

UNIVERSIDAD DEL VALLE DE GUATEMALA  
Facultad de Ingeniería



**Optimización de un algoritmo de inteligencia de enjambre  
enfocado en sincronización y control de formaciones de  
sistemas robóticos multi-agente para escenarios con obstáculos  
móviles**

Protocolo de trabajo de graduación presentado por Gerardo Paz  
Fuentes, estudiante de Ingeniería Mecatrónica

Guatemala,

2024

## **Resumen**

El presente protocolo presenta una propuesta para realizar la optimización de un algoritmo de inteligencia de enjambre y control de formaciones de sistemas robóticas multi-agente desarrollado en trabajos anteriores. El algoritmo será validado con plataformas robóticas móviles como el Pololu 3Pi+ disponible en el Robotat de la Universidad del Valle de Guatemala en escenarios controlados con obstáculos móviles.

Para llevar a cabo esta fase, se tomarán en cuenta las recomendaciones de los trabajos anteriores. Además, se buscará las deficiencias y puntos de mejora en la implementación del algoritmo para su ejecución en un ambiente físico de manera que se reduzcan los tiempos de ejecución y obtener un mejor rendimiento del mismo.

El objetivo final de esta fase es mejorar la implementación del algoritmo para que se pueda seguir explorando su alcance al utilizarse en robótica de enjambre con escenarios más complejos y dinámicos. Con esto se busca poder recrear situaciones similares a la realidad como operaciones de búsqueda y rescate, mapeo de entornos, simulación de comportamientos biológicos e incluso utilizar nuevas plataformas robóticas.

## **Antecedentes**

### **Robótica de enjambre**

La robótica de enjambre consiste en el uso de robots relativamente sencillos que, al organizarse y funcionar en conjunto, pueden llevar a cabo tareas complejas que un solo robot no podría realizar. Esto proporciona soluciones flexibles, optimizadas y de un menor costo. Además, no se requiere un número específico de agentes robóticos ya que pueden ir desde las dos unidades hasta miles de ellas [1].

Algunas de las aplicaciones de la robótica de enjambre son el control de tráfico, realizar formaciones en movimiento, misiones de búsqueda y rescate, mapeo de entornos, simulación de comportamientos biológicos, exploración de zonas y comunicación de rutas.

Estas últimas dos han sido estudiadas a profundidad para su implementación en la cosecha de la fresa donde el enjambre realiza la exploración de una zona de cultivo, analiza el estado de los frutos por medio de imágenes computacionales para determinar el momento óptimo para su cosecha y comunica los datos procesados para realizar una cosecha automatizada [2].

### **Robótica de enjambre inspirada en peces**

En 2021 un equipo de investigadores del instituto Wyss de Harvard y la Escuela de Ingeniería y Ciencias Aplicadas John A. Paulson (SEAS) desarrollaron un robot llamado Bluebot inspirado en un pez. Este cuenta con dos cámaras y tres luces LED para su sistema de visión guiado.

Los investigadores模拟aron una misión de búsqueda con una luz roja intermitente e implementaron un algoritmo de dispersión. Para iniciar, los Bluebots se dispersaron en el tanque y una vez que un robot detectó la luz roja, sus leds comenzaron a parpadear, lo que activó el algoritmo de agregación al resto de robots que lograron sincronizar sus movimientos y agruparse al rededor del robot que detectó la luz, imitando a un banco de peces real [3].

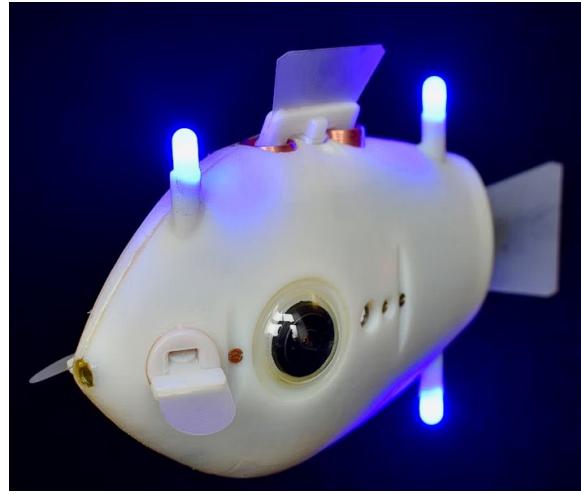


Figura 1: Prototipo físico de Bluebot[3].

