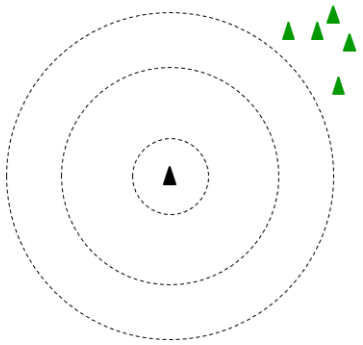


Figura 3.11: Comportamiento de explorar

3.5.6 Evasión de Obstáculos

Para que cada robot tenga la capacidad de evitar obstáculos, se designa un vector $Qobs_i = \begin{bmatrix} qobs_1 & qobs_2 & \dots & qobs_K \end{bmatrix}$ donde $qobs_k$ corresponde al sector de visión del sensor k del robot y su valor corresponde a la distancia al obstáculo $d_{i,obs}$ dentro del rango de detección:

$$qobs_k = \begin{cases} 0, & \text{no se detecta obstáculo} \\ d_{i,obs}, & \text{si se detecta obstáculo} \end{cases} \tag{3.10}$$

El vector *ddobs* correspondiente a la dirección deseada para esquivar el obstáculo está dado por:

$$ddobs_{(t+\tau)} = -\frac{Qobs_{i(t)} \cdot S_{i(t)}}{\|Qobs_{i(t)} \cdot S_{i(t)}\|} + dd_{i(t)} \tag{3.11}$$

donde $dd_{i(t)} = \begin{bmatrix} \cos(\psi_i) & \sin(\psi_i) \end{bmatrix}$ es la dirección actual del robot i y se agrega para que el robot pueda esquivar el obstáculo sin alejarse completamente del rumbo que llevaba.

3.5.7 Influencia

La zona de influencia puede ser usada para guiar al enjambre a un objetivo deseado, por medio de un tipo de estímulo como una fuente de luz [54]. Otra forma de influir en un enjambre de robots es similar a lo que ocurre en los comportamientos de depredador - presa [55]. Para ciertas tareas llamaremos punto de influencia al objetivo que el enjambre debe seguir. Cuando el punto de influencia se encuentra dentro del rango de detección de un robot, se calcula un vector que apunta en la dirección de este punto:

$$dd\text{inf}_{(t+\tau)} = \frac{Q\text{inf}_{i(t)} \cdot S_{i(t)}}{\|Q\text{inf}_{i(t)} \cdot S_{i(t)}\|} \tag{3.12}$$

donde $Q\text{inf}_i = \begin{bmatrix} q\text{inf}_1 & q\text{inf}_2 & \dots & q\text{inf}_K \end{bmatrix}$ es el vector que guarda la distancia $d_{i,inf}$ en el sector de visión k donde se detecta el punto de influencia:

$$q\text{inf}_k = \begin{cases} 0, & \text{no se detecta punto de influencia} \\ d_{i,\text{inf}}, & \text{si se detecta punto de influencia} \end{cases} \tag{3.13}$$

3.5.8 Dirección Deseada Final

Los comportamientos de repulsión y evasión de obstáculos tienen prioridad sobre los demás. Por lo tanto, si al menos un robot vecino se encuentra en la zona de repulsión, la dirección deseada será:

$$dd_i = w_1\, ddR + w_2\, ddobs \tag{3.14}$$

donde los pesos w_1 y w_2 se usan para dar mayor prioridad a algún comportamiento. Cuando no se detectan vecinos en la zona de repulsión, la dirección deseada final está dada por:

$$dd_i = ddO + ddA + ddinf + ddobs \tag{3.15}$$

Capítulo 4

Resultados

En este capítulo se describe la plataforma de simulación para realizar los experimentos con las reglas de comportamiento planteadas en el capítulo anterior. Se presentan las métricas para evaluar el desempeño del enjambre en los experimentos y por último se presentan los resultados obtenidos al simular experimentos en dos y en tres dimensiones, utilizando diversos conjuntos de parámetros.

4.1 Plataforma de Simulación

Ante la dificultad de realizar experimentos con un gran número de robots físicos, se implementó una plataforma que nos permite simular el enjambre de robots. De esta forma se pueden simular un gran número de experimentos con un número relativamente grande de robots.

Esta plataforma de simulación servirá para evaluar la operación del enjambre con diferentes parámetros de comportamiento y bajo distintas circunstancias. El simulador está implementado en la versión 6.0 de *Scilab*, el cual es un software de código abierto para cálculo numérico. En la Figura 4.1 se muestra un diagrama con los componentes necesarios para simular cada miembro del enjambre. Para cada robot, en primer lugar se determina la distancia a los demás robots y a los obstáculos

o puntos de influencia si es que existen. Con estas distancias se clasifican los vecinos que pueden ser detectados en función de la percepción de los sensores. Esto constituye la localización relativa y el resultado se usa en las reglas de comportamiento para determinar la velocidad y dirección deseada del robot. Estas variables se utilizan en el control de velocidad para calcular los ángulos deseados y la altitud deseada, que se usan como entradas en el control de posición angular y en el control de altitud. Por último se calculan las velocidades angulares de los rotores y se introducen en el modelo dinámico del cuadrirrotor para obtener nuevas posiciones cartesianas y angulares del cuadrirrotor.

Para representar posibles errores en las mediciones de los sensores, se agrega ruido Gaussiano a la velocidad angular del cuadrirrotor, su altitud y también en las distancias medidas hacia los vecinos del enjambre, obstáculos y puntos de influencia.

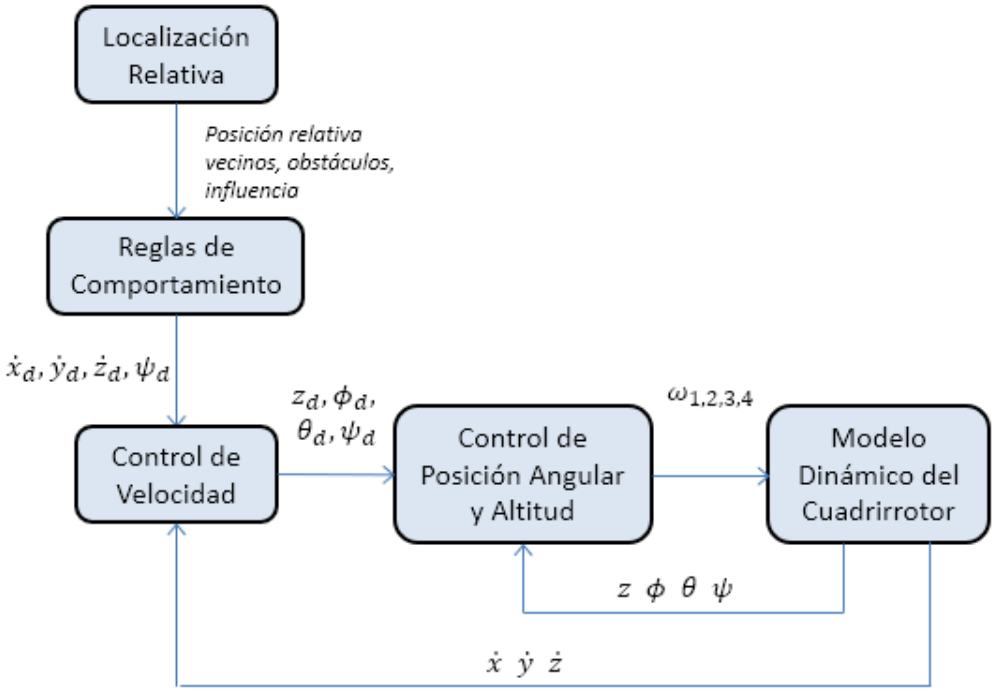


Figura 4.1: Componentes de cada robot en el simulador.

Entrada: $\Delta_R, \Delta_O, \Delta_A, \Delta_I$

Iniciar posiciones y velocidades lineales y angulares;

para *s = 1 : T* **hacer**

  para *i = 1 : N* **hacer**

    Calcular distancia y ángulo con punto de influencia;

    Actualizar vector $Q\text{inf}_i$;

    Calcular distancia y ángulo con obstáculos;

    Actualizar vector $Q\text{obs}_i$;

    para *j = 1 : N, j* $\neq$ *i* **hacer**

      Calcular distancia y ángulo con cuadrirrotor j ;

      Actualizar vector QR_i ;

      Actualizar vector QA_i ;

      Actualizar vector $\hat{o_j}$;

    fin

    Calcular vector $dd\text{inf}_i$;

    Calcular vector $dd\text{obs}_i$;

    Calcular vector ddR_i ;

    Calcular vector ddO_i ;

    Calcular vector ddA_i ;

    Calcular dirección deseada final dd_i ;

    Usar control de velocidad ;

    Usar control de posición angular ;

    Usar control de altitud ;

  fin

fin

Algoritmo 1: Simulador de enjambre de cuadrirrotores

4.2 Métricas de Evaluación

Para evaluar el comportamiento de un enjambre existen diferentes mediciones que pueden ser utilizadas. A continuación se definen la utilizadas en esta tesis.

4.2.1 Unión del enjambre

La teoría de grafos es una herramienta muy útil para analizar como interactúan los elementos de un sistema [56]. Es posible representar el enjambre de robots como un grafo no dirigido $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, con el conjunto de vértices o nodos $\mathcal{V}$ representando a los robots y el conjunto de enlaces o aristas $\mathcal{E}$ representando los pares de robots (i, j) tal que el robot i percibe al robot j.

La matriz de adyacencia $A \in \mathbb{R}^{N \times N}$ es una matriz cuadrada simétrica tal que $A_{ij} = 1$ si el robot i puede percibir al robot j y $A_{ij} = 0$ si el robot i no puede percibir al robot. Además, los elementos de la diagonal son cero, $A_{ii} = 0$.

$$A_{ij} = \begin{cases} 1, & \text{si } (i,j) \in \mathcal{E} \\ 0, & \text{de otra forma} \end{cases} \tag{4.1}$$

Un componente es un subconjunto de nodos de un grafo, en el que existe un camino entre dos vértices cualesquiera y está conectado a ningún vértice adicional del grafo. En el caso del enjambre de robots, un componente es un subgrupo de robots que no interactúa con todo el enjambre. El número de componentes del grafo equivale a la multiplicidad de 0 como valor propio de la matriz laplaciana $\mathcal{L} \in \mathbb{R}^{n \times n}$, la cual es una matriz cuadrada simétrica definida como [57]:

$$\mathcal{L} = diag(\deg_i) - A \tag{4.2}$$

$$\deg_i = \sum_{j=1}^{N} A_{ij} \tag{4.3}$$

Este número de componentes n_{cc} puede usarse para determinar que tan unidos están los miembros del enjambre [58]:

$$\Phi_{\text{unión}}(k) = 1 - \frac{n_{cc} - 1}{N - 1} \tag{4.4}$$

donde el valor máximo para esta métrica es 1 cuando existe sólo un grupo de robots, mientras que el valor mínimo es cero cuando todos los robots están separados.

4.2.2 Número de colisiones

Uno de los comportamientos individuales implementados en el presente modelo es la repulsión. Con el fin de evaluar el desempeño de este comportamiento es necesario analizar el número de colisiones que ocurren en cada experimento. En cada instante de tiempo las colisiones detectadas entre robots son almacenadas en una matriz $B_{(t)}$:

$$B = \begin{bmatrix} 0 & b_{12} & \dots & b_{1N} \\ b_{21} & 0 & \dots & b_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ b_{N1} & b_{N2} & \dots & 0 \end{bmatrix} \tag{4.5}$$

donde el valor de b_{ij} está determinado por:

$$b_{ij} = \begin{cases} 1, & \text{si } d_{ij} \leq 0,\ i \neq j \\ 0, & \text{de otra forma} \end{cases} \tag{4.6}$$

El total de colisiones para cada instante de tiempo se obtiene con:

$$n_{\text{col}}(k) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} b_{ij} \tag{4.7}$$

4.2.3 Polarización

El movimiento colectivo emerge por medio de una transición de una fase desordenada a una ordenada. Una forma de caracterizar esta transición es por medio de un parámetro de orden que se relaciona con el grado de simetría de una fase del sistema [59]. La polarización del grupo mostrada en [1], aumenta a medida que la alineación entre los miembros del grupo aumenta y se mueven en la misma dirección. Se define como:

$$\rho(k) = \frac{1}{N}\left\|\sum_{i=1}^{N} \hat{o}_i\right\| \tag{4.8}$$

donde $\hat{o}_i$ es el vector unitario de dirección del robot i y $[0 \leq \rho \leq 1]$.

4.2.4 Distancia promedio al centro del enjambre

La métrica de unión nos permite determinar si el enjambre está formado por un sólo grupo o si existen subgrupos que se mueven independientes entre sí. Cuando existe un sólo grupo en el enjambre, la distancia promedio al centro del enjambre nos permite obtener una medida de que tan cerca están los miembros del enjambre entre sí. Podemos calcular esta distancia promedio al centro del enjambre con:

$$\delta_{\text{cen}}(k) = \frac{1}{N}\sum_{i=1}^{N} d_E(c_i - c_{\text{grupo}}) \tag{4.9}$$

donde $d_E(c_i - c_{\text{grupo}})$ es la distancia euclidiana entre la posición del robot i y

la posición del centro del grupo:

$$c_{\text{grupo}} = \frac{1}{N} \sum_{i=1}^{N} c_i \tag{4.10}$$

Las métricas anteriores están definidas para un instante de tiempo determinado $t_k = k\,dt$, $k \in \{1, 2, 3, \ldots T_t\}$. Para evaluar el comportamiento global del experimento durante todo el tiempo de simulación t_{sim} es necesario promediar el valor obtenido en cada instante de tiempo. Por ejemplo, en el caso de la polarización:

$$\rho = \frac{1}{T_t} \sum_{k=1}^{T_t} \rho(k) \tag{4.11}$$

donde el total de muestras de tiempo T_t se obtiene con

$$T_t = \frac{t_{\text{sim}}}{dt} \tag{4.12}$$

siendo t_{sim} el tiempo total de simulación y dt el intervalo de tiempo o paso de integración.

4.3 Experimentos en tres dimensiones

Para evaluar el funcionamiento del modelo de comportamiento se simularon experimentos en tres dimensiones, considerando la discretización del campo de percepción del robot en sectores correspondientes a los sensores. Esto nos permite emular un comportamiento similar al mostrado por aves o peces como describe el modelo propuesto por Couzin et al. [1].

4.3.1 Movimiento colectivo

Como se mencionó en el Capítulo 2, la tarea de movimiento colectivo consiste en el desplazamiento de individuos en una misma dirección. Se utilizó además la zona de influencia para obtener un comportamiento de tipo presa - depredador, en el cual la presa es un punto de influencia que sigue una determinada trayectoria y el enjambre de robots actúa como depredadores siguiendo este punto de influencia. El punto de influencia se mueve a una velocidad de 0.65 m/s, siguiendo una trayectoria circular con un radio de 5 metros, al mismo tiempo que realiza cambios de altitud. En la Figura 4.2 el punto de influencia se muestra como un cuadrirrotor en color rojo y la trayectoria que sigue se muestra como una línea punteada.

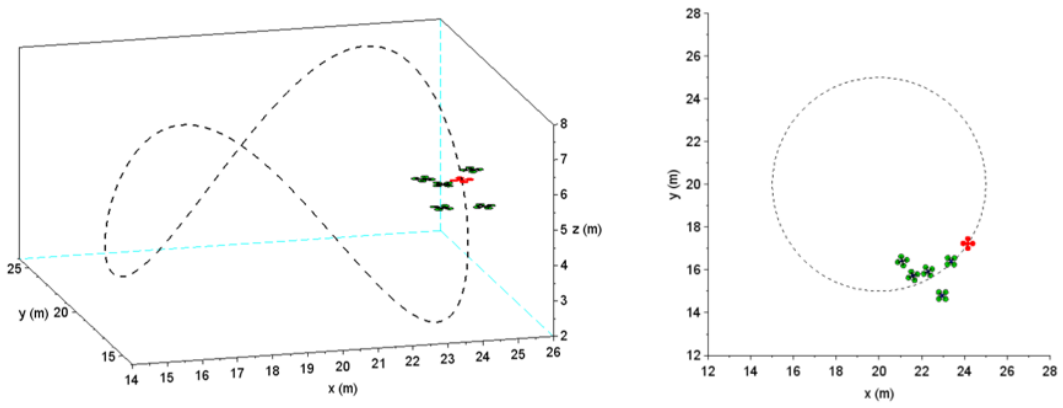


Figura 4.2: Trayectoria seguida por el punto de influencia

Investigamos diferentes combinaciones de parámetros de comportamiento que determinen condiciones en las que el enjambre se mueva como un grupo y al mismo tiempo siga al punto de influencia. De esta forma se puede guiar a todo un grupo hacia una ubicación cuando sólo un individuo conoce la trayectoria que debe seguir. Estos parámetros de comportamiento son el ancho de la zonas de repulsión Δ_R, orientación Δ_O, atracción Δ_A e influencia Δ_I. La Tabla 4.1 muestra todos los parámetros usados en las simulaciones de esta tarea y para cada combinación de parámetros de comportamiento se realizaron 5 simulaciones.

Tabla 4.1: Parámetros de simulación de movimiento en tres dimensiones

Parámetro	Descripción	Valor	Unidades
Δ_R	Zona de repulsión	0.5, 1.0, 1.5	m
Δ_O	Zona de orientación	1, 2, 3, 4, 5	m
Δ_A	Zona de atracción	1, 3, 5	m
Δ_I	Zona de influencia	5, 10	m
N	Número de individuos	5, 10, 20	
t_{max}	tiempo de simulación	90	s
dt	paso de integración	0.01	s
v_{max}	velocidad máxima	1	m/s

En la Figura 4.3 se puede observar el resultado de tres métricas usadas para evaluar al enjambre en la tarea desarrollada, cuando el ancho de la zona de influencia Δ_I es de 5 m. El número de colisiones por robot n_{col}/N es elevado para el enjambre de 20 robots con $\Delta_R = 0.5$ m, pero con valores de Δ_R de 1 o 1.5 m el número de colisiones por robot se vuelve cero en muchos casos. Las gráficas de polarización ρ nos muestran que a mayor número de robots es menor la alineación entre ellos, obteniendo mejores resultados mientras menor sea el valor de Δ_R y mayor sea el valor de Δ_O. En cuanto a la distancia promedio al centro del enjambre δ_{cen}, como es de esperase aumenta al aumentar el número de robots, pero además se ve afectada por el valor de Δ_R.