## Robotarium de Georgia Tech

En el Instituto Tecnológico de Georgia se ha desarrollado el proyecto Robotarium [4]. Se realizó para proveer acceso gratuito a una plataforma de robótica de enjambre a la que pueda acceder cualquier persona del mundo. Lo único que se necesita para experimentar con la plataforma es descargar un simulador en Matlab o Python, registrarse en la página de Robotarium y esperar la aprobación por parte de un administrador para realizar el experimento.



Figura 2: Laboratorio Robotarium de Georgia Tech [4].

## Robotat de la Universidad del Valle de Guatemala

En el Centro de Innovación y Tecnología de la Universidad del Valle de Guatemala, se encuentra una plataforma de robótica para experimentación llamada Robotat la cual

está inspirada en el Robotarium del Instituto Tecnológico de Georgia. Esta se conforma de una plataforma de acero y pycem con un espacio útil de  $5 \times 5 \times 3$  m, capaz de soportar cargas puntuales de hasta dos toneladas. También cuenta con un sistema de captura de movimiento de OptiTrack, compuesto de seis cámaras Prime<sup>x</sup> 41 de alta precisión y baja latencia para realizar experimentos en tiempo real. Además, el sistema funciona con una red local inalámbrica WiFi a través de la cual se realiza la comunicación entre robots [5].

Para el funcionamiento del sistema OptiTrack, se utilizan “marcadores” que son figuras plásticas con reflectivos. Estos permiten reflejar la luz infrarroja que emiten las cámaras para obtener, de manera precisa, la posición y el movimiento de objetos en un espacio tridimensional.



Figura 3: Plataforma Robotat de la Universidad del Valle de Guatemala [6].

### Validación de algoritmos de algoritmos de *Particle Swarm Optimization* (PSO) y *Ant Colony Optimization* (ACO)

En la Universidad del Valle de Guatemala, la tesis de Jonathan Menéndez [7] se enfocó en realizar pruebas físicas para validar los algoritmos de robótica de enjambre PSO y ACO utilizando los robots móviles Pololu 3pi+ en la plataforma de Robotat.

Luego de realizar múltiples pruebas físicas, se encontró que el algoritmo de ACO es capaz de generar trayectorias de manera satisfactoria en el ecosistema del Robotat, respetando las limitaciones físicas de espacio en la mesa para no colisionar y evitar que las cámaras de OptiTrack pierdan la detección de los agentes. Además, este algoritmo demostró tener una mayor eficiencia al corregir las diferencias entre el nodo de inicio y la posición inicial del robot al orientarlo hacia el punto de inicio.

Dichos estudios se limitaron al espacio disponible en la mesa de la plataforma del Robotat, a un espacio libre de obstáculos y a la implementación de solo diez Pololu 3pi+ debido a la disponibilidad de equipo.

## **Algoritmo de sincronización y control de sistemas de robots multiagente para misiones de búsqueda**

El trabajo de investigación de Andrea Maybell Peña [8] se basó en utilizar un sistema de robots multi-agente para realizar misiones de búsqueda. El algoritmo se basa en la teoría de grafos y el control moderno para tener formaciones específicas de los agentes que permiten su movilización a través de obstáculos que se limitaron a una geometría toroidal y la cantidad de agentes robóticos se limitó a diez unidades del modelo E-Puck de GCtronic.

Se realizó la implementación del algoritmo en el simulador Webots de Cyberbotics. Como resultado se obtuvo una tasa de éxito del 80 % utilizando el algoritmo completo para llegar a la meta.



Figura 4: Simulación en Webots implementando el algoritmo para evadir obstáculos [6].

## **Validación de un algoritmo de inteligencia de enjambre enfocado en sincronización y control de formaciones de sistemas robóticos multi-agente en un entorno físico**

En el trabajo de investigación de Alejandro Rodríguez [9] se realizó una validación del algoritmo de inteligencia de enjambre enfocado a sincronización vertical y control de formaciones de sistemas robóticos multi-agente. La validación se realizó utilizando el ecosistema del Robotat de la Universidad del Valle de Guatemala y los robots Pololu 3pi+ modificados.