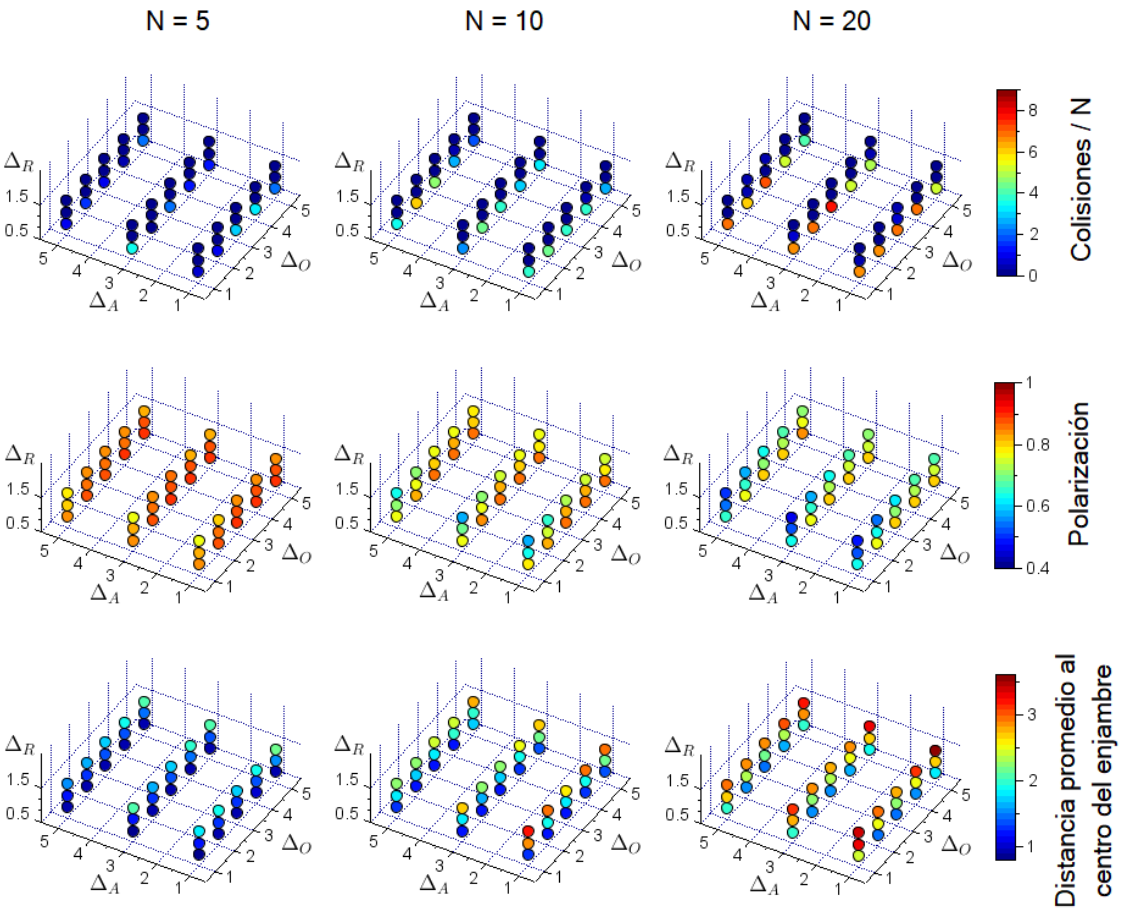


Figura 4.3: Cantidad de colisiones por robot, polarización de grupo y distancia promedio al centro del grupo en el experimento en tres dimensiones. Zona de Influencia $\Delta_I = 5$ m. Cada punto en la gráfica es el promedio de 5 repeticiones.

Como se ve en la Figuras 4.4, 4.5 y 4.6 es posible obtener un enjambre de robots que se mueva en grupo, al mismo tiempo que sigue un punto de influencia, para diferentes cantidades de robots (5, 10 y 20 respectivamente).

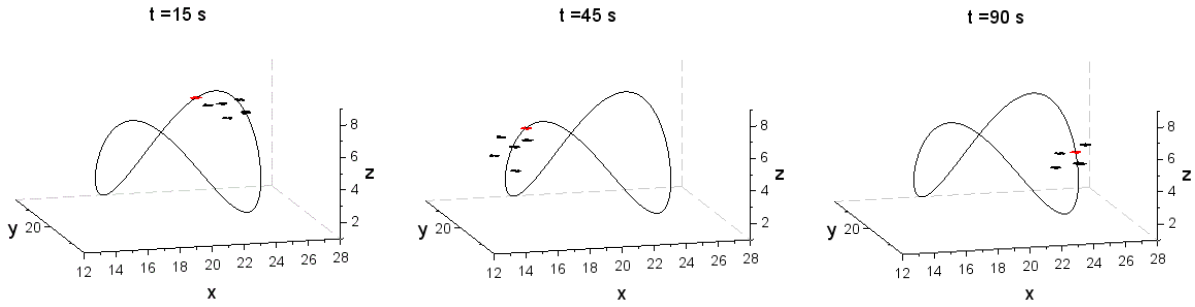


Figura 4.4: Secuencia de comportamiento del experimento en tres dimensiones con N = 5, $\Delta_R = 1$ m, $\Delta_O = 2$ m, $\Delta_A = 1$ m y $\Delta_I = 5$ m

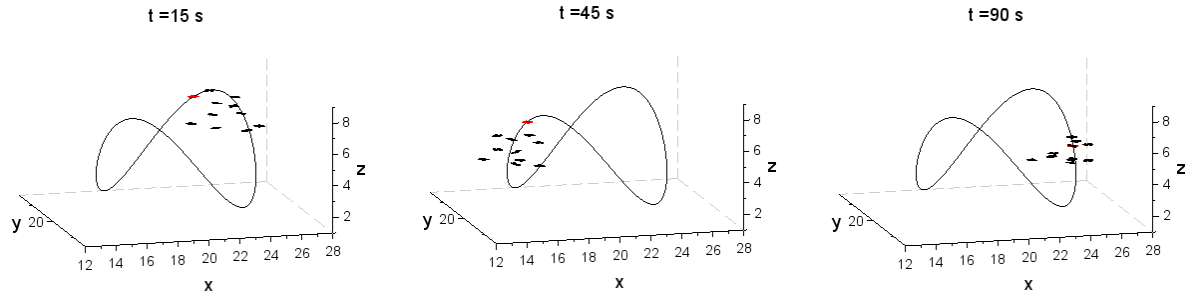


Figura 4.5: Secuencia de comportamiento del experimento en tres dimensiones con N = 10, $\Delta_R = 1$ m, $\Delta_O = 3$ m, $\Delta_A = 1$ m y $\Delta_I = 5$ m

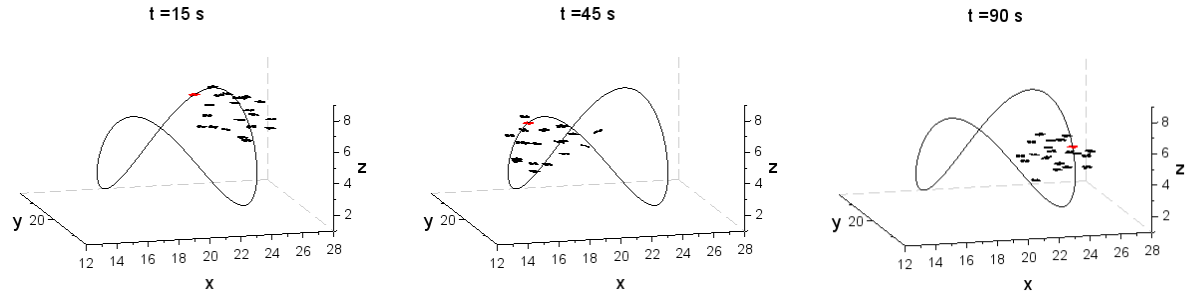


Figura 4.6: Secuencia de comportamiento del experimento en tres dimensiones con N = 20, $\Delta_R = 1$ m, $\Delta_O = 3$ m, $\Delta_A = 5$ m y $\Delta_I = 5$ m

4.3.2 Movimiento colectivo y obstáculo

Para comprobar que el comportamiento es robusto ante cambios en el entorno, se agrega un obstáculo con forma de cilindro que tiene un diámetro de 2 metros. La trayectoria del punto de influencia es igual que en el experimento anterior. Se puede apreciar en la Figura 4.7 que la trayectoria del punto de influencia pasa a través del obstáculo, sin embargo el enjambre de robots debe evitarlo.

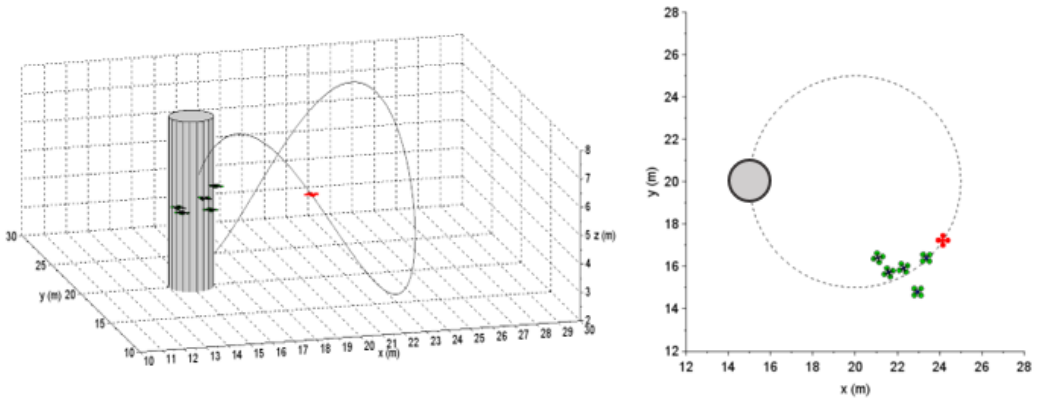


Figura 4.7: Trayectoria seguida por el punto de influencia y ubicación del obstáculo

Los parámetros empleados en este experimento son los que se observan en la Tabla 4.2 y de igual forma que en el experimento anterior cada combinación de parámetros es simulada 5 veces.

Tabla 4.2: Parámetros de simulación de movimiento en tres dimensiones con obstáculo.

Parámetro	Descripción	Valor	Unidades
Δ_R	Zona de repulsión	0.5, 1.0, 1.5	m
Δ_O	Zona de orientación	1, 3, 5	m
Δ_A	Zona de atracción	1, 3, 5	m
Δ_I	Zona de influencia	5, 10	m
N	Número de individuos	5, 10, 20	
t_{max}	tiempo de simulación	90	s
dt	paso de integración	0.01	s
v_{max}	velocidad máxima	1	m/s

Como se puede apreciar en la Figura 4.8 el número de colisiones por robot disminuye considerablemente para valores de la zona de repulsión de 1 y 1.5 m, lo cual concuerda con lo observado en el experimento sin obstáculo. La polarización aumenta mientras mayor sea la zona de orientación y también mientras menor sea la zona de repulsión. A medida que aumentamos el número de robots aumenta el volumen total del enjambre, como se refleja en la distancia promedio al centro del enjambre. Este aumento en el volumen del enjambre ocasiona que los robots en los extremos consideren menos vecinos al momento de elegir la dirección a seguir.

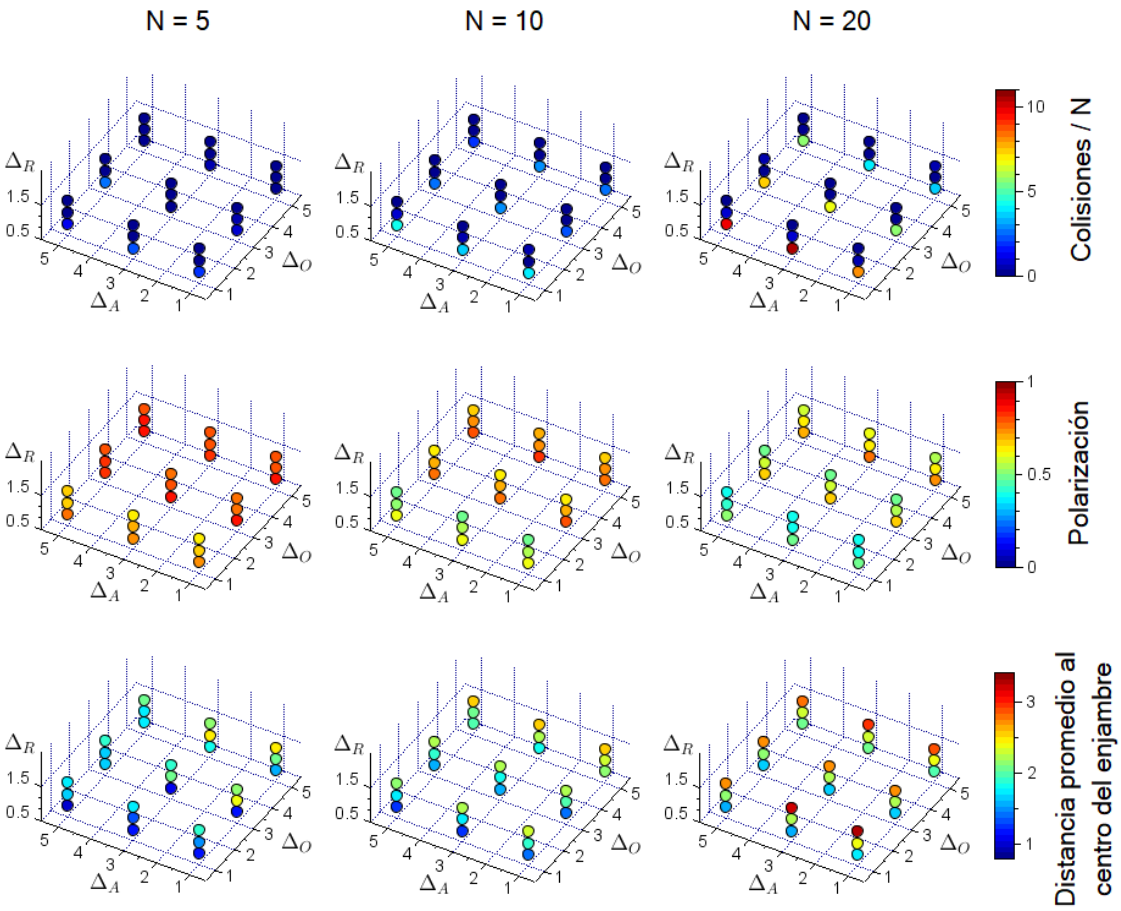


Figura 4.8: Cantidad de colisiones por robot, polarización de grupo y distancia promedio al centro del grupo en el experimento en tres dimensiones con obstáculo. Zona de Influencia $\Delta_I = 5$ m. Cada punto en la gráfica es el promedio de 5 repeticiones.

En la Figura 4.9 se observa un ejemplo de las trayectorias dibujadas por un enjambre de 5 robots. A pesar de que la trayectoria de el punto de influencia pasa a través del obstáculo, los robots se alejan para evitar el obstáculo y vuelven a acercarse al punto de influencia.

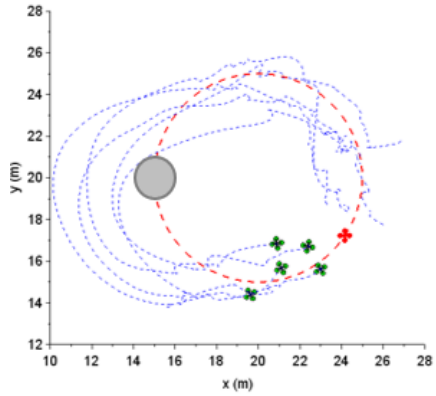


Figura 4.9: Ejemplo de las trayectorias seguidas por el enjambre para evitar el obstáculo. Enjambre de 5 robots con $\Delta_R = 1.0$, $\Delta_O = 3.0$, $\Delta_A = 3.0$ y $\Delta_I = 5.0$

A continuación se muestran secuencias de este experimento con diferentes parámetros de comportamiento. Las Figuras 4.10, 4.11 y 4.12 corresponden a enjambres de 5, 10 y 20 robots respectivamente, con zonas de influencia $\Delta_I = 10$ m. Para este experimento, las combinaciones de parámetros con $\Delta_I = 10$ m tuvieron un mayor porcentaje de éxito al seguir la trayectoria del punto del influencia. En el caso de las combinaciones con $\Delta_I = 5$ m, en ciertas ocasiones algunos robots o todo el enjambre dejan de seguir el punto de influencia.

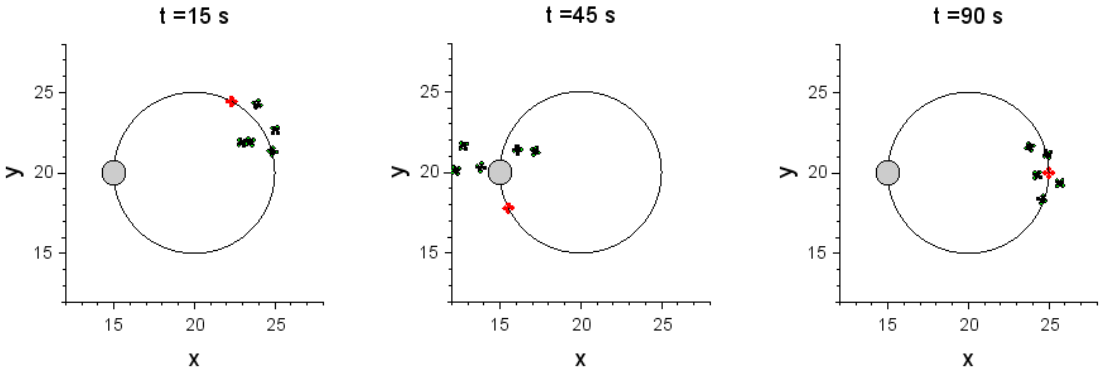


Figura 4.10: Secuencia de comportamiento del experimento en tres dimensiones con N = 5, $\Delta_R = 1$ m, $\Delta_O = 3$ m, $\Delta_A = 1$ m y $\Delta_I = 10$ m

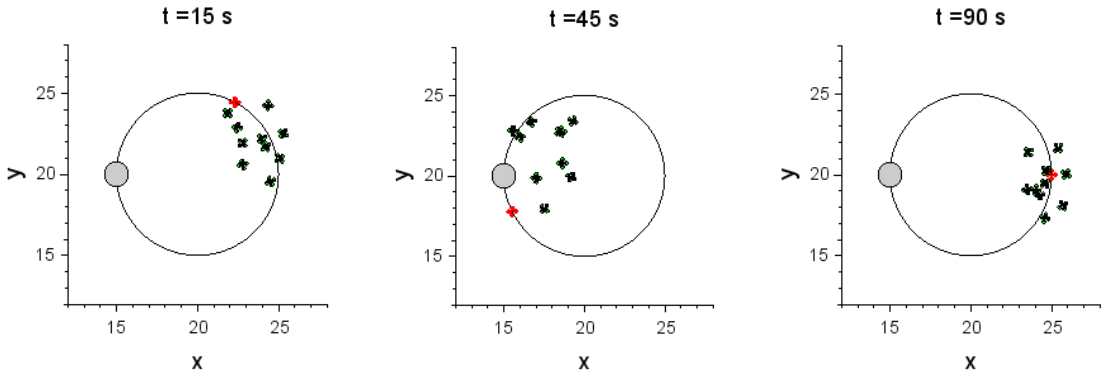


Figura 4.11: Secuencia de comportamiento del experimento en tres dimensiones con N = 10, $\Delta_R = 1$ m, $\Delta_O = 3$ m, $\Delta_A = 1$ m y $\Delta_I = 10$ m

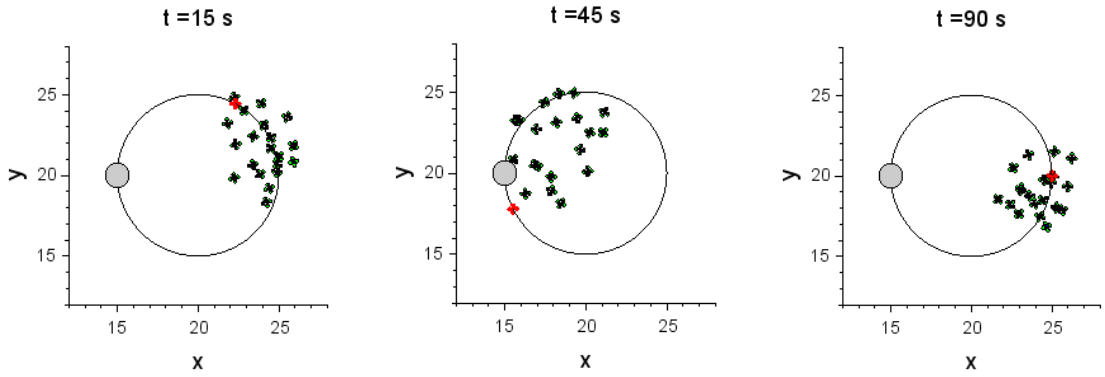


Figura 4.12: Secuencia de comportamiento del experimento en tres dimensiones con N = 20, $\Delta_R = 1$ m, $\Delta_O = 5$ m, $\Delta_A = 5$ m y $\Delta_I = 10$ m

Es importante considerar la zona de influencia al momento de diseñar un comportamiento como el mostrado en este experimento. Si la zona de influencia es muy pequeña y la polarización del enjambre es pobre, entonces el punto de influencia terminaría alejándose hasta salir del rango de percepción de los robots. Como se observa en la Figura 4.12, cuando el enjambre es muy grande algunos robots tienden a quedarse muy atrás; si la zona de influencia y atracción fueran muy pequeñas, entonces estos robots no podrían detectar el punto de influencia ni a otros robots, por lo que terminarían separándose del grupo.

4.4 Experimentos en dos dimensiones

La detección de robots vecinos puede ser complicada en un espacio tridimensional, debido a que se requiere una mayor cantidad de sensores, lo que se dificulta en robots pequeños por cuestión de espacio y capacidad de carga. Para evitar estas complicaciones en futuras implementaciones del modelo, en este caso la altura deseada para cada robot se fija en un metro, lo que restringe el movimiento del enjambre a un plano.

4.4.1 Navegación con obstáculos

Para evaluar la capacidad del enjambre para responder a cambios en su entorno, en esta tarea los robots deben desplazarse de un lado al otro de la arena y evitar dos obstáculos. Como se muestra en la Figura 4.13, los robots inician con orientaciones y posiciones aleatorias dentro del rectángulo en el lado izquierdo y deben desplazarse hasta la línea verde en el lado derecho de la arena, evitando dos obstáculos de 1.5 m de diámetro. Para que los robots se muevan en la dirección deseada se simula un punto de influencia en el lado derecho del área de prueba y el parámetro del ancho de la zona de influencia se fija en un valor alto para que puedan detectar su objetivo en todo momento. El resto de parámetros de la simulación se muestran en la Tabla 4.3.

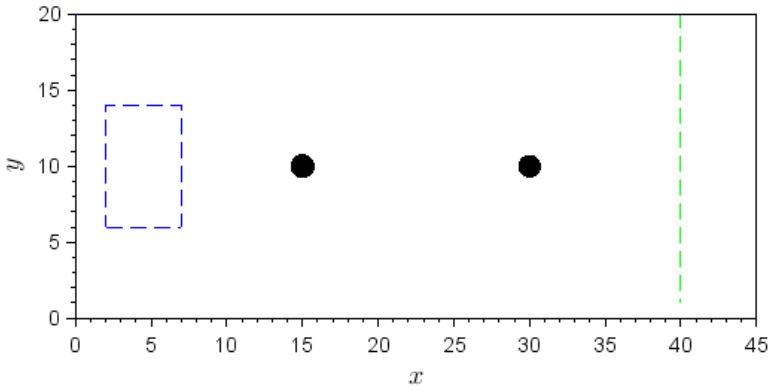


Figura 4.13: Arena para experimento de navegación con obstáculos.

Tabla 4.3: Parámetros de simulación de navegación con obstáculos.

Parámetro	Descripción	Valor	Unidades
Δ_R	Zona de repulsión	0.5 - 2.5	m
Δ_O	Zona de orientación	1 - 5	m
Δ_A	Zona de atracción	1 - 5	m
N	Número de individuos	5 - 20	
t_{max}	tiempo de simulación	300	s
dt	paso de integración	0.05	s
z_d	altitud deseada	1	m
v_{max}	velocidad máxima	0.5	m/s

En la Figura 4.14 se presentan las colisiones por robots para todas las combinaciones de parámetros. Como en los experimentos anteriores, el número de colisiones disminuye considerablemente con zonas de repulsión a partir de 1 m. En cuanto al tiempo necesario para que el enjambre complete el recorrido se muestra en la Figura 4.15 que valores altos de repulsión logran menores tiempos.

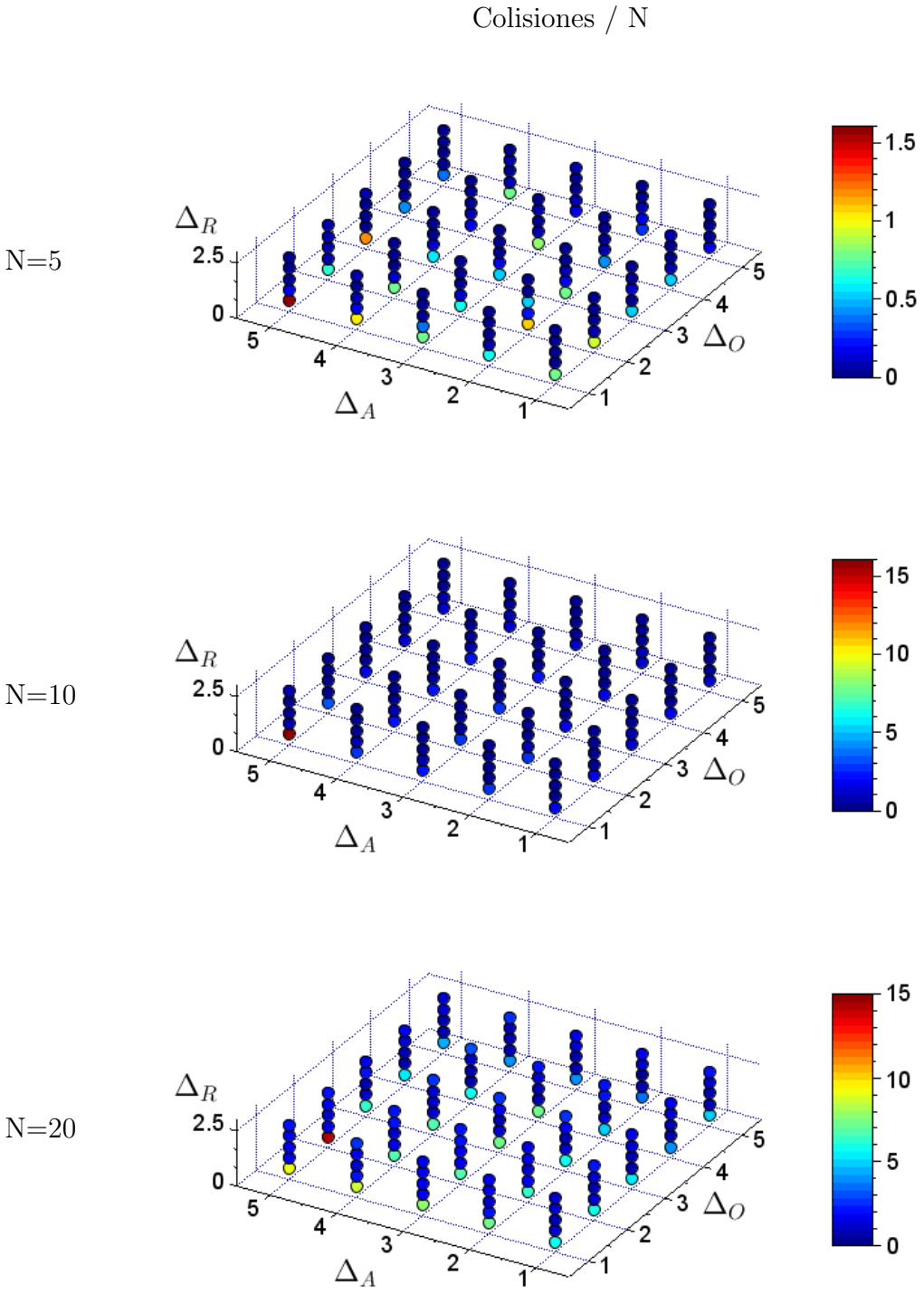


Figura 4.14: Colisiones por robot para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el experimento de navegación con obstáculos. Cada punto en la gráfica es el promedio de 10 repeticiones.

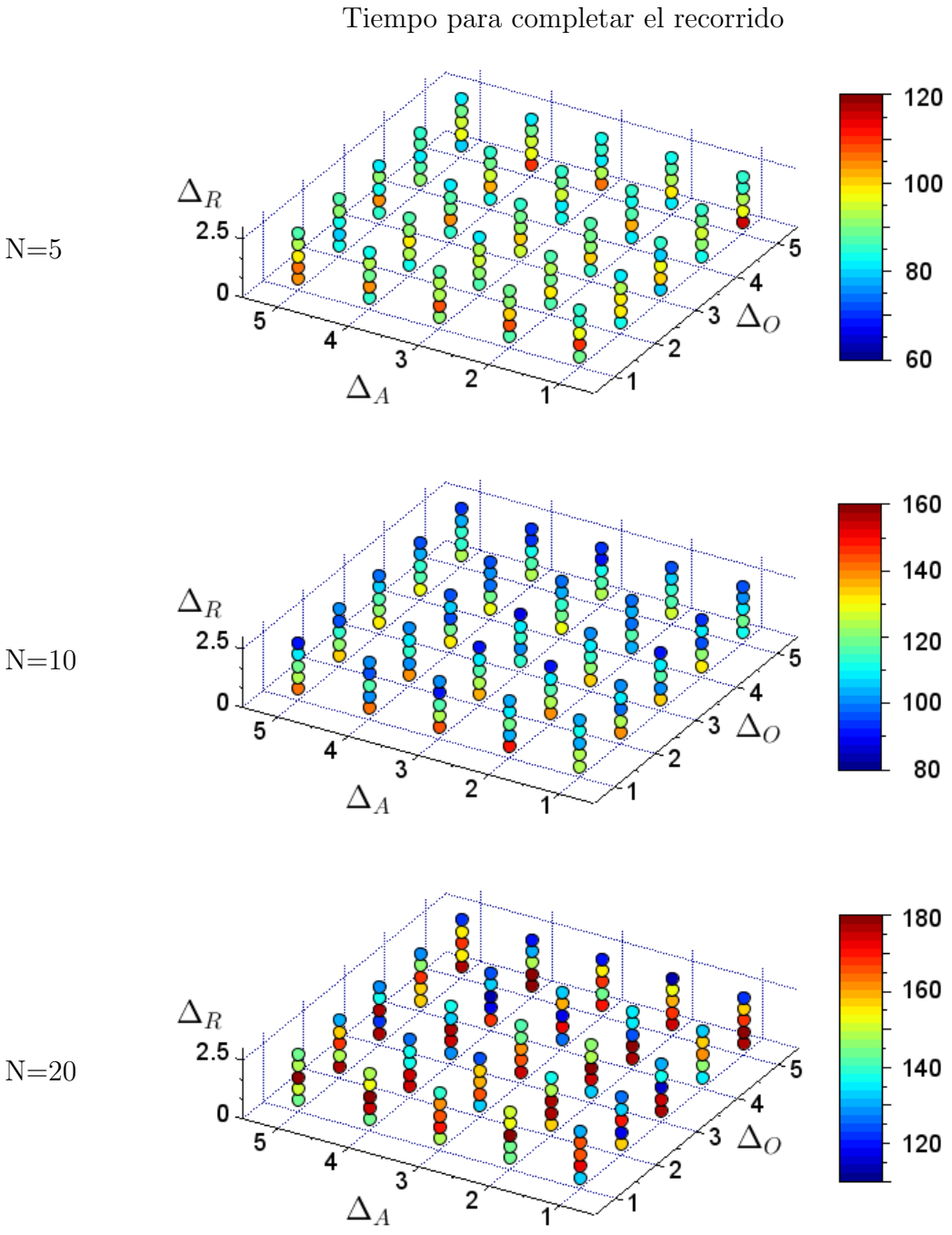


Figura 4.15: Tiempo en que todos los robots completan el recorrido para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el experimento de navegación con obstáculos. Cada punto en la gráfica es el promedio de 10 repeticiones.

En la Tabla 4.4 se muestran las combinaciones de parámetros que obtuvieron los mejores resultados en esta tarea. Se presentan los resultados del número de colisiones por robot, así como el tiempo que tardó el primer robot en llegar al objetivo t_1 y el tiempo que tardó todo el enjambre t_N. Mientras más grande es el enjambre, mayor es el tiempo que tarda en completar el recorrido porque aumenta el número de interacciones entre robots al momento de esquivar los obstáculos.

Tabla 4.4: Resultados del experimento de navegación con obstáculos

N	Δ_R	Δ_O	Δ_A	n_{col} / N	t_1 (s)	t_N (s)
	1	4	5	0	70.96	81.88
	1	1	1	0.02	69.52	81.43
5	1	4	1	0.08	69.11	81.33
	1.5	1	4	0	69.41	83.70
	1	1	1	0.10	70.50	81.53
	0.5	5	4	1.55	68.39	88.79
	1	5	3	0.50	68.82	93.96
10	1	5	4	0.55	69.21	93.91
	1	5	1	0.26	68.90	95.06
	0.5	5	1	1.46	68.63	90.66
	1	5	5	0.89	69.07	120.04
	1	5	4	0.81	71.86	121.09
20	1	5	3	0.94	70.38	121.43
	1	5	2	0.92	69.20	121.87
	1	4	4	0.78	69.62	124.39

En las Figuras 4.16, 4.17 y 4.18 se muestran ejemplos de las simulaciones de este experimento. Las líneas punteadas en color azul representan las trayectorias de los robots del enjambre. Se puede observar que los robots logran detectar los obstáculos y giran para evitarlos.

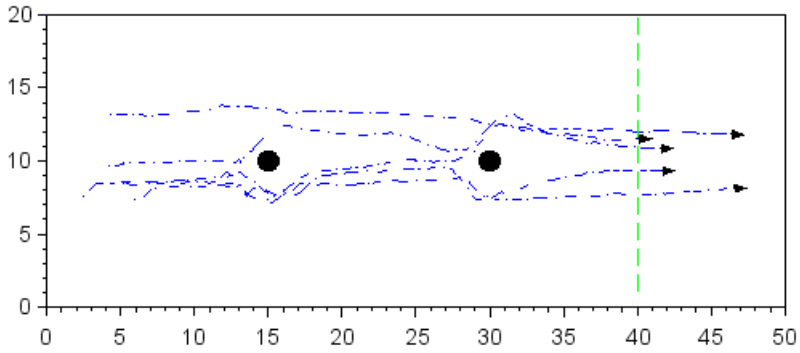


Figura 4.16: Ejemplo de simulación de navegación con obstáculos. Enjambre de 5 robots, $\Delta_R = 1$ m, $\Delta_O = 4$ m y $\Delta_A = 5$ m

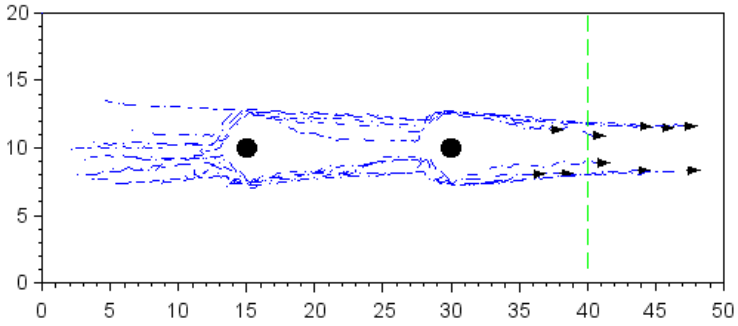


Figura 4.17: Ejemplo de simulación de navegación con obstáculos. Enjambre de 10 robots, $\Delta_R = 0.5$ m, $\Delta_O = 5$ m y $\Delta_A = 4$ m

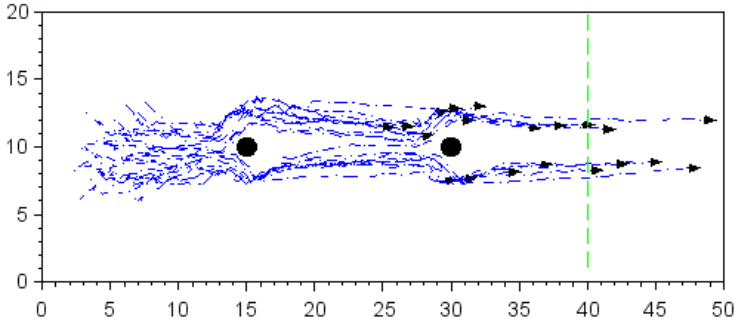


Figura 4.18: Ejemplo de simulación de navegación con obstáculos. Enjambre de 20 robots, $\Delta_R = 1$ m, $\Delta_O = 5$ m y $\Delta_A = 4$ m

4.4.2 Agregación y movimiento colectivo

La tarea de agregación consiste en reunir un conjunto de individuos muy cerca entre sí, mientras que el movimiento colectivo implica el desplazamiento de un grupo de individuos en una misma dirección.

Para poner a prueba estas dos tareas, los miembros del enjambre inician dispersos, con posiciones y orientaciones aleatorias dentro de una arena cuadrada de 500 a 2000 m^2 dependiendo de la cantidad de individuos. Los robots se mueven dentro de esta arena e interactúan para formar un grupo que se mueva coordinadamente. Para analizar el comportamiento emergente del enjambre, se exploran diferentes valores para los anchos de las zonas de comportamiento (0.5, 1.0, 1.5 2.0, 2.5) m para la zona de repulsión Δ_R y (1.0, 2.0, 3.0, 4.0, 5.0) m para las zonas de orientación Δ_O y atracción Δ_A. Estos valores dan un total de 125 combinaciones de parámetros. Cada combinación de parámetros es repetida 10 veces para obtener un promedio. Para verificar la capacidad del modelo ante cambios en el tamaño del enjambre se proponen tres tamaños de enjambre para la realización de la tarea, 5, 10 y 20 robots. La Tabla 4.5 resume los parámetros utilizados para simular la tarea de agregación y movimiento coordinado.

Tabla 4.5: Parámetros de simulación para agregación y movimiento colectivo

Parámetro	Descripción	Valor	Unidades
Δ_R	Zona de repulsión	0.5 - 2.5	m
Δ_O	Zona de orientación	1 - 5	m
Δ_A	Zona de atracción	1 - 5	m
N	Número de individuos	5 - 20	
t_{max}	tiempo de simulación	300	s
dt	paso de integración	0.05	s
z_d	altitud deseada	1	m
v_{max}	velocidad máxima	0.5	m/s

En la Figura 4.19 se presenta el número de colisiones por robot para cada tamaño de enjambre. Similar a los experimentos en 3 dimensiones, la mayor cantidad de colisiones se presenta cuando el ancho de la zona de repulsión Δ_R es de 0.5 m, mientras que con valores mayores o iguales a 1 m se reducen considerablemente las colisiones. El ancho de las zonas de atracción Δ_A y orientación Δ_O también tienen efecto en el número de colisiones pero en menor medida.

En el caso de la polarización del enjambre (Figura 4.20) los mejores resultados se observan con valores altos de la zona de orientación y atracción, así como valores intermedios de la zona de repulsión. Mientras mayor es el número de robots en el enjambre menor es la polarización que se alcanza. En cuanto a la métrica de unión Φ_{union} (Figura 4.21), los mejores resultados se obtienen con valores grandes en la zonas de orientación y atracción.