Para la validación física, se realizaron diversos experimentos donde se evaluó el desempeño de cada agente, la generación de trayectorias, el posicionamiento de los agentes, las distintas configuraciones de formación y los escenarios utilizando obstáculos.

Los experimentos realizados se limitaron a utilizar un máximo de nueve agentes robóticos debido a su disponibilidad en la universidad, el tiempo máximo por semana para realizar pruebas en el Robotat fue de seis horas y estas se realizaron utilizando obstáculos estáticos.

Los resultados de la experimentación demostraron que, el algoritmo evade los obstáculos satisfactoriamente y mantiene una distancia adecuada entre agentes. Sin embargo, el algoritmo en simulación es aproximadamente un 70 % más rápido que en físico. Esto se debe a que cada prueba física tomaba alrededor de 8 minutos debido a la latencia del servidor que aumentaba con el número de agentes conectados.

## **Justificación**

La inteligencia de enjambre es un campo reciente en el área de inteligencia artificial. Sus aplicaciones abarcan desde realizar tareas cotidianas hasta resolver problemas complejos. Uno de sus mayores beneficios es utilizar agentes robóticos de bajo costo para ejecutar acciones en conjunto en lugar de utilizar un solo robot complejo. Además, esto crea un sistema robótico robusto lo que resulta útil para diversas aplicaciones.

Se decidió estudiar la robótica de enjambre ya que en la Universidad del Valle de Guatemala, en trabajos anteriores, se han realizado pruebas implementando algoritmos de sincronización y control utilizando simuladores como Webots, así como agentes físicos Pololu 3Pi+. Estas pruebas tuvieron éxito, sin embargo, estuvieron limitadas a generar trayectorias en un ambiente con obstáculos fijos y los tiempos de ejecución eran muy largos. Por lo tanto, en este trabajo de graduación se desea optimizar el algoritmo desarrollado previamente y realizar nuevas pruebas para validar las trayectorias generadas en un ambiente controlado con obstáculos móviles.

Esto permitirá conocer el alcance de los algoritmos de sincronización y control, además que abrirá las puertas a su implementación en aplicaciones de la vida real como misiones de búsqueda y rescate o análisis de zonas de cultivos, donde en los escenarios se encontrarán obstáculos móviles como personas, animales o incluso cambios en el propio escenario debido a factores externos.

## **Objetivos**

### **Objetivo General**

Optimizar la implementación del algoritmo de inteligencia de enjambre enfocado en sincronización y control de formaciones de sistemas robóticos multi-agente desarrollado anteriormente, y validarla en escenarios con obstáculos móviles, en el ecosistema Robotat.

### **Objetivos Específicos**

- Evaluar la implementación del algoritmo de sincronización y control desarrollado anteriormente e identificar deficiencias y puntos de mejora en código, lenguaje de programación y métodos de comunicación.
- Optimizar el algoritmo tomando en cuenta los puntos de mejora identificados y evaluar su rendimiento en escenarios similares a los utilizados anteriormente.
- Adaptar el algoritmo para generar trayectorias en escenarios con obstáculos móviles.
- Validar el rendimiento del algoritmo optimizado con agentes robóticos como el Pololu 3Pi+ en el ecosistema Robotat.

## Marco teórico

### Conceptos fundamentales en robótica de enjambre

#### Enjambre

En la robótica, un enjambre es el conjunto de individuos que colaboran entre sí para lograr un objetivo. En la Figura 5 se muestra un enjambre de robots.

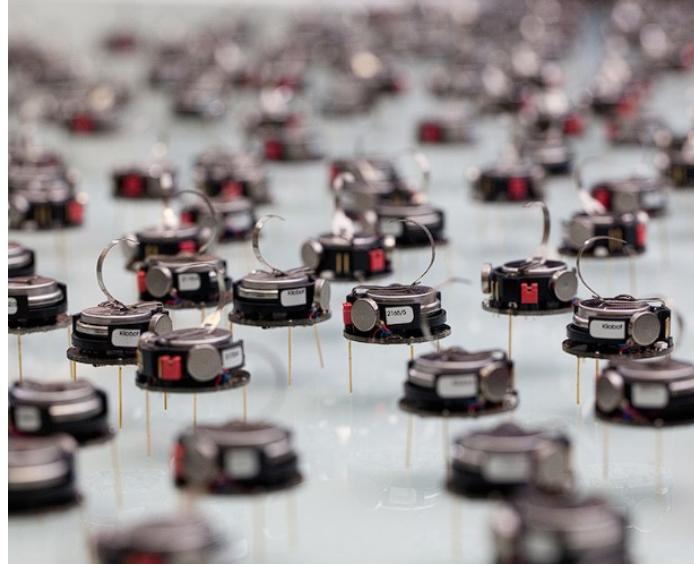


Figura 5: Ejemplo de enjambre de robots [10].

#### Agente

Se le conoce como agente a un individuo que forma parte del enjambre. Este es capaz de realizar acciones simples para lograr un objetivo de forma colaborativa con los demás individuos. En la robótica de enjambre, los agentes son robots [11].

#### Formaciones

Es común que en la naturaleza se observen enjambres que forman patrones. Esto es gracias a que los individuos realizan movimientos coordinados para crear formaciones. En la robótica de enjambre sucede lo mismo, se tiene un conjunto de agentes que se organizan y coordinan sus movimientos para lograr objetivos. Estos pueden ser evadir obstáculos, mapear un entorno, llegar a un objetivo siguiendo una trayectoria, entre otros. Para lograr la coordinación de formaciones, se requiere aplicar un control que puede ser centralizado o descentralizado [11].

## Control centralizado y descentralizado

En la robótica de enjambre, el control centralizado consiste en una red de comunicación donde una unidad central de procesamiento (CPU) tiene comunicación con cada agente para transmitir y recibir información. Asimismo, cada agente se comunica únicamente con el CPU.

El control descentralizado, también llamado control distribuido, es donde cada agente tiene comunicación con los demás para transmitir información. Este tipo de control requiere un protocolo de comunicación más robusto y complejo, sin necesidad de un CPU [12].

En la Figura 6 se observa la diferencia entre ambos tipos de control.

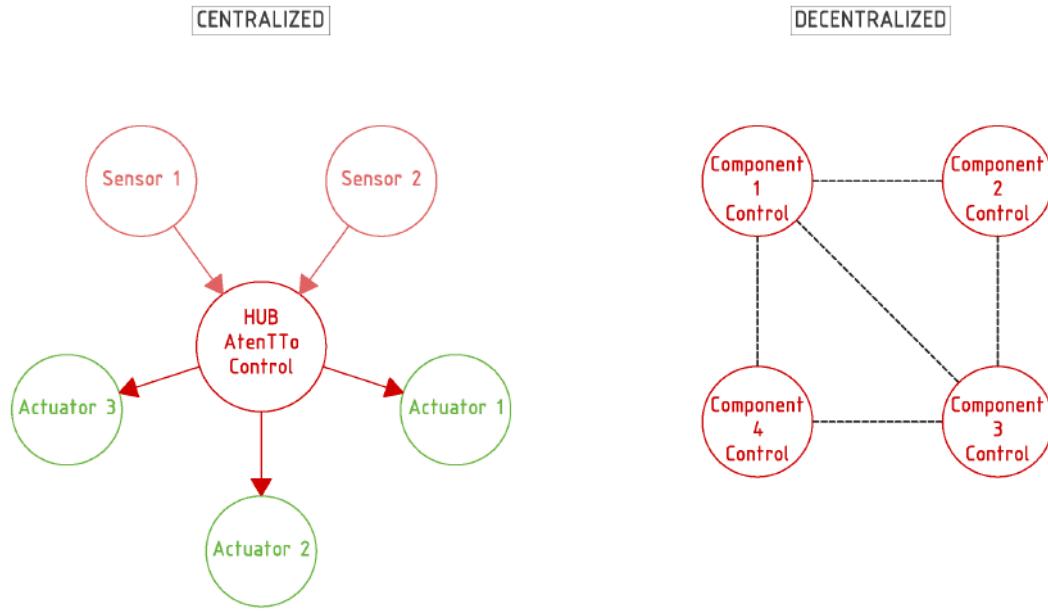


Figura 6: Ejemplo de un control centralizado y descentralizado [12].