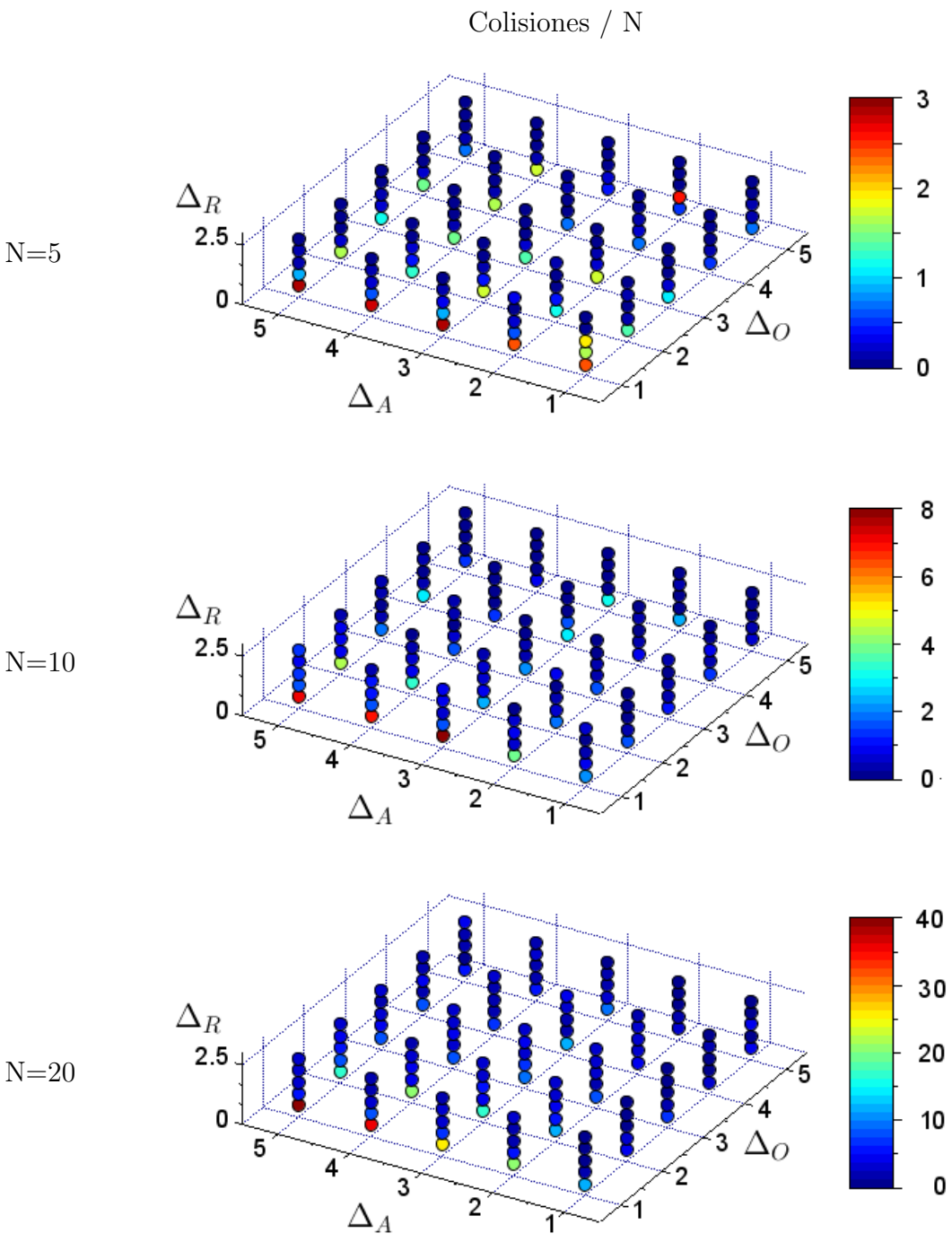


Figura 4.19: Cantidad de colisiones por robot para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el experimento de agregación y movimiento colectivo. Cada punto en la gráfica es el promedio de 10 repeticiones.

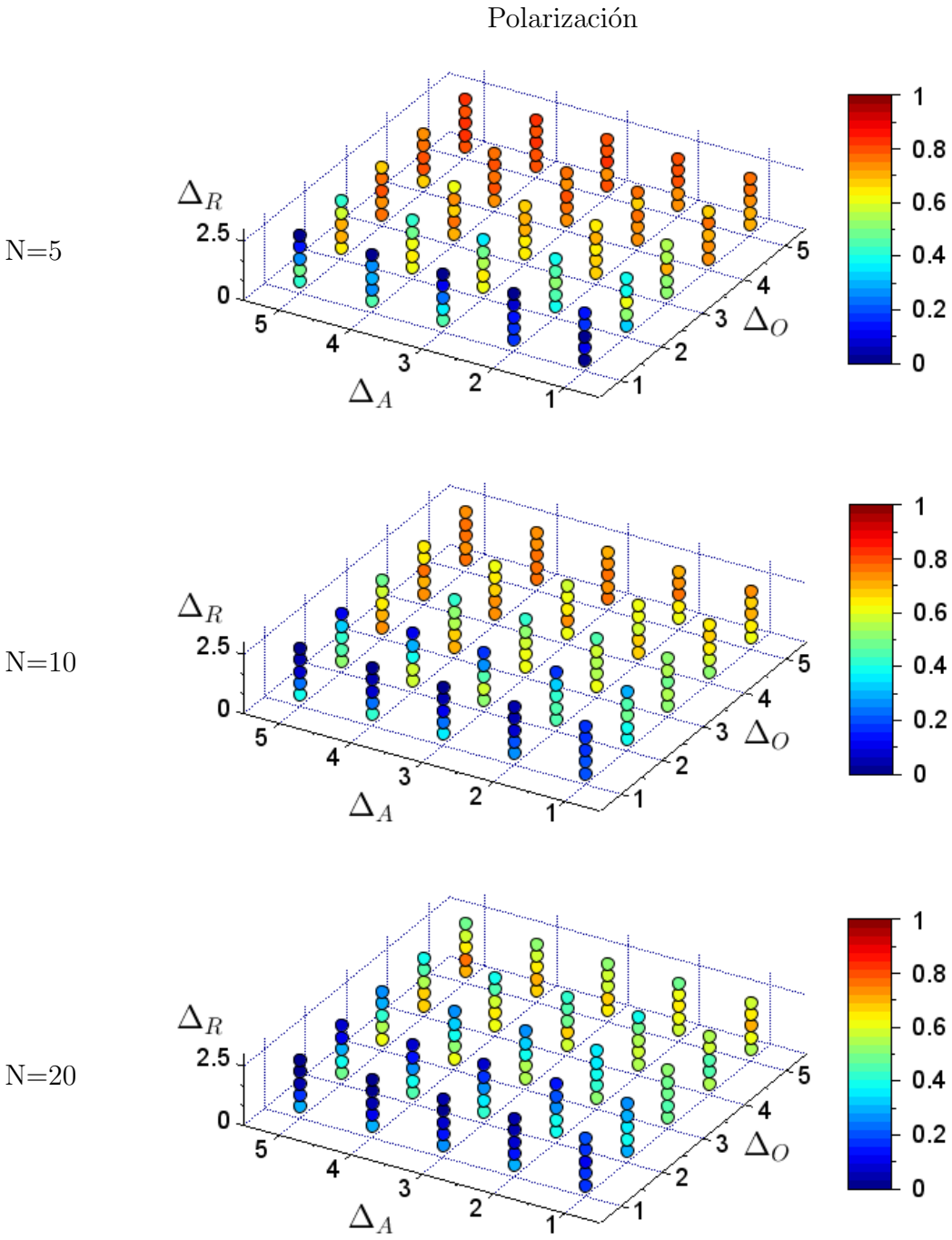


Figura 4.20: Polarización promedio durante el tiempo de simulación para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el experimento de agregación y movimiento colectivo. Cada punto en la gráfica es el promedio de 10 repeticiones.

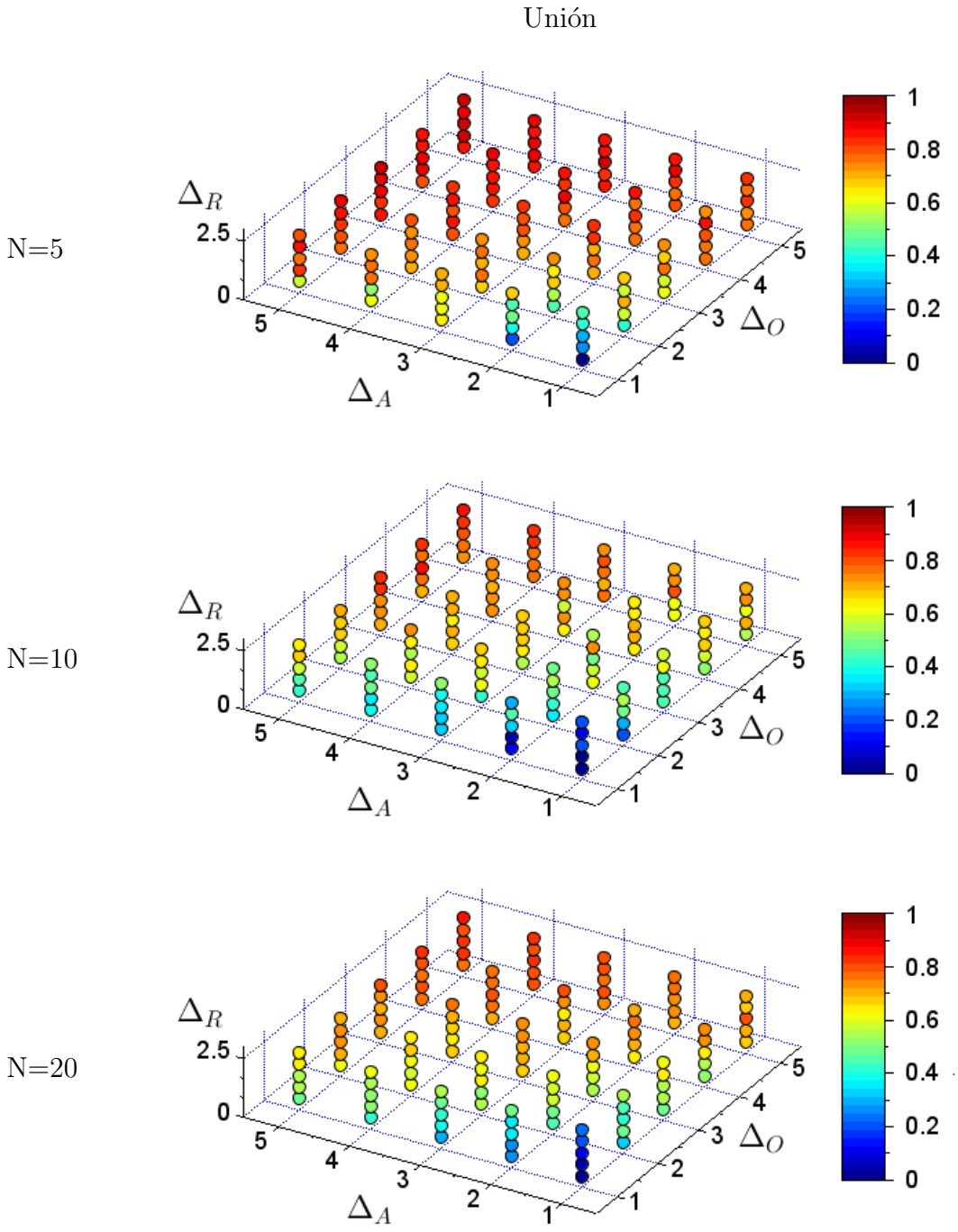


Figura 4.21: Unión promedio durante el tiempo de simulación para diferentes tamaños de enjambre y diferentes parámetros de comportamiento en el experimento de agregación y movimiento colectivo. Cada punto en la gráfica es el promedio de 10 repeticiones.

La Tabla 4.6 muestra las combinaciones de parámetros que obtuvieron los mejores resultados para los 3 tamaños de enjambre.

Tabla 4.6: Resultados del experimento de agregación y movimiento colectivo

N	Δ_R	Δ_O	Δ_A	n_{col} / N	ρ	Φ_{union}
	1.5	5	3	0.000	0.941	0.984
	2.5	5	5	0.000	0.940	0.981
5	2.5	5	4	0.000	0.933	0.976
	1.5	4	3	0.000	0.926	0.971
	1.5	5	4	0.000	0.929	0.968
	1.5	4	5	0.100	0.853	0.960
	2	5	4	0.010	0.837	0.954
10	2	5	5	0.060	0.843	0.953
	1.5	5	5	0.140	0.859	0.943
	1	5	4	0.140	0.862	0.937
	1	5	5	1.390	0.824	0.958
	1	5	4	1.650	0.771	0.943
20	1.5	5	2	0.700	0.741	0.924
	1.5	5	4	1.990	0.736	0.957
	1	4	5	2.485	0.758	0.942

En la Figuras 4.22, 4.23 y 4.24 se muestran capturas del experimento con 5, 10 y 20 robots respectivamente, utilizando zonas de orientación $\Delta_O = 1$ m y de atracción $\Delta_A = 1$ m. Además, debajo de las capturas se muestra la polarización del enjambre a lo largo del tiempo del experimento. Se puede observar que se forman pequeños subgrupos que se mueven en diferentes direcciones, sin lograr agrupar a todos los robots en un único grupo.

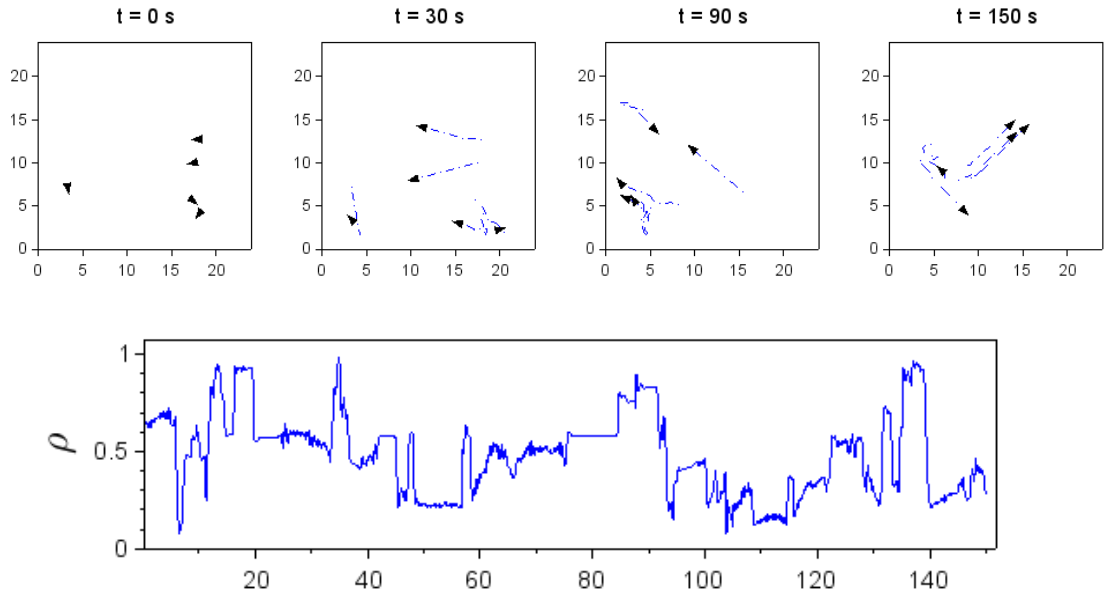


Figura 4.22: Ejemplo de simulación de agregación y movimiento colectivo con 5 robots. $\Delta_R = 1$ m, $\Delta_O = 1$ m y $\Delta_A = 1$ m. Se muestra además la polarización del grupo.

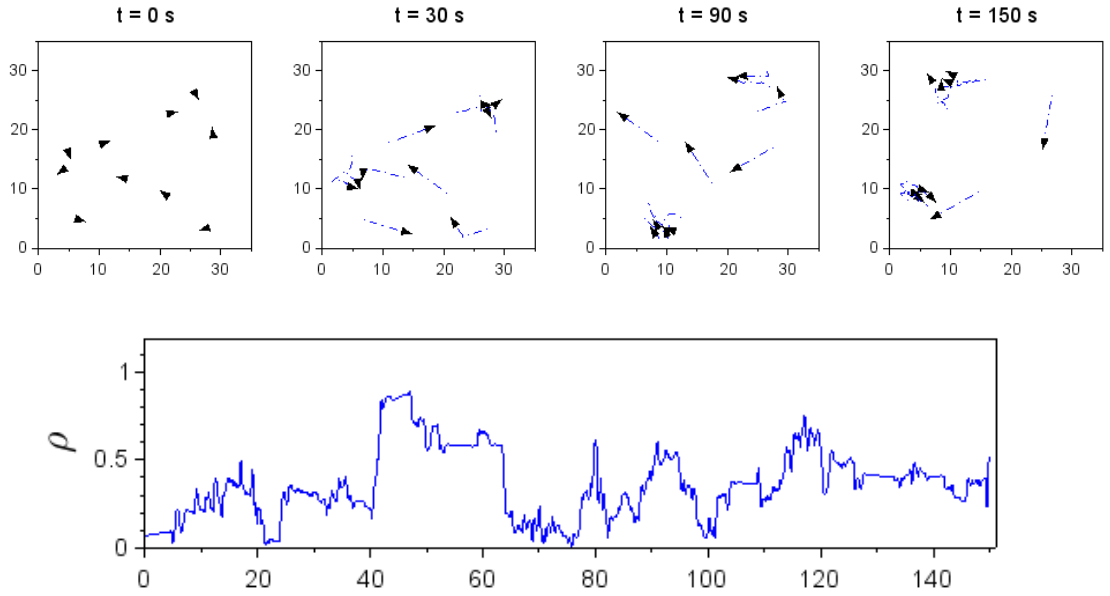


Figura 4.23: Ejemplo de simulación de agregación y movimiento colectivo con 10 robots. $\Delta_R = 1$ m, $\Delta_O = 1$ m y $\Delta_A = 1$ m. Se muestra además la polarización del grupo.

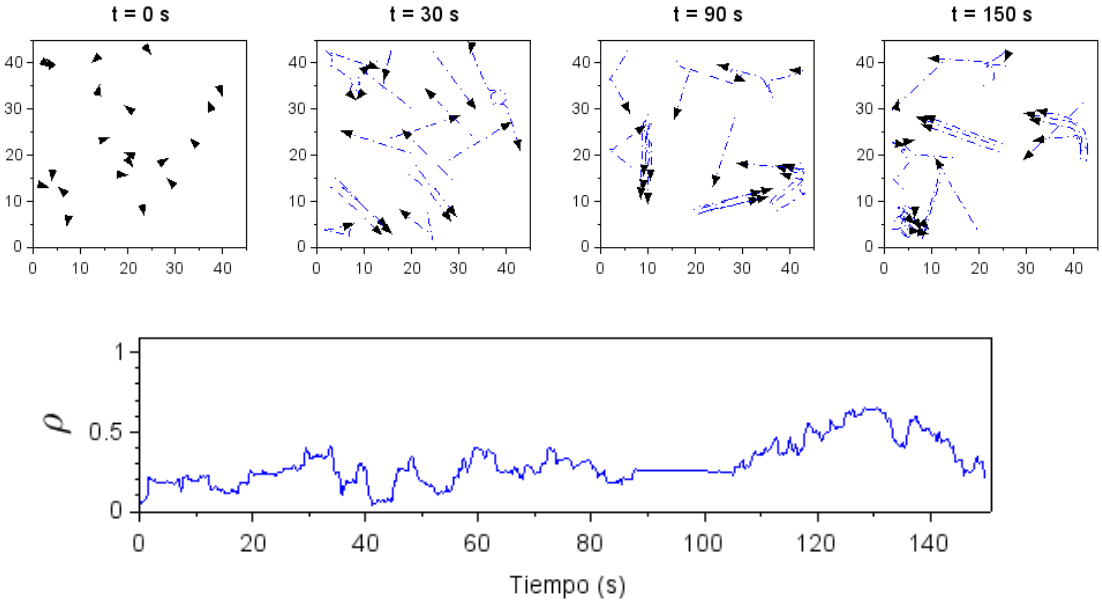


Figura 4.24: Ejemplo de simulación de agregación y movimiento colectivo con 20 robots. $\Delta_R = 1$ m, $\Delta_O = 1$ m y $\Delta_A = 1$ m. Se muestra además la polarización del grupo.

En la Figura 4.25 se muestran capturas de diferentes instantes de tiempo del experimento y también la polarización del enjambre de 5 robots. En esta ocasión los valores de la zonas de orientación y atracción son mayores, lo que nos da como resultado un comportamiento más deseado para esta tarea. Aproximadamente partir de los 35 segundos los 5 robots se mueven como un sólo grupo y siguen así hasta que llegan al límite del área, cuando algunos comienzan a girar y se pierde la polarización por un pequeño instante. Algo importante a resaltar es que no todos los robots necesitan detectar el límite del área como ocurre en la imagen de los 90 s, donde sólo tres robots se acercan lo suficiente a este límite y los otros dos giran para moverse en la misma dirección que todo el grupo.

Para el enjambre de 10 robots mostrado en la Figura 4.26 se utiliza una zona de orientación $\Delta_O = 4$ m y una zona de atracción $\Delta_A = 5$ m. El tiempo para alcanzar el movimiento colectivo es aproximadamente 53 segundos y se mantiene durante el resto de la simulación. Algo similar ocurre en el ejemplo de 20 robots que se presenta

en la Figura 4.27, donde se utiliza una zona de orientación $\Delta_O = 5$ m y una zona de atracción $\Delta_A = 5$ m. En este caso se alcanza una polarización del enjambre de forma lenta. El movimiento colectivo se mantiene hasta que el enjambre llega al límite del área donde se produce un desorden en la orientación del enjambre y toma un poco de tiempo para volver al valor máximo de polarización.

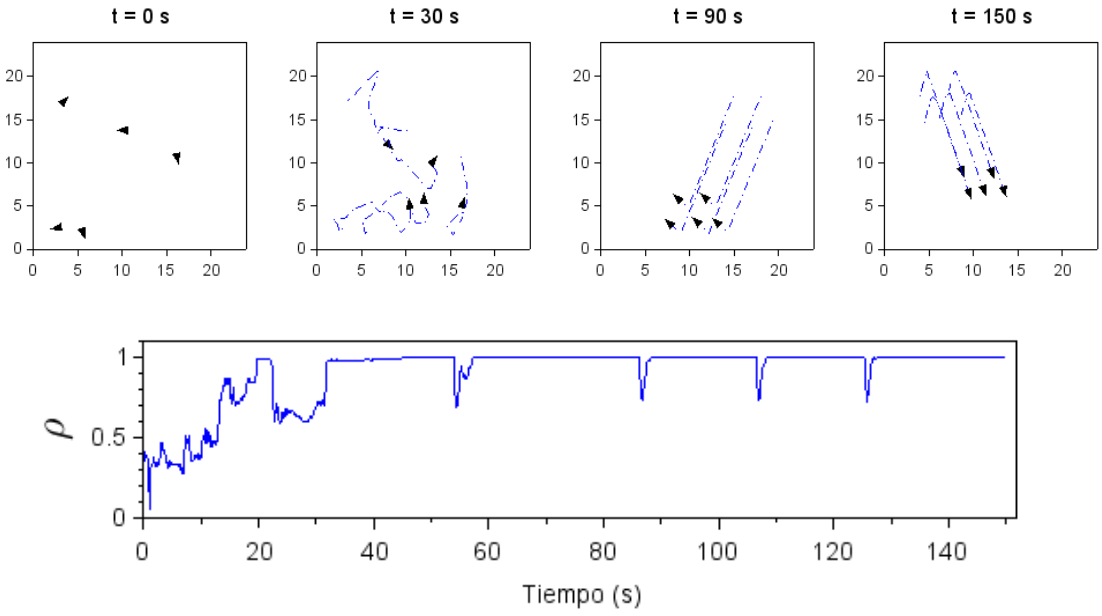


Figura 4.25: Ejemplo de simulación de agregación y movimiento colectivo con 5 robots. $\Delta_R = 1.5$ m, $\Delta_O = 5$ m y $\Delta_A = 3$ m. Se muestra además la polarización del grupo.

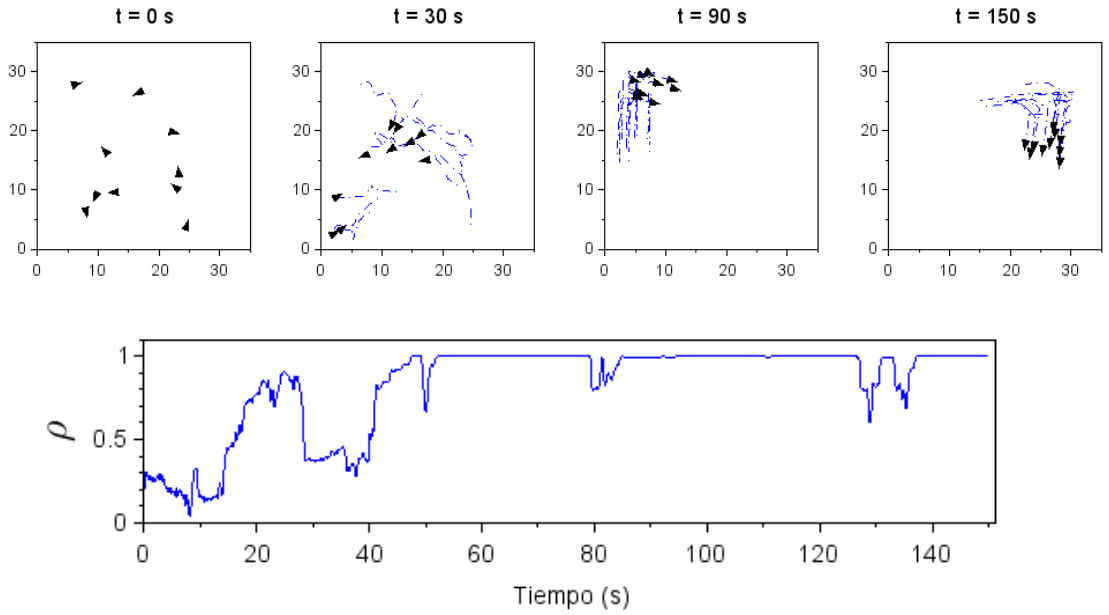


Figura 4.26: Ejemplo de simulación de agregación y movimiento colectivo con 10 robots. $\Delta_R = 1.5m$, $\Delta_O = 4m$ y $\Delta_A = 5m$. Se muestra además la polarización del grupo.

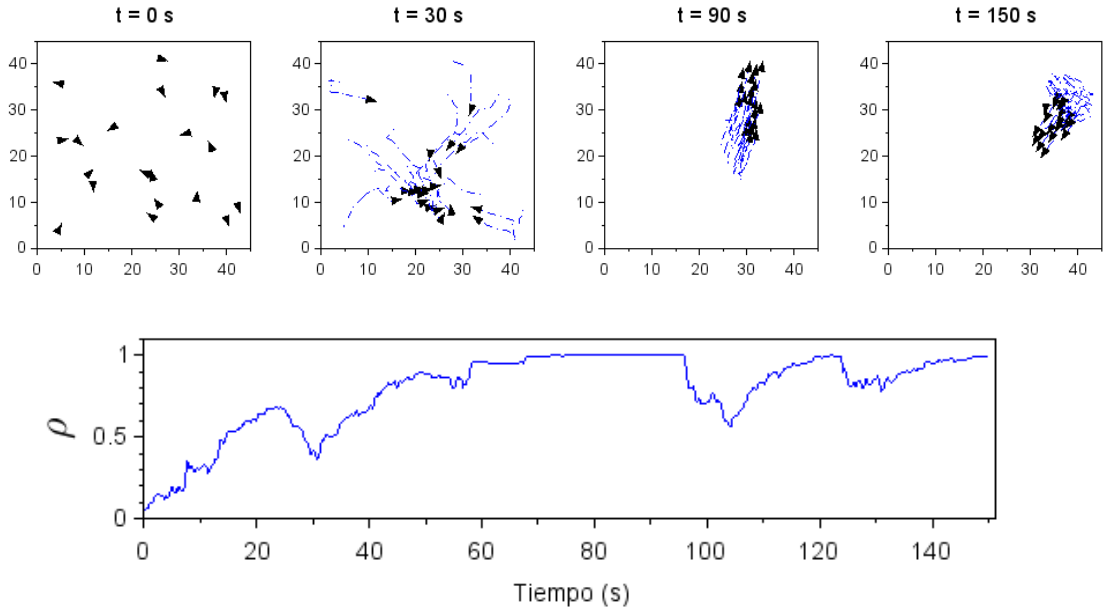


Figura 4.27: Ejemplo de simulación de agregación y movimiento colectivo con 20 robots. $\Delta_R = 1m$, $\Delta_O = 5m$ y $\Delta_A = 5m$. Se muestra además la polarización del grupo.

Mientras mayor es el número de robots en el enjambre, mayor es la zona de orientación necesaria para obtener un movimiento colectivo. En la Figura 4.28 se muestra un ejemplo de simulación con 50 robots, en un área de 75 x 75 m y un tiempo de simulación de 600 segundos. En este caso se utiliza un zona de orientación $\Delta_O = 5$ m y una zona de atracción $\Delta_A = 5$ m. El tamaño de la zona de atracción es suficiente para mantener a todos los robots juntos, pero el tamaño de la zona de orientación no es suficiente para que el enjambre llegue a un consenso sobre la dirección a seguir, lo cual se aprecia en la gráfica de la polarización del enjambre en la Figura 4.29.

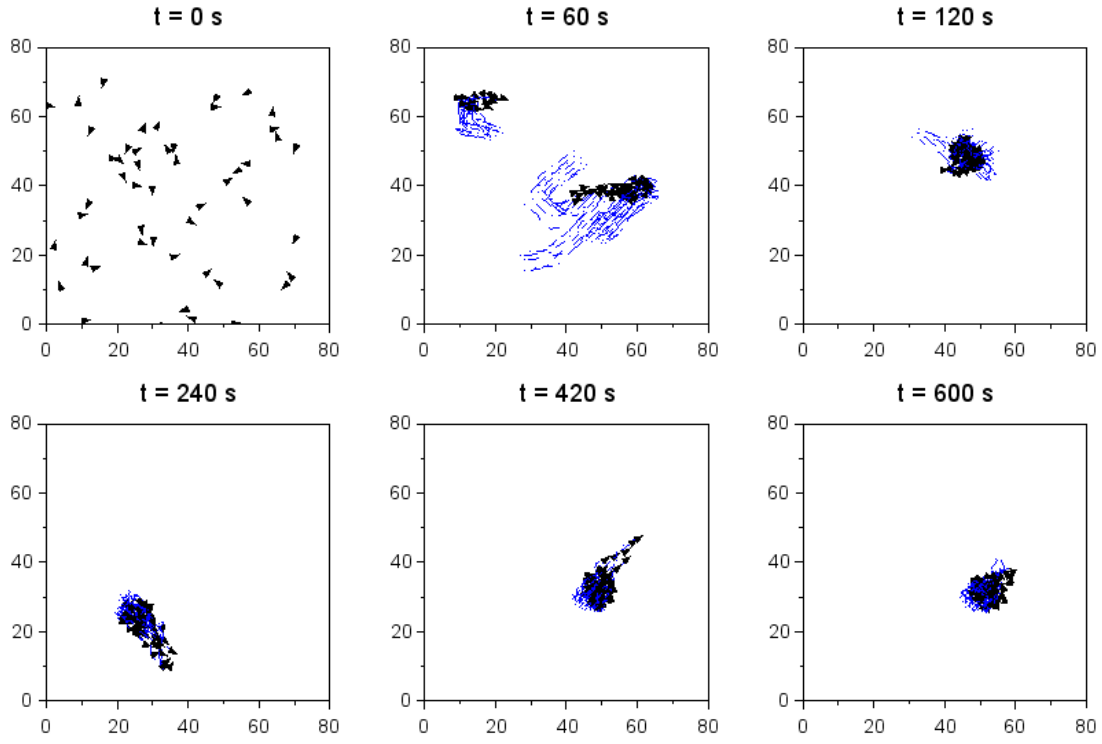


Figura 4.28: Ejemplo de simulación de agregación y movimiento colectivo con 50 robots. $\Delta_R = 1m$, $\Delta_O = 5m$ y $\Delta_A = 5m$.

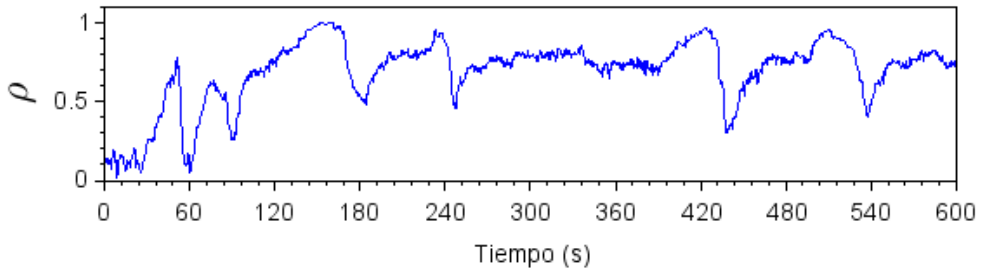


Figura 4.29: Polarización de grupo del enjambre de 50 robots. $\Delta_R = 1m$, $\Delta_O = 5m$ y $\Delta_A = 5m$.

En la Figura 4.30 se incrementó el tamaño de la zona de orientación a un valor de 15 m, con el fin de obtener un comportamiento similar al observado previamente con enjambres de menos robots. De esta forma los 50 robots logran moverse en la misma dirección y el enjambre se mueve como un solo grupo.

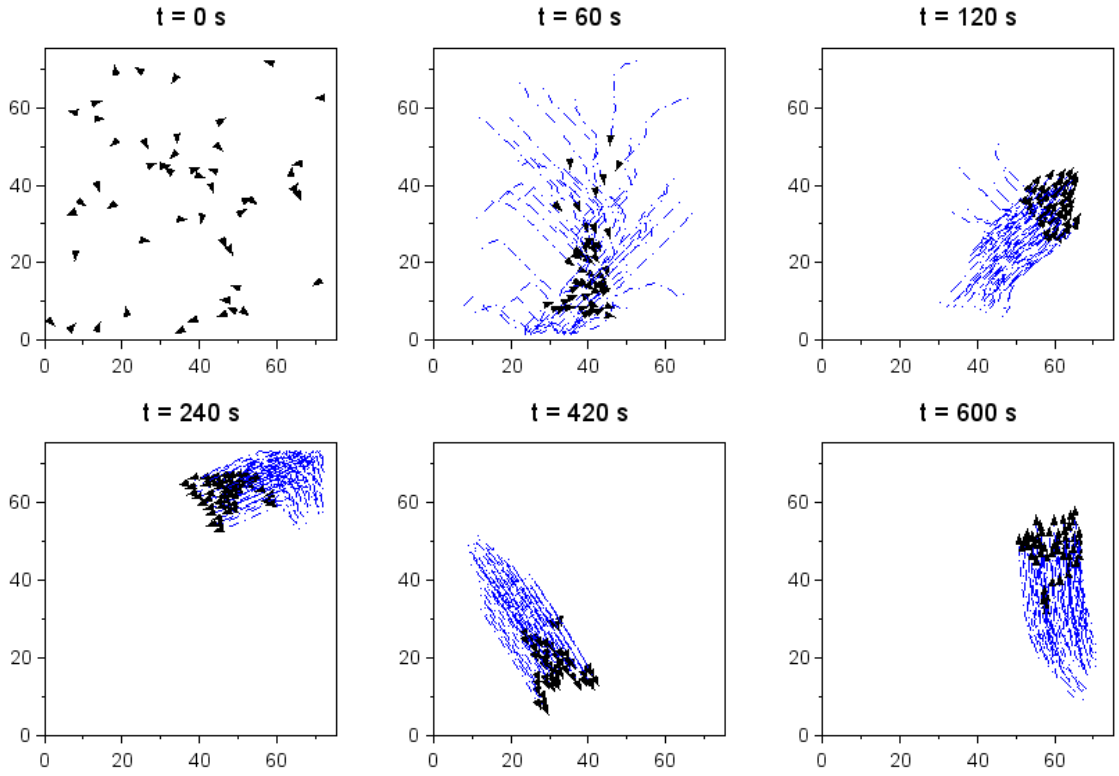


Figura 4.30: Ejemplo de simulación de agregación y movimiento colectivo con 50 robots. $\Delta_R = 1m$, $\Delta_O = 15m$ y $\Delta_A = 5m$.

Una vez que los robots están cerca unos a otros comienzan a moverse en la misma dirección como un sólo grupo. Esto se puede apreciar en la figura 4.31 donde se grafica la polarización del enjambre durante todo el tiempo de simulación. Esta gráfica nos muestra que todos los robots se mueven en la misma dirección la mayor parte del tiempo a excepción de los primeros 90 segundos y en los momentos en que el enjambre llega al límite del área de prueba. En la gráfica de la Figura 4.32 se muestra la distancia promedio de cada robot al centro del enjambre. Se puede ver que los robots inician dispersos y por eso el valor inicial es muy grande, pero cuando logran agruparse permanecen juntos y la distancia promedio al centro del enjambre varía muy poco.

Por último, en la Figura 4.33 se muestra el número de robots en cada estado de comportamiento a lo largo del experimento. Se muestra el promedio de 10 repeticiones con su desviación estándar. En un inicio la mayoría de los robots están en un estado de orientación - atracción, pero a medida que se acercan unos con otros la mayoría de los robots pasa a un estado de orientación. Se pueden apreciar también incrementos en el número de robots en el estado de repulsión que corresponden con los momentos en que el enjambre llega al límite del área y comienza a girar, por lo que el enjambre tiene que reordenarse.

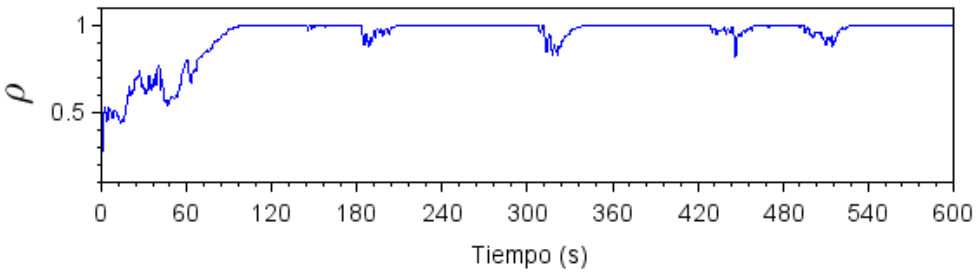


Figura 4.31: Polarización de grupo del enjambre de 50 robots. $\Delta_R = 1m$, $\Delta_O = 15m$ y $\Delta_A = 5m$.

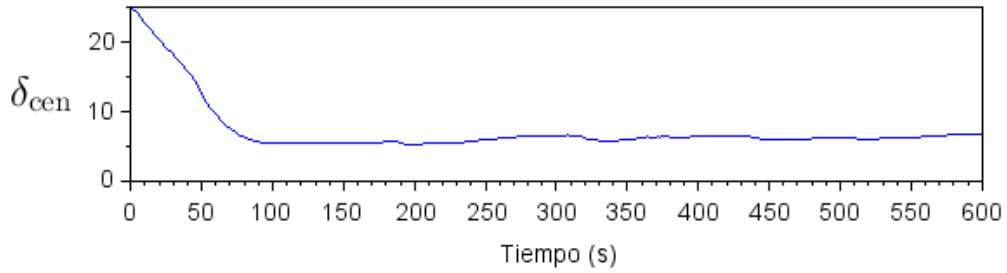


Figura 4.32: Distancia promedio al centro del enjambre de 50 robots. $\Delta_R = 1m$, $\Delta_O = 15m$ y $\Delta_A = 5m$.