## Conceptos básicos en Teoría de grafos

### Teoría de grafos

La teoría de grafos es una rama de la matemática y ciencias de la computación que se dedica a estudiar los grafos [13]. El grafo es un conjunto de vértices conectados por aristas tal como se observa en la Figura 7. Esta teoría se utiliza en diversas aplicaciones como mapeo de entornos, generación de rutas, análisis de datos, telecomunicaciones, entre otras. En este trabajo de graduación, se utilizó la teoría de grafos para definir formaciones y la red de comunicación entre agentes.

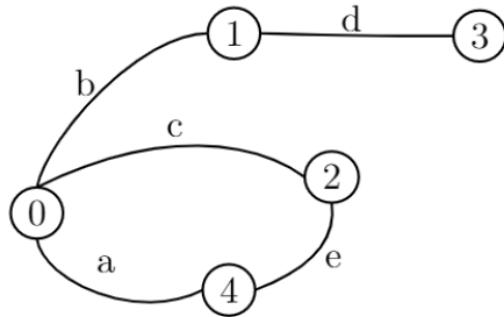


Figura 7: Ejemplo de un grafo [8].

### Vértices y aristas

Los vértices, también llamados nodos, son los puntos donde se conectan las aristas de un grafo. El grado de un vértice será el número de aristas asociados a él [13].

Las aristas, también llamadas arcos o enlaces, son las líneas que conectan un par de vértices y se clasifican en [13]:

- Lazo: Es una arista que tiene en sus extremos el mismo vértice.
- Paralelas o múltiples: Es cuando dos o más aristas tienen en los extremos el mismo par de vértices.

### Tipos de grafos

En un grafo, los vértices se puede conectar de distintas maneras según la aplicación que se requiera tal como se observa en la Figura 8. Algunas de sus clasificaciones son [14]:

- Simple o anillo: Cada par de nodos se conecta por una sola arista.
- Multigrafo: En cada par de nodos puede haber más de un enlace.
- Árbol: Es un grafo que no tiene ciclos.
- Dirigidos o dígrafos: Las aristas entre los nodos tienen dirección.
- Estrella: Tiene un nodo central que conecta con los demás nodos.
- No conexos: Hay uno o más nodos que no conectan con el resto.
- Completo: Cada nodo está conectado con los demás.
- Regular: Todos los nodos tienen el mismo número de aristas.

- De celosía y mundo pequeño: Estos grafos se utilizan para entender el comportamiento y estructura de ciertos fenómenos en la naturaleza.

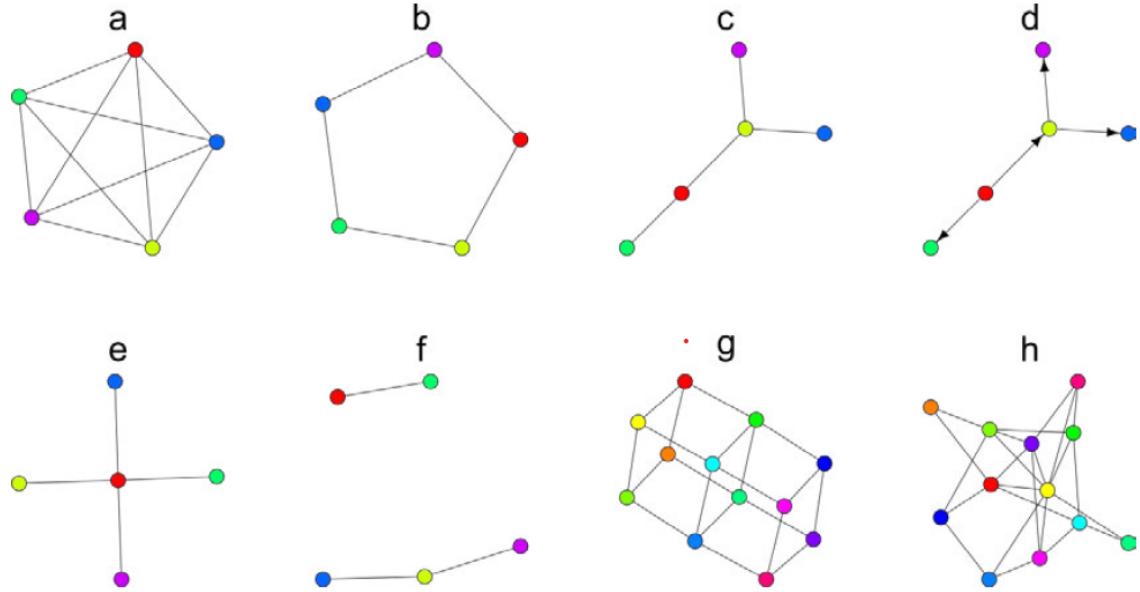


Figura 8: Ejemplo de algunos tipos de grafos. a: completo, b: simple, c: árbol, d: dígrafo, e: estrella, f: no conexo, g: de celosía, h: de mundo pequeño [14].

### Matrices asociadas a un grafo

La representación gráfica de un grafo es poco práctica para su análisis, por esto es mejor utilizar su representación matricial [15]. A continuación, se mencionan algunas matrices útiles y su implementación para el grafo de la Figura 7.

- Matriz de incidencia ( $I$ ): Se tiene una matriz de  $v$  vértices por  $e$  aristas. El elemento  $a_{ve} = 1$  si la arista conecta con el vértice, de lo contrario  $a_{ve} = 0$

$$I = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

- Matriz de adyacencia ( $A$ ): El grafo se representa con una matriz cuadrada de tamaño  $v \times v$ . Si hay una arista entre el vértice  $v_1$  y  $v_2$ , el elemento  $a_{v_1 v_2} = 1$ .

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

- Matriz de grados (D): Se tiene una matriz diagonal de tamaño  $v \times v$  que contiene los grados de cada vértice.

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

- Matriz laplaciana (L): Esta matriz es resultado de operar otras matrices asociadas al grafo. Para grafos no dirigidos, esta se calcula como  $L = D - A$ . Para grafos dirigidos se calcula como  $L = II^T$ .

$$l = \begin{bmatrix} 3 & -1 & -1 & 0 & -1 \\ -1 & 2 & 0 & -1 & 0 \\ -1 & 0 & 2 & 0 & -1 \\ 0 & -1 & 0 & 1 & 0 \\ -1 & 0 & -1 & 0 & 2 \end{bmatrix}$$

- Matriz de rigidez (K): Esta matriz surge de qué tanto es posible deformar el grafo sin doblar ni modificar la longitud de las aristas [16]. A continuación se muestra la matriz de adyacencia totalmente rígida.

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

Esta corresponde a la matriz de adyacencia del grafo completo.

## Control de la formación

Para resolver el problema de control de formaciones se utilizan dos grafos asociados. El primero es el grafo de formación. Este es un grafo no dirigido que contiene la configuración deseada. Además, se utiliza un grafo ponderado donde las longitudes de las aristas representan la distancia entre los agentes. El segundo grafo es el que representa la red de comunicación entre agentes. Este es un grafo dirigido donde las aristas están definidas por la dinámica de lazo cerrado del sistema multi-agente [8].

## Grafo de formación

Para definir un grafo de formación se debe entender el concepto de rigidez. Una estructura es rígida cuando todas las distancias entre los vértices están definidas y se mantienen constantes. Para evaluar la rigidez de un grafo se debe cumplir  $e = 2v - 3$ , donde  $v$  es el número de vértices y  $e$  el número de aristas. Si un grafo tiene menos aristas que vértices, este ya no se considera rígido [17].

Con esto, se introduce el término de un grafo mínimamente rígido. Para que un grafo se considere mínimamente rígido, debe tener un estado donde si se elimina cualquier arista, el grafo deja de ser rígido.

La ventaja de trabajar con grafos mínimamente rígidos es que no tienen restricciones innecesarias a la hora de mantener la formación.