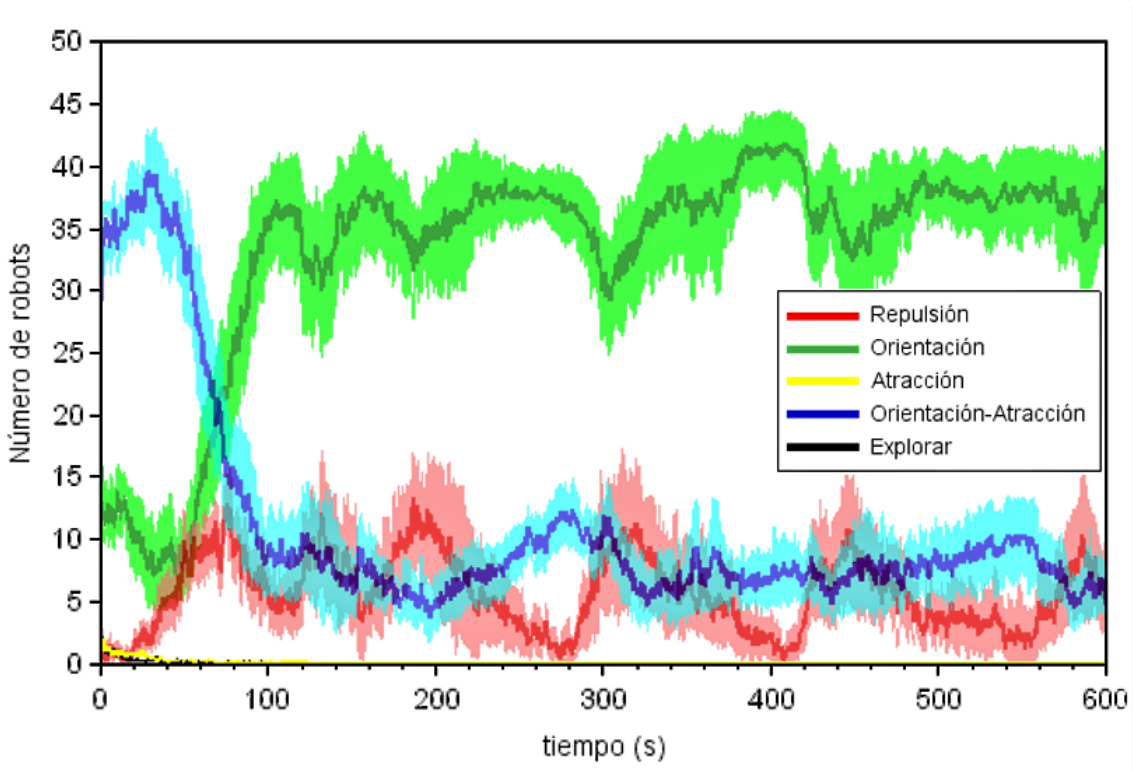


Figura 4.33: Número de robots en cada estado de comportamiento en la tarea de agregación y movimiento coordinado con 50 robots, $\Delta_R = 1m$, $\Delta_O = 15m$ y $\Delta_A = 5m$. Se muestra la media y desviación estándar de 10 repeticiones

Capítulo 5

Conclusiones y Trabajo a Futuro

En este trabajo de investigación se ha propuesto un enfoque de modelo de comportamiento bioinspirado para la coordinación de un grupo de robots cuadrirrotores. Este enfoque se basa en reglas simples de comportamiento que dependen de información local percibida por cada robot, contrario a otros enfoques que dependen de información global o control centralizado. El modelo de comportamiento bioinspirado está basado en modelos propuestos en la literatura para estudiar el comportamiento de animales moviéndose en grupo, pero adaptado a las capacidades sensoriales de los robots. Las principales contribuciones de este trabajo son la adaptación del modelo de comportamiento y la implementación de la plataforma de simulación para su validación.

Para validar el modelo de comportamiento se desarrolló una plataforma de simulación que incorpora el modelo dinámico de un cuadrirrotor y controladores PID para regular su vuelo, así como las limitaciones sensoriales que podría tener un robot real. De esta forma se logró simular enjambres con distintas configuraciones paramétricas para estudiar el comportamiento colectivo emergente.

Los resultados de las simulaciones tanto en 3 dimensiones como en 2 dimen-

siones muestran que es posible lograr un movimiento colectivo de los miembros del enjambre con el modelo de comportamiento implementado. El cambio en los parámetros de cada individuo (Δ_R, Δ_O, Δ_A) afecta el comportamiento general del enjambre. El ancho de la zona de repulsión ayuda a mantener una distancia entre robots, consiguiendo disminuir el número de colisiones. Los experimentos realizados nos muestran que el valor más adecuado para la zona de repulsión es de 1 m. Valores menores a 1 m ocasionan un gran número de colisiones, mientras que valores mayores no muestran una mejoría significativa y tienen un efecto negativo en otras métricas como la polarización.

La polarización se ve mejorada principalmente por la zona de orientación y en menor medida por la zona de atracción; mientras más grandes son estas dos zonas mejor es la alineación que se obtiene entre los individuos y por consecuencia el enjambre se mueve como un grupo bien ordenado. La zona de atracción permite mantener cerca a los robots vecinos, evitando que se separen y ayudando a mantener un sólo grupo.

La simulación de tareas con obstáculos demostraron que el comportamiento que se obtiene se puede adaptar a cambios en el entorno. Esto es de gran importancia ya que no limita el área de operación del enjambre a un espacio predefinido, sino que le permite operar en entornos desconocidos. Los diferentes tamaños de enjambre simulados demuestran que el comportamiento es escalable en cierta medida; al incrementa el número de robots en el enjambre, el desempeño tiende a decaer. Esta decremento del desempeño se debe a que el modelo de comportamiento considera sólo una parte de los robots dentro del rango de detección, en función del número de sensores. El uso del parámetro de influencia Δ_I nos permite interactuar con el enjambre para guiarlo a una actividad deseada; en este caso se utilizó para que el enjambre se desplace sin que todos los robots conozcan la trayectoria a seguir.

Para un trabajo posterior se recomienda estudiar el efecto de utilizar mayores velocidades en los robots y adaptar las reglas de comportamiento para responder

mejor a velocidades más altas. El uso de velocidades más altas permitiría utilizar el enjambre en tareas al aire libre donde se necesite recorrer mayores distancias. Además, queda pendiente la implementación del modelo de comportamiento en un enjambre de robots reales. Para esto aún existen algunas complicaciones como el rango de alcance de los sensores disponibles y establecer mecanismos que permitan diferenciar entre otros robots y obstáculos.

Bibliografía

[1] I. D. Couzin, J. Krause, R. James, G. D. Ruxton, and N. R. Franks, "Collective memory and spatial sorting in animal groups," *Journal of theoretical biology*, vol. 218, no. 1, pp. 1–11, 2002.

[2] M. Hassanalian and A. Abdelkefi, "Classifications, applications, and design challenges of drones: A review," *Progress in Aerospace Sciences*, vol. 91, pp. 99–131, 2017.

[3] E. Şahin, "Swarm robotics: From sources of inspiration to domains of application," in *International workshop on swarm robotics*, pp. 10–20, Springer, 2004.

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

[5] Y. Tan and Z.-Y. Zheng, "Research advance in swarm robotics," *Defence Technology*, vol. 9, no. 1, pp. 18–39, 2013.

[6] L. Bayındır, "A review of swarm robotics tasks," *Neurocomputing*, vol. 172, pp. 292–321, 2016.

[7] D. P. Stormont, "Autonomous rescue robot swarms for first responders," in *Computational Intelligence for Homeland Security and Personal Safety, 2005. CIHSPS 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 151–157, IEEE, 2005.

[8] G. Kantor, S. Singh, R. Peterson, D. Rus, A. Das, V. Kumar, G. Pereira, and J. Spletzer, "Distributed search and rescue with robot and sensor teams," in *Field and Service Robotics*, pp. 529–538, Springer, 2003.

[9] E. U. Acar, H. Choset, Y. Zhang, and M. Schervish, "Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods," *The International journal of robotics research*, vol. 22, no. 7-8, pp. 441–466, 2003.

[10] F. Morbidi, R. A. Freeman, and K. M. Lynch, "Estimation and control of uav swarms for distributed monitoring tasks," in *American Control Conference (ACC), 2011*, pp. 1069–1075, IEEE, 2011.

[11] R. Gross and M. Dorigo, "Towards group transport by swarms of robots," *International Journal of Bio-Inspired Computation*, vol. 1, no. 1-2, pp. 1–13, 2009.

[12] S. Gupte, P. I. T. Mohandas, and J. M. Conrad, "A survey of quadrotor unmanned aerial vehicles," in *Southeastcon, 2012 Proceedings of IEEE*, pp. 1–6, IEEE, 2012.

[13] Intel, "Intel and drone technology – breaking new ground." url=https://newsroom.intel.com/editorials/intel-and-drone-technology-breaking-new-ground/, 2016.

[14] Intel, "Intel breaks guinness world records title for drone light shows in celebration of 50th anniversary." url=https://newsroom.intel.com/news/intel-breaks-guinness-world-records-title-drone-light-shows-celebration-50th-anniversary/, 2018.

[15] T. Nestmeyer, P. R. Giordano, H. H. Bülthoff, and A. Franchi, "Decentralized simultaneous multi-target exploration using a connected network of multiple robots," *Autonomous Robots*, vol. 41, no. 4, pp. 989–1011, 2017.

[16] D. Albani, J. IJsselmuiden, R. Haken, and V. Trianni, "Monitoring and mapping with robot swarms for agricultural applications," in *2017 14th IEEE Interna-*

tional Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 1–6, IEEE, 2017.

[17] C. Reymann, A. Renzaglia, F. Lamraoui, M. Bronz, and S. Lacroix, "Adaptive sampling of cumulus clouds with uavs," *Autonomous robots*, vol. 42, no. 2, pp. 491–512, 2018.

[18] Amazon.com, "Amazon prime air." `https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011`.

[19] Wing Aviation LLC, "Air delivery." `https://wing.com/`.

[20] Deutsche Post DHL Group, "Dhl parcelcopter." `https://www.dpdhl.com/en/media-relations/specials/dhl-parcelcopter.html`.

[21] B. Rabta, C. Wankmüller, and G. Reiner, "A drone fleet model for last-mile distribution in disaster relief operations," *International Journal of Disaster Risk Reduction*, vol. 28, pp. 107–112, 2018.

[22] J. Scott and C. Scott, "Drone delivery models for healthcare," in *Proceedings of the 50th Hawaii international conference on system sciences*, 2017.

[23] A. Bürkle, F. Segor, and M. Kollmann, "Towards autonomous micro uav swarms," *Journal of intelligent & robotic systems*, vol. 61, no. 1-4, pp. 339–353, 2011.

[24] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Precision flight control for a multi-vehicle quadrotor helicopter testbed," *Control engineering practice*, vol. 19, no. 9, pp. 1023–1036, 2011.

[25] G. Vásárhelyi, C. Virágh, G. Somorjai, N. Tarcai, T. Szorenyi, T. Nepusz, and T. Vicsek, "Outdoor flocking and formation flight with autonomous aerial robots," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 3866–3873, IEEE, 2014.

[26] G. Vásárhelyi, C. Virágh, G. Somorjai, T. Nepusz, A. E. Eiben, and T. Vicsek, "Optimized flocking of autonomous drones in confined environments," *Science Robotics*, vol. 3, no. 20, p. eaat3536, 2018.

[27] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, "Towards a swarm of agile micro quadrotors," *Autonomous Robots*, vol. 35, no. 4, pp. 287–300, 2013.

[28] M. Saska, T. Baca, J. Thomas, J. Chudoba, L. Preucil, T. Krajnik, J. Faigl, G. Loianno, and V. Kumar, "System for deployment of groups of unmanned micro aerial vehicles in gps-denied environments using onboard visual relative localization," *Autonomous Robots*, vol. 41, no. 4, pp. 919–944, 2017.

[29] V. Walter, N. Staub, M. Saska, and A. Franchi, "Mutual localization of uavs based on blinking ultraviolet markers and 3d time-position hough transform," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, pp. 298–303, IEEE, 2018.

[30] T. Stirling, J. Roberts, J.-C. Zufferey, and D. Floreano, "Indoor navigation with a swarm of flying robots," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 4641–4647, IEEE, 2012.

[31] M. Coppola, K. N. McGuire, K. Y. Scheper, and G. C. de Croon, "On-board communication-based relative localization for collision avoidance in micro air vehicle teams," *Autonomous Robots*, pp. 1–19, 2018.

[32] K. McGuire, C. De Wagter, K. Tuyls, H. Kappen, and G. C. de Croon, "Minimal navigation solution for a swarm of tiny flying robots to explore an unknown environment," *Science Robotics*, vol. 4, no. 35, p. eaaw9710, 2019.

[33] G. Beni, "From swarm intelligence to swarm robotics," in *International Workshop on Swarm Robotics*, pp. 1–9, Springer, 2004.

[34] F. Aznar, M. Sempere, M. Pujol, R. Rizo, and M. Pujol, "Modelling oil-spill detection with swarm drones," in *Abstract and Applied Analysis*, vol. 2014, Hindawi, 2014.

[35] Y. Altshuler, V. Yanovsky, I. A. Wagner, and A. M. Bruckstein, "Efficient cooperative search of smart targets using uav swarms," *Robotica*, vol. 26, no. 4, pp. 551–557, 2008.

[36] N. Nigam, S. Bieniawski, I. Kroo, and J. Vian, "Control of multiple uavs for persistent surveillance: algorithm and flight test results," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 5, pp. 1236–1251, 2011.

[37] Y. C. Tan and B. Bishop, "Evaluation of robot swarm control methods for underwater mine countermeasures," in *Thirty-Sixth Southeastern Symposium on System Theory, 2004. Proceedings of the*, pp. 294–298, IEEE, 2004.

[38] E. Vassev, R. Sterritt, C. Rouff, and M. Hinchey, "Swarm technology at nasa: building resilient systems," *IT Professional*, vol. 14, no. 2, pp. 36–42, 2012.

[39] B. T. Fine and D. A. Shell, "Unifying microscopic flocking motion models for virtual, robotic, and biological flock members," *Autonomous Robots*, vol. 35, no. 2-3, pp. 195–219, 2013.

[40] K. Nonami, F. Kendoul, S. Suzuki, W. Wang, and D. Nakazawa, *Autonomous flying robots: unmanned aerial vehicles and micro aerial vehicles*. Springer Science & Business Media, 2010.

[41] S. Kernbach, *Handbook of collective robotics: fundamentals and challenges*. CRC Press, 2013.

[42] S. Musa, "Techniques for quadcopter modeling and design: A review," *Journal of Unmanned System Technology*, vol. 5, no. 3, pp. 66–75, 2018.

[43] S. Bouabdallah, P. Murrieri, and R. Siegwart, "Design and control of an indoor micro quadrotor," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 5, pp. 4393–4398, IEEE, 2004.

[44] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 2520–2525, IEEE, 2011.

[45] L. Derafa, T. Madani, and A. Benallegue, "Dynamic modelling and experimental identification of four rotors helicopter parameters," in *2006 IEEE International Conference on Industrial Technology*, pp. 1834–1839, IEEE, 2006.

[46] S. Bouabdallah, *Design and control of quadrotors with application to autonomous flying*. PhD thesis, Ecole Polytechnique Federale de Lausanne, 2007.

[47] H. Bouadi and M. Tadjine, "Nonlinear observer design and sliding mode control of four rotors helicopter," *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, vol. 1, no. 7, pp. 354–359, 2007.

[48] C. Wang, M. Nahon, and M. Trentini, "Controller development and validation for a small quadrotor with compensation for model variation," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 902–909, IEEE, 2014.

[49] B. L. Partridge, "The structure and function of fish schools," *Scientific american*, vol. 246, no. 6, pp. 114–123, 1982.

[50] W. K. Potts, "The chorus-line hypothesis of manoeuvre coordination in avian flocks," *Nature*, vol. 309, no. 5966, pp. 344–345, 1984.

[51] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, pp. 25–34, 1987.

[52] I. D. Couzin, J. Krause, N. R. Franks, and S. A. Levin, "Effective leadership and decision-making in animal groups on the move," *Nature*, vol. 433, no. 7025, pp. 513–516, 2005.

[53] J. F. Roberts, T. Stirling, J.-C. Zufferey, and D. Floreano, "3-d relative positioning sensor for indoor flying robots," *Autonomous Robots*, vol. 33, no. 1-2, pp. 5–20, 2012.

[54] E. Ordaz-Rivas, A. Rodriguez-Liñan, M. Aguilera-Ruiz, and L. Torres-Treviño, "Collective tasks for a flock of robots using influence factor," *Journal of Intelligent & Robotic Systems*, vol. 94, no. 2, pp. 439–453, 2019.

[55] S.-Y. Jung, D. S. Brown, and M. A. Goodrich, "Shaping couzin-like torus swarms through coordinated mediation," in *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1834–1839, IEEE, 2013.

[56] A.-L. Barabási *et al.*, *Network science*. Cambridge university press, 2016.

[57] F. R. Chung and F. C. Graham, *Spectral graph theory*. No. 92, American Mathematical Soc., 1997.

[58] E. Soria, F. Schiano, and D. Floreano, "The influence of limited visual sensing on the reynolds flocking algorithm," in *2019 Third IEEE International Conference on Robotic Computing (IRC)*, pp. 138–145, IEEE, 2019.

[59] T. Vicsek and A. Zafeiris, "Collective motion," *Physics reports*, vol. 517, no. 3-4, pp. 71–140, 2012.

[60] "Multiwii code." `https://code.google.com/archive/p/multiwii/`.

[61] STMicroelectronics, "Vl53l0x - world's smallest time-of-flight (tof) ranging sensor." `https://www.st.com/en/imaging-and-photonics-solutions/vl53l0x.html`.

[62] A. Chovancová, T. Fico, L. Chovanec, and P. Hubinsk, "Mathematical modelling and parameter identification of quadrotor (a survey)," *Procedia Engineering*, vol. 96, pp. 172–181, 2014.

[63] M. Elsamanty, A. Khalifa, M. Fanni, A. Ramadan, and A. Abo-Ismail, "Methodology for identifying quadrotor parameters, attitude estimation and control," in *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 1343–1348, IEEE, 2013.

[64] W. Dong, G.-Y. Gu, X. Zhu, and H. Ding, "Development of a quadrotor test bed—modelling, parameter identification, controller design and trajectory generation," *International Journal of Advanced Robotic Systems*, vol. 12, no. 2, p. 7, 2015.

Apéndice A

Prototipo de cuadrirrotor

Durante el desarrollo de esta investigación se creó un prototipo de cuadrirrotor con las características necesarias para implementar las reglas de comportamiento propuestas. El objetivo de este prototipo es realizar algunos experimentos para la identificación de los parámetros necesarios para las simulaciones, así como servir para futuros trabajos de investigación. El nombre del prototipo es *Starling* y se puede observar en la Figura A.1.

Figura A.1: Prototipo Cuadrirrotor Starling

A.1 Diseño

Se utilizó un software de diseño 3D para crear un modelo CAD (Diseño Asistido por Computadora) de los componentes del cuadrirrotor. Las dimensiones (Figura A.2) son de 382 mm de diámetro considerando los protectores de las hélices y 120 mm de altura. El cuerpo principal del cuadrirrotor se fabricó por medio de impresión 3D con material PLA. De esta forma es fácil la fabricación de piezas de repuesto, así como el rediseño de las piezas para acomodar distintos componentes. Algunas de las piezas impresas se pueden ver en la Figura A.3. Una vez ensamblada la estructura del cuadrirrotor y montados todos los componentes, la masa total es de 420 g. La batería de 1300 mAh le otorga una autonomía de vuelo de aproximadamente 10 minutos.

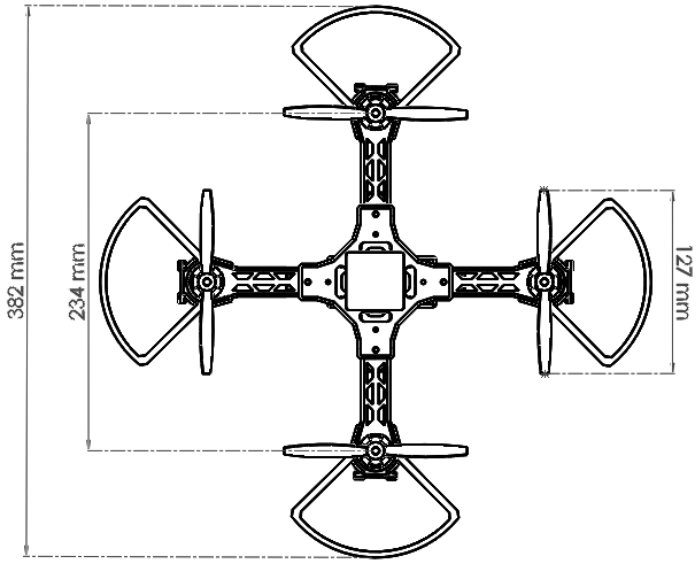


Figura A.2: Dimensiones del cuadrirrotor prototipo

Figura A.3: Partes fabricadas con impresión 3D

A.2 Componentes

La Figura A.4 ilustra algunos de los componentes y sus ubicación en el cuadrirrotor. A continuación se describen con mayor detalle algunos de los componentes de mayor relevancia.

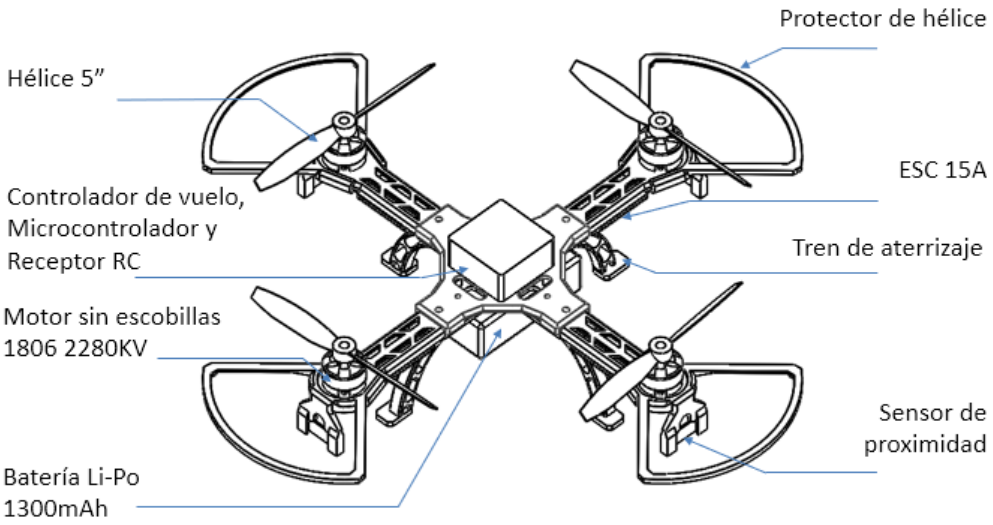


Figura A.4: Componentes del prototipo

A.2.1 Controlador de Vuelo

En el centro del cuadrirrotor se ubica una tarjeta controladora de vuelo CRIUS MultiWii SE v2.6 la cual utiliza el software de código abierto MultiWii [60]. Esta tarjeta incorpora los sensores y sistemas de control necesarios para que el cuadrirrotor pueda ejecutar las órdenes enviadas por el usuario. Algunas de sus características son:

Tabla A.1: Características del controlador de vuelo

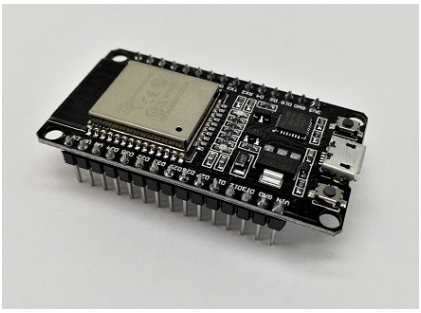
CRIUS MultiWii SE v2.6
- Microcontrolador ATmega328p - MPU-6050 acelerómetro 3 ejes + giroscopio 3 ejes - Magnetómetro de 3 ejes HMC5883L - Sensor digital de presión BMP85 - 6 canales de entrada - 8 canales de salida - Puerto FTDI para programación - Puerto I^2C - Voltaje de operación 5 V - Dimensiones 40 mm $\times$ 40 mm

A.2.2 Módulo ESP32

Se agregó un módulo ESP32-WROOM-32 para dotar al cuadrirrotor con la capacidad de realizar vuelo autónomo. Este módulo nos permite tomar las lecturas de los sensores de proximidad, procesar esta información y enviar las señales al controlador de vuelo. Además la capacidad Wi-Fi y Bluetooth le permite establecer comunicación inalámbrica con otros dispositivos. A continuación se enlistan algunas de las características de este módulo:

Tabla A.2: Características del módulo ESP32

ESP32-WROOM-32
- CPU Xtensa LX6 32-bits
- Frecuencia de reloj 80 MHz
- Wi-Fi 802.11 b/g/n
- Bluetooth v4.2 BLE
- 34 pines E/S programables
- Hasta 18 ADC de 12-bits
- Hasta 16 canales PWM
- 3 interfaces UART
- 2 interfaces I^2C
- 4 temporizadores de 64-bits
- Voltaje de operación 3.3 V
- Dimensiones 51 mm $\times$ 27 mm



A.2.3 Sensores de proximidad

Para poder detectar a otros robots u obstáculos, se agregaron sensores láser de rango por Tiempo de Vuelo (ToF) para medir distancias de manera precisa [61].

Tabla A.3: Características del sensor VL53L0X

Sensor VL53L0X
- Módulo miniatura
- Láser de 940 nm
- Microcontrolador embebido
- Medición de hasta 2 m
- Alta inmunidad a luz ambiental
- Alta inmunidad a la reflectancia del objetivo
- Interfaz I^2C
- Voltaje de operación 3.3 V

A.2.4 Sistema de Propulsión

El sistema de propulsión consta de 4 motores sin escobillas MT1806 2280KV, 4 controladores electrónicos de velocidad (ESC) de 15 A y 4 hélices de 5 pulgadas. El ESC recibe como entrada señales PWM del controlador de velocidad y envía pulsos de corriente a los embobinados del motor para controlar la velocidad. Los motores sin escobillas presentan ventajas como mayor potencia, alta velocidad, menor peso, control electrónico y bajo mantenimiento. Con las hélices seleccionadas cada motor puede brindar un empuje máximo aproximado de 400 g.

A.2.5 Transmisor y Receptor de Radio Control

Se utiliza un mando a distancia para transmitir de manera inalámbrica las señales de referencia (T_d, ϕ_d, θ_d, ψ_d) del usuario que pilotea el cuadrirrotor. Se utiliza un canal extra de este transmisor para cambiar entre modo manual (pilotado por el usuario) y modo automático (pilotado por el CPU a bordo). En el cuadrirrotor

se incorpora un receptor de radio control, el cual recibe las señales del mando a distancia y las pasa al módulo ESP32. En la Figura A.5 se puede ver un diagrama de la conexión de los principales componentes del prototipo.

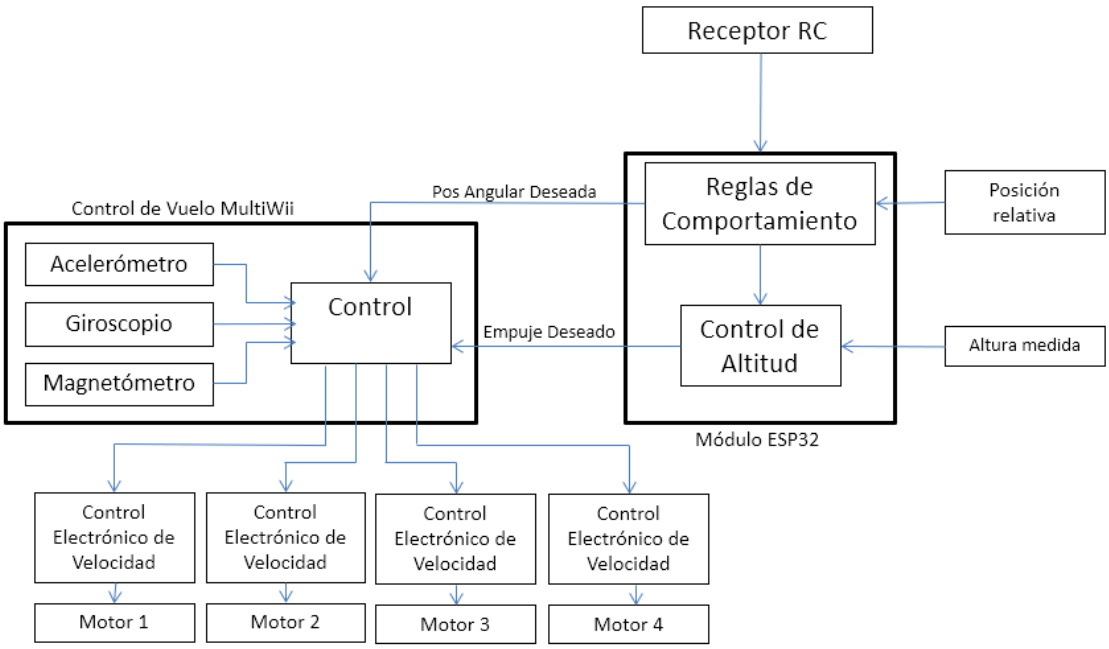


Figura A.5: Diagrama de conexión de componentes del prototipo

A.3 Identificación de Parámetros

Para implementar una plataforma de simulación que sea consistente con el sistema real, es necesaria la identificación de los parámetros usados en el modelo del cuadrirrotor [62, 63, 64]. Estos parámetros son la masa del cuadrirrotor, longitud del brazo, matrix de inercia, coeficiente de empuje y coeficiente de arrastre.

Una vez que se fabricó la estructura del cuadrirrotor y se ensamblaron todos los componentes, se utilizó una báscula para determinar la masa del cuadrirrotor. La longitud del brazo puede ser fácilmente medida con una regla o a partir del modelo CAD. Al agregar la masa de cada componente en el modelo CAD es posible obtener la matriz de inercia directamente del software de diseño 3D.

Para obtener los coeficientes de empuje k_T y de arrastre d, fue necesario construir un banco de pruebas que permita medir la velocidad del motor y el empuje generado por la hélice (ver Figuras A.6). Este banco de pruebas consiste en un brazo de palanca pivotado al centro, con un motor en un extremo y en el otro extremo apoyado sobre una báscula digital. Se utiliza además un sensor infrarrojo para medir la velocidad de giro de la hélice acoplada al motor. El giro de la hélice produce una fuerza de empuje que hace rotar el brazo de palanca aplicando un fuerza equivalente en el otro extremo apoyado en la báscula.

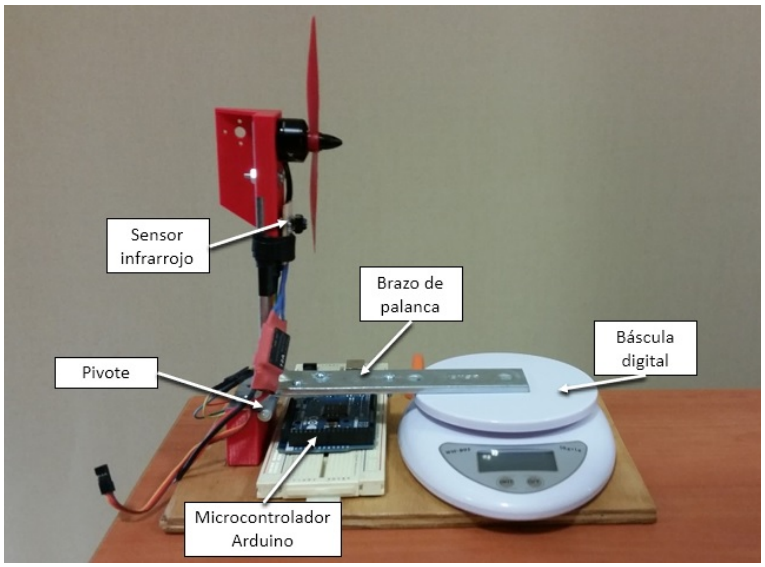


Figura A.6: Experimento para determinar coeficiente de empuje

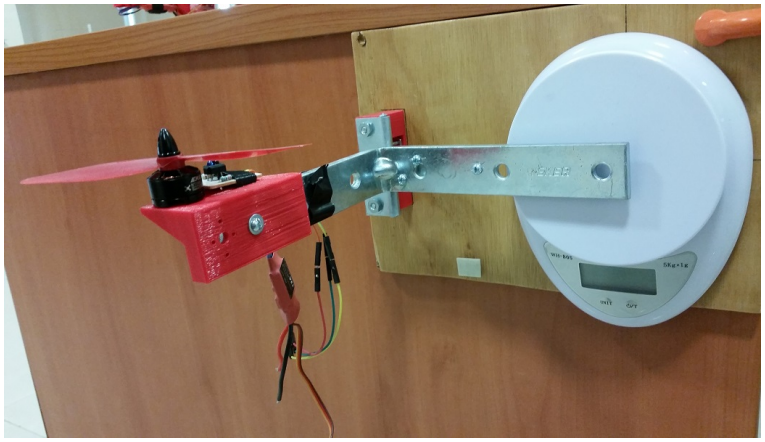


Figura A.7: Experimento para determinar coeficiente de arrastre

Obteniendo la velocidad angular del rotor ω_h, que genera un empuje suficiente

para mantener el cuadrirrotor en el aire, entonces podemos estimar el parámetro k_T como:

$$k_T = \frac{mg}{4\omega_h^2} \tag{A.1}$$

Para determinar el coeficiente de arrastre es necesario modificar la posición del motor y de todo el banco el pruebas como se observa en la Figura A.7. De esta forma la fuerza producida por el brazo de palanca es debida al momento de torsión generado por el giro de la hélice y d puede ser estimado como:

$$d = \frac{mgl}{\omega_h^2} \tag{A.2}$$

donde l es la longitud del brazo de palanca.

Los parámetros del cuadrirrotor Starling se muestran en la Tabla A.4.

Tabla A.4: Parámetros físicos del prototipo de cuadrirrotor

Parámetro	Valor	Unidades
m	0.42	Kg
l	0.11	m
I_{xx}	7.92×10^{-6}	Kg m^2
I_{yy}	7.92×10^{-6}	Kg m^2
I_{zz}	1.56×10^{-5}	Kg m^2
k_T	8.50×10^{-7}	N/(rad/s)^2
d	1.46×10^{-8}	N m /(rad/s)^2

Apéndice B

Código Simulador de Enjambre de Cuadrirrotores

```
1 ////////////////////////////////////////////////////
  // Simulador de enjambre de cuadrirrotores
3 // Mario Aguilera Ruiz
  // Doctorado en Ingenieria Electrica - FIME, UANL
5 ////////////////////////////////////////////////////

7 function result=simulate(TVeh,dR,dO,dA,dI,vmax)

9     //Constantes fisicas
      g = 9.81;                          //constante de gravedad
11    m = 0.42;                          //masa
      L = 0.11;                          //distancia centro rotor
13    k = 0.0000008;                     //coeficiente de empuje
      b = 0.0000000146;                  //coeficiente de arrastre
15    I = diag([7.92e-6, 7.92e-6, 1.56e-5]); //inercias
      kd = 0.25;                         //coeficiente de friccion
17
      //Tiempos de simulacion
19    tstart = 0;                      //tiempo inicial
      tend = 150;                       //tiempo final
21    dt = 0.05;                       //intervalo
```

```
    ts = [tstart:dt:tend];          //vector de tiempos
23  N = 1 + tend/dt;                //puntos en la simulacion

25  //Vectores unitarios con direccion de los sensores
     sens =  [0.923  0.382;
27              0.382  0.923;
                -0.382  0.923;
29              -0.923  0.382;
                -0.923  -0.382;
31              -0.382  -0.923;
                0.382  -0.923;
33              0.923  -0.382];

35  //Limites del area de simulacion
    x_min = 0;
37  x_max = 45;
    y_min = 0;
39  y_max = 45;

41 //   //Punto de influencia y obstaculos
   //   inf = [60 10 1];              //localizacion de influencia
43 //   obs = [14.5 10; 29.5 10];     //localizacion de obstaculos
   //   Mobs = size(obs);
45 //   Tobs = Mobs(1);               //total de obstaculos
   //   seeObs = 2;                   //rango de deteccion de obst
47 //   rObs = 0.5;                   //radio de los obstaculos

49  //Estado inicial del sistema
    x = ones(3,TVeh);
51  //Posiciones iniciales aleatorias
    for i = 1:TVeh
53    x(3,i) = 1;       //altura de 1 m
      if i == 1 then
55      x(1,i) = x_min + rand()*(x_max - x_min);
        x(2,i) = y_min + rand()*(y_max - y_min);
57    else
```

```
        while %T
59          safe = 0;
            x(1,i) = x_min + rand()*(x_max - x_min);
61          x(2,i) = y_min + rand()*(y_max - y_min);
            for j = 1:i-1
63            dother = sqrt((x(1,i)-x(1,j))^2 + (x(2,i)-x(2,j))^2);
              if dother > 0.72 then
65                safe = safe + 1;
              end
67          end
            if safe == j then
69            break;
            end
71        end
      end
73  end

75  xdot = zeros(3,TVeh);              //velocidades cartesianas
    theta = zeros(3,TVeh);             //posiciones angulares
77  theta(3,:) = rand(1,TVeh)*2*%pi; //orientacion aleatoria
    thetadot = zeros(3,TVeh);          //velocidades angulares
79  theta_d = zeros(3,TVeh);           //pos angulares deseadas
    theta_d(3,:)=theta(3,:);
81  zd = x(3,:);                       //altura deseada
    vd=zeros(3,TVeh);                  //velocidad deseada
83

85  flagi = 0;
    flagf = 1;
87  x0 = zeros(3,1);
    t0 = zeros(1,1);
89  dinf = zeros(1,TVeh);

91  //Estructura para guardar estado del controlador
    for i = 1:TVeh
93      controller_params(i) = struct('dt', dt, 'I', I, 'k', k,
```

```
        'L', L, 'b', b, 'm', m, 'g', g);
95      pc_state(i) = struct('flagi', flagi, 'flagf', flagf,
        'x0', x0, 't0', t0);
97    end


99

    for t=ts        //ciclo principal
101   ind = ind + 1;

103   for veh = 1:TVeh        //Para cada robot
        //variables de vecinos detectados, obstaculos e influencia
105     nR = 0;  nO = 0;  nA = 0;  nW = 0;
        DetR = zeros(1,8);
107     DetOx = 0;
        DetOy = 0;
109     detO = zeros(1,8);
        DetA = zeros(1,8);
111     Detinf = zeros(1,8);
        Detobs = zeros(1,8);
113
        ////////////////////////////////////////////////
115     ////////////////////////////////////////////////
        //Influencia
117     //Calcular distancia al punto de influencia
        dinf(veh) = sqrt((inf(1)-x(1,veh))^2 + (inf(2)-x(2,veh))^2);
119     //agregar ruido
        dinf(veh) = dinf(veh) + grand(1,1,'nor',0,0.03);
121     //transformar coordenadas globales a coordenas del robot i
        [Xinftr,Yinftr] = transform1(x(1,veh),x(2,veh),inf(1),inf(2),
123     theta(3,veh));
        //angulo del punto de influencia con respecto al robot i
125     be = atan(Yinftr,Xinftr);
        if be < 0 then
127         be = (2*%pi)+be;
        end
129     //Determinar el sensor que detecta la influencia
```

```
        for kk = 1:8
131         a1 = (kk-1)*%pi/4-0.218;
            a2 = (kk-1)*%pi/4+0.218;
133         if a1 < 0 then
                a1 = a1 + 2*%pi;
135         end
            if be >= a1 & gamatr <= a2 then
137             if dinf(veh) < dI then  //verificar si esta en rango
                    Detinf(1,kk) = 1;
139             end
            end
141     end
        //Calcular vector ddinf
143     ddinf = Detinf * sens2;
        ddinf = ddinf / (sqrt(ddinf(1)^2 + ddinf(2)^2) + 0.000001);
145     //transformar coordenadas del robot i a coordenadas globales
        [ddinfxtr,ddinfytr]=transform2(ddinf(1),ddinf(2),theta(3,veh));
147     Ainf = atan(ddinfytr,ddinfxtr);
        if Ainf < 0 then
149         Ainf = (2*%pi)+Ainf;
        end
151
        ////////////////////////////////////////////////
153     ////////////////////////////////////////////////
        //Obstaculos
155     if Tobs > 0 then
          for o = 1:Tobs
157         //calcular distancia al obstaculo
            dobs = sqrt((obs(o,1)-x(1,veh))^2 + (obs(o,2)-x(2,veh))^2);
159         //agragar ruido
            dobs = dobs + grand(1,1,'nor',0,0.03);
161         //transformar coordenadas globales a coordenadas del robot i
            [Xobtr,Yobtr] = transform1(x(1,veh),x(2,veh),obs(o,1),obs(o,2),
163         theta(3,veh));
            //angulo del obstaculo con respecto al robot
165         beobs = atan(Yobtr,Xobtr);
```

```
            if beobs < 0 then
167             beobs = beobs + 2*%pi;
            end
169         //Determinar el sensor que detecta el obstaculo
            for kk = 1:8
171             a1 = (kk-1)*%pi/4-0.218;
                a2 = (kk-1)*%pi/4+0.218;
173             if a1 < 0 then
                    a1 = a1 + 2*%pi;
175             end
                if beobs >= a1 & gamatr <= a2 then
177                 if dobs < seeObs + rObs then
                        nW = 1;
179                     Detobs(1,kk) = 1;
                    end
181             end
            end
183       end
        end
185
        //Detectar limites del area de simulacion
187
        for kk = 1:8
189         ang(kk) = atan(sens(kk,2),sens(kk,1)) + (theta(3,veh) - %pi/2);
            if ang(kk) >= 2*%pi then
191             ang(kk) = ang(kk) - 2*%pi;
            end
193         if ang(kk) < 0 then
                ang(kk) = ang(kk) + 2*%pi;
195         end
            newSens(kk,:) = [cos(ang(kk)), sin(ang(kk))];
197         limitX = x(1,veh) + (newSens(kk,1)*seeObs);
            limitY = x(2,veh) + (newSens(kk,2)*seeObs);
199         if limitX>x_max | limitX<x_min | limitY>y_max | limitY<y_min
                nW = 1;
201             Detobs(1,kk) = 1;
```

```
                //cambiar angulo de exploracion
203             Aexp(veh) = ang(kk)+(%pi/2)+(rand()*%pi);
            end
205     end
        //Calcular vector de direccion ddObs
207     ddObs1 = -Detobs * sens2;
        ddObs = ddObs1 / (sqrt(ddObs1(1)^2 + ddObs1(2)^2) +0.000001);
209     //Transformar coordenadas locales a coordenadas globales
        [ddObsxr,ddObsyr]=transform2(ddObs(1),ddObs(2),theta(3,veh));
211


213     ////////////////////////////////////////////////
        ////////////////////////////////////////////////
215     //Distancia a otros vehiculos
        for jj = 1:TVeh
217       if veh <> jj then
            dist = sqrt((x(1,jj)-x(1,veh))^2 + (x(2,jj)-x(2,veh))^2);
219         //agregar ruido
            dist = dist + grand(1,1,'nor',0,0.03);
221

            //posicion del robot jj con respecto al robot i
223         [Xtr,Ytr]=transform1(x(1,veh),x(2,veh),x(1,jj),x(2,jj),
            theta(3,veh));
225         //angulo del robot jj con respecto al robot i
            gamatr = atan(Ytr,Xtr);
227         if gamatr < 0 then
              gamatr = 2*%pi + gamatr;
229         end

231         //Determinar que sensor detecta al robot
            for kk=1:8
233           a1 = (kk-1)*%pi/4-0.218;
              a2 = (kk-1)*%pi/4+0.218;
235           if a1 < 0 then
                  a1 = a1 + 2*%pi;
237           end
```

```
            if gamatr >= a1 & gamatr <= a2 then
239             //Determinar la zona de comportamiento
                if dist <= (0.36+dR) then
241                 nR = nR + 1;
                    if DetR(1,kk)==0 | DetR(1,kk)>dist then
243                     DetR(1,kk)=dist;
                    end
245             end
                if dist > (0.36+dR) & dist <= (0.36+dR+dO) then
247                 nO = nO + 1;
                    detO(1,kk) = 1;
249                 DetOx = DetOx + cos(theta(3,jj));
                    DetOy = DetOy + sin(theta(3,jj));
251             end
                if dist > (0.36+dR+dO) & dist <= (0.36+dR+dO+dA) then
253                 if kk<=5 then
                      nA = nA + 1;
255                 end
                    if DetA(1,kk)==0 | DetA(1,kk)>dist then
257                   DetA(1,kk)=dist;
                    end
259             end
            end
261     end


263
      end //end of if veh <> jj then
265   end // end of for jj = 1:TVeh


267       ////////////////////////////////////////////////
          ////////////////////////////////////////////////
269           //Comportamiento de repulsion
      if nR >= 1 then
271           //Calculo de direccion deseada
          ddR1 = -DetR * sens3;
273       normaR=(sqrt(ddR1(1)^2 + ddR1(2)^2) +0.000001);
```

```
            ddR = ddR1 / normaR;
275             //Transformar direccion deseada a coordenadas globales
            [ddRxr,ddRyr]=transform2(ddR(1),ddR(2),theta(3,veh));
277         AR = atan(ddRyr,ddRxr);
            if AR < 0 then
279             AR = 2*%pi + AR;
            end
281
            target = -0.5*vmax/dR* normaR + vmax;
283
            if nW >= 1 then
285             xT = ddRxr + ddObsxr;
                yT = ddRyr + ddObsyr;
287         else
                xT = ddRxr;
289             yT = ddRyr;
            end
291         vd(:,veh) = target*[xT; yT; 0];

293     end

295         ////////////////////////////////////////////////////
            ////////////////////////////////////////////////////
297             //Comportamiento de orientacion
        if nR == 0 & nO >= 1 & nA == 0 then
299
            ddO = atan(DetOy,DetOx);
301         ddOx = cos(ddO);
            ddOy = sin(ddO);
303
            Aexpx = cos(Aexp(veh));
305         Aexpy = sin(Aexp(veh));

307         target = vmax;

309         if nW >= 1 then
```

```
                xT = ddOx + ddObsxr + Aexpx;
311             yT = ddOy + ddObsyr + Aexpy;
            else
313             xT = ddOx;
                yT = ddOy;
315         end

317         vd(:,veh) = target*[xT; yT; 0];
            theta_d(3,veh)=atan(yT,xT);
319
        end
321
        ////////////////////////////////////////////////////
323     ////////////////////////////////////////////////////

325     //Comportamiento de atraccion
        if nR == 0 & nO == 0 & nA >= 1 then
327         //Calculo de direccion deseada
            ddA1 = DetA * sens3;
329         normaA= (sqrt(ddA1(1)^2 + ddA1(2)^2) + 0.000001);
            ddA = ddA1 / normaA;
331         //Transformacion a coordenadas globales
            [ddAxr,ddAyr]=transform2(ddA(1),ddA(2),theta(3,veh));
333         AA = atan(ddAyr,ddAxr);
            if AA < 0 then
335             AA = 2*%pi + AA;
            end
337
            Aexpx = cos(Aexp(veh));
339         Aexpy = sin(Aexp(veh));

341         target = 0.5*vmax/dA* normaA + 0.5*vmax;

343         if nW >= 1 then
                xT = ddAxr + ddObsxr + Aexpx;
345             yT = ddAyr + ddObsyr + Aexpy;
```

```
            else
347             xT = ddAxr;
                yT = ddAyr;
349         end
            theta_d(3,veh) =  atan(yT,xT);
351         vd(:,veh) = target*[xT;yT;0.0]

353     end
        ////////////////////////////////////////////////////
355     ////////////////////////////////////////////////////

357     //Comportamiento de orientacion y atraccion
    if nR == 0 & nO >= 1 & nA >= 1 then
359
        ddO = atan(DetOy,DetOx);
361     ddOx = cos(ddO);
        ddOy = sin(ddO);
363
        ddA1 = DetA * sens3;
365     normaA= (sqrt(ddA1(1)^2 + ddA1(2)^2) + 0.000001);
        ddA = ddA1 / normaA;
367     [ddAxr,ddAyr]=transform2(ddA(1),ddA(2),theta(3,veh));
        AA = atan(ddAyr,ddAxr);
369     if AA < 0 then
            AA = (6.2832)+AA;
371     end

373     Aexpx = cos(Aexp(veh));
        Aexpy = sin(Aexp(veh));
375
        target = vmax;
377
        if nW >= 1 then
379         xT = ddOx + ddAxr + ddObsxr + Aexpx;
            yT = ddOy + ddAyr + ddObsyr + Aexpy;
381     else
```

```
                xT = ddOx + ddAxr;
383             yT = ddOy + ddAyr;
            end
385         theta_d(3,veh) =  atan(yT,xT);
            vd(:,veh)=target*[xT;yT;0.0]
387
        end
389         ////////////////////////////////////////////////////
            ////////////////////////////////////////////////////
391
            //Comportamiento de explorar
393     if nR == 0 & nO == 0 & nA == 0 then

395         Aexpx = cos(Aexp(veh));
            Aexpy = sin(Aexp(veh));
397
            target = vmax;
399
            if nW >= 1 then
401             xT = ddObsxr + Aexpx;
                yT = ddObsyr + Aexpy;
403         else
                xT = Aexpx;
405             yT = Aexpy;
            end
407
            theta_d(3,veh) = atan(yT,xT);
409         vd(:,veh)=target*[xT;yT;0.0]

411     end

413         ////////////////////////////////////////////////////
            ////////////////////////////////////////////////////
415
                //Detectar si angulo es mayor a 360 grados o negativo
417         if theta_d(3,veh) >= 2*%pi then
```

```
        theta_d(3,veh) = theta_d(3,veh) - 2*%pi;
    end
    if theta_d(3,veh) < 0 then
        theta_d(3,veh) = theta_d(3,veh) + 2*%pi;
    end
    ////////////////////////////////////////////////////
    ////////////////////////////////////////////////////
    //Control de velocidad cada 100 ms
    if modulo(t,0.1) == 0
        [theta_d(:,veh), pc_state(veh)] = speed_pid(pc_state(veh),
        vd(:,veh),xdot(:,veh),theta_d(3,veh),t)
    end
    //Control de posicion angular  y altitud
    [i(:,veh), controller_params(veh)] =
    pid_controller(controller_params(veh), thetadot(:,veh), theta(:,veh),
     xdot(:,veh), x(:,veh), zd(veh), theta_d(:,veh));

    //Calcular fuerzas, momentos y aceleraciones
    omega(:,veh) = thetadot2omega(thetadot(:,veh), theta(:,veh));
    a(:,veh) = acceleration(i(:,veh), theta(:,veh), xdot(:,veh),
     m, g, k, kd);
    omegadot(:,veh) = angular_acceleration(i(:,veh), omega(:,veh),
     I, L, b, k);

    //Actualizar estado del sistema
    omega(:,veh) = omega(:,veh) + dt*omegadot(:,veh);
    thetadot(:,veh) = omega2thetadot(omega(:,veh), theta(:,veh));
    theta(:,veh) = theta(:,veh) + dt*thetadot(:,veh);
    xdot(:,veh) = xdot(:,veh) + dt*a(:,veh);
    x(:,veh) = x(:,veh) + dt*xdot(:,veh);

    //Detectar si angulo es mayor a 360 grados
    if theta(3,veh) >= 6.2832 then
        theta(3,veh) = theta(3,veh) - 2*%pi;
    end
    if theta(3,veh) < 0 then
```

```
                theta(3,veh) = theta(3,veh) + 2*%pi;
455             end

457       end  //end of veh = 1:TVeh

459     end  //end of for t=ts

461     ////////////////////////////////////////////////////////////
        ////////////////////////////////////////////////////////////
463


465

    endfunction
467
    //Calcular empuje
469 function T = thrust(inputs, k)
        T = [0; 0; k * sum(inputs)];
471 endfunction

473 //Calcular momentos.
    function tau = torques(inputs, L, b, k)
475     tau = [
            L * k * (inputs(1) - inputs(3))
477         L * k * (inputs(2) - inputs(4))
            b * (inputs(1) - inputs(2) + inputs(3) - inputs(4))
479     ];
    endfunction
481
    //Calcular aceleracion en el marco de referencia inercial
483 function a = acceleration(inputs, angles, vels, m, g, k, kd)
        gravity = [0; 0; -g];
485     R = rotation(angles);
        T = R * thrust(inputs, k);
487     Fd = -kd * vels;
        a = gravity + 1 / m * (T + Fd);
489 endfunction
```

```
491 //Calcular aceleracion angular en el marco de referencia del cuadrirrotor
    function omegad = angular_acceleration(inputs, omega, I, L, b, k)
493     tau = torques(inputs, L, b, k);
        omegad = inv(I) * (tau - cross(omega, I * omega));
495 endfunction


497

    function omega = thetadot2omega(thetadot, angles)
499     phi = angles(1);
        theta = angles(2);
501     psi = angles(3);
        W = [
503         1, 0, -sin(theta)
            0, cos(phi), cos(theta)*sin(phi)
505         0, -sin(phi), cos(theta)*cos(phi)
        ];
507     omega = W * thetadot;
    endfunction
509


511 function thetadot = omega2thetadot(omega, angles)
        phi = angles(1);
513     theta = angles(2);
        psi = angles(3);
515     W = [
            1, 0, -sin(theta)
517         0, cos(phi), cos(theta)*sin(phi)
            0, -sin(phi), cos(theta)*cos(phi)
519     ];
        thetadot = inv(W) * omega;
521 endfunction


523

    //Calcular matriz de rotacion
525 function R = rotation(angles)
```

```
    phi = angles(3);
527 theta = angles(2);
    psi = angles(1);
529
    R = zeros(3,3);
531 R(:, 1) = [cos(phi) * cos(theta);
              cos(theta) * sin(phi);
533           - sin(theta)];
    R(:, 2) = [
535     cos(phi) * sin(theta) * sin(psi) - cos(psi) * sin(phi);
        cos(phi) * cos(psi) + sin(phi) * sin(theta) * sin(psi);
537     cos(theta) * sin(psi)
    ];
539 R(:, 3) = [
        sin(phi) * sin(psi) + cos(phi) * cos(psi) * sin(theta);
541     cos(psi) * sin(phi) * sin(theta) - cos(phi) * sin(psi);
        cos(theta) * cos(psi)
543 ];
endfunction
545
//Controlador PID de posicion angular y altitud
547 function [entrada, state] = pid_controller(state, thetadot,
 theta, xdot, x, zd, theta_d)
549 Kp = 30;
    Ki = 0.5;
551 Kd = 6;

553 //Iniciar integrar a cero
    if ~isfield(state, 'integral')
555     state.integral = zeros(3, 1);
        state.integral2 = zeros(3, 1);
557 end

559 //agregar ruido a las velocidades angulares
    thetadot_ruido=thetadot + grand(3,1,'nor',0,0.01);
561
```

```
    //Calcular empuje total
563 total = (state.g + Kd * (0.0-xdot(3)) + Kp * (zd-x(3)+
    grand(1,1,'nor',0,0.01))) * state.m / state.k /
565 (cos(state.integral(1)) * cos(state.integral(2)));

567 //Calcular errorers y entradas
    errpitch1 = theta(1) - theta_d(1);
569 if errpitch1 >= 0 then
        errpitch2 = errpitch1 - 6.2832;
571 else
        errpitch2 = 6.2832 + errpitch1;
573 end
    if abs(errpitch1) <= abs(errpitch2) then //elegir rotacion mas corta
575     errpitch = errpitch1;
    else
577     errpitch = errpitch2;
    end
579
    errroll1 = theta(2) - theta_d(2);
581 if errroll1 >= 0 then
        errroll2 = errroll1 - 6.2832;
583 else
        errroll2 = 6.2832 + errroll1;
585 end
    if abs(errroll1) <= abs(errroll2) then //elegir rotacion mas corta
587     errroll = errroll1;
    else
589     errroll = errroll2;
    end
591
    erryaw1 = theta(3) - theta_d(3);
593 if erryaw1 >= 0 then
        erryaw2 = erryaw1 - 6.2832;
595 else
        erryaw2 = 6.2832 + erryaw1;
597 end
```

```
    if abs(erryaw1) <= abs(erryaw2) then //elegir rotacion mas corta
599     erryaw = erryaw1;
    else
601     erryaw = erryaw2;
    end
603
    errangle = [errpitch; errroll; erryaw];
605 err = Kd*thetadot_ruido + Kp*errangle + Ki*state.integral2;

607 entrada = err2inputs(state, err, total);

609 //Actualizar estado del controlador
    state.integral = state.integral + state.dt .* thetadot;
611 state.integral2 = state.integral2 + state.dt .* state.integral;
endfunction
613

615 //Calcular las entradas requeridas por el sistema
function inputs = err2inputs(state, err, total)
617 e1 = err(1);
    e2 = err(2);
619 e3 = err(3);
    Ix = state.I(1, 1);
621 Iy = state.I(2, 2);
    Iz = state.I(3, 3);
623 k = state.k;
    L = state.L;
625 b = state.b;

627 inputs = zeros(4, 1);
    inputs(1) = total/4 -(2 * b * e1 * Ix + e3 * Iz * k * L)/(4 * b * k * L);
629 inputs(2) = total/4 + e3 * Iz/(4 * b) - (e2 * Iy)/(2 * k * L);
    inputs(3) = total/4 -(-2 * b * e1 * Ix + e3 * Iz * k * L)/(4 * b * k * L);
631 inputs(4) = total/4 + e3 * Iz/(4 * b) + (e2 * Iy)/(2 * k * L);

633 endfunction
```

```
635 ////////////////////////////////////////////////////////////////////////////
    ////////////////////////////////////////////////////////////////////////////
637

639 //Controlador  PID de velocidad
    function [theta_d, state] = speed_pid(state,vd,xdot,psid,t)
641
        if ~isfield(state, 'esum')
643         state.esum = zeros(3, 1);
        end
645     if ~isfield(state, 'epre')
            state.epre = zeros(3, 1);
647     end
        state.t0 = t;
649
        Kp1=1.75;
651     Kd1=0.1;
        Ki1=0.16;
653
        ev = vd - xdot;
655
        rxc = Kp1*ev(1,1) + Ki1*state.esum(1,1) + Kd1*state.epre(1,1);
657     ryc = Kp1*ev(2,1) + Ki1*state.esum(2,1) + Kd1*state.epre(2,1);

659     state.esum = state.esum + ev;
        state.epre = ev;
661
        if state.esum(1,1)>100 then
663         state.esum(1,1)=0;
        end
665     if state.esum(2,1)>100 then
            state.esum(2,1)=0;
667     end

669     thetac = 1/g * (rxc*cos(psid)+ ryc*sin(psid));
```

```
    phic =   1/g * (rxc*sin(psid)- ryc*cos(psid));
671
    theta_d=[phic; thetac; psid];
673
endfunction
675

677 //Transformar coordenadas globales a sist coordenado del robot
function [xtr,ytr] = transform1(xi,yi,xj,yj,ang)
679     xtr = (xj-xi)*sin(ang) - (yj-yi)*cos(ang);
    ytr = (xj-xi)*cos(ang) + (yj-yi)*sin(ang);
681 endfunction

683 //Transformar coordenas locales a  coordenadas globales
function [ddxtr,ddytr] = transform2(dd1,dd2,ang)
685     ddxtr = dd1*cos(ang-%pi/2) - dd2*sin(ang-%pi/2);
    ddytr = dd1*sin(ang-%pi/2) + dd2*cos(ang-%pi/2);
687 endfunction
```