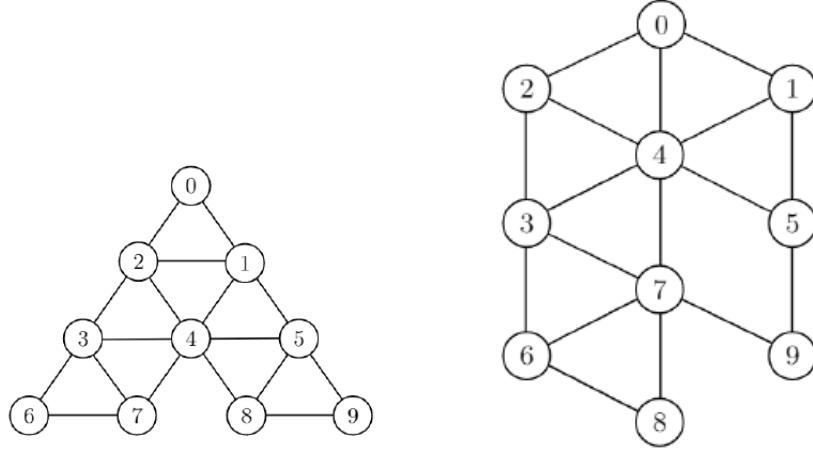


Figura 9: Grafos de formación en triángulo y hexágono [8].

## Construcción de un grafo mínimamente rígido

El método de Henneberg es utilizado para construir grafos mínimamente rígidos donde cada vértice mantiene dos grados de libertad [17]. Los pasos para realizar el método son:

1. Numerar todos los vértices.
2. Agregar una arista entre el vértice 1 y el vértice 2.
3. Agregar los demás vértices a la estructura utilizando dos aristas.
4. Verificar que se cumple la condición de  $e = \frac{v^2-v}{2}$

## Grafo para la red de comunicación

La comunicación entre agentes se representa por un grafo de comunicación. Este es un dígrafo que indica cuáles agentes tienen comunicación entre ellos y hacia qué dirección se

comunican [18]. A continuación se muestran tres tipos de redes.

- Red estática: Las aristas se mantienen invariantes en el tiempo.
- Red dinámica o dependiente del estado: Las aristas son variantes en el tiempo y pueden desaparecer o aparecer según el estado de la red de agentes.
- Red aleatoria: La existencia de una arista se da mediante una distribución de probabilidad.

## Teoría de control

En la robótica, se utilizan sistemas de control que integran varios subsistemas y procesos para estabilizar una respuesta inestable. Esto se logra mediante la implementación de un controlador que manipula las entradas del sistema para obtener la salida deseada [19]. En la Figura 10 se muestra el sistema de control más utilizado.

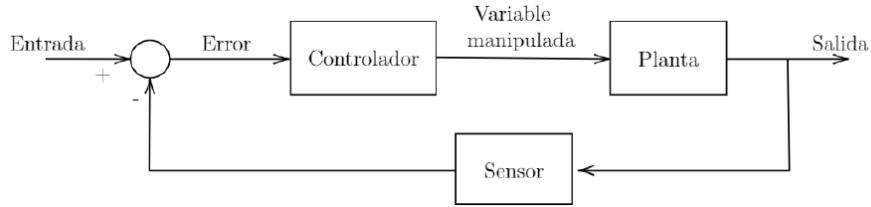


Figura 10: Sistema de control en lazo cerrado [19].

## Control de formación

Para realizar el control de formaciones se requieren dos niveles de control, uno superior y otro inferior. El control de capa superior maneja el comportamiento de los agentes y sus posiciones. El control de capa inferior controla la velocidad de las ruedas en cada agente [8].

## Cinemática de robots diferenciales con ruedas

Para la implementación física en la robótica de enjambre se requiere un modelo de movimiento que contemple las dimensiones y características físicas del robot como se observa en la Figura 11. Para esto se tienen las ecuaciones (1) y (2). Donde  $\ell$  es la distancia del motor hasta su centro,  $\varphi$  es el ángulo de orientación del uniciclo dentro del plano  $XY$ ,  $v$  es la velocidad lineal del robot,  $w$  es la velocidad angular de las ruedas y  $r$  es el radio de las ruedas [20].

$$v = \frac{r(\dot{\varphi}_R + \dot{\varphi}_L)}{2} \quad (1)$$

$$w = \frac{r(\dot{\varphi}_R + \dot{\varphi}_L)}{2\ell} \quad (2)$$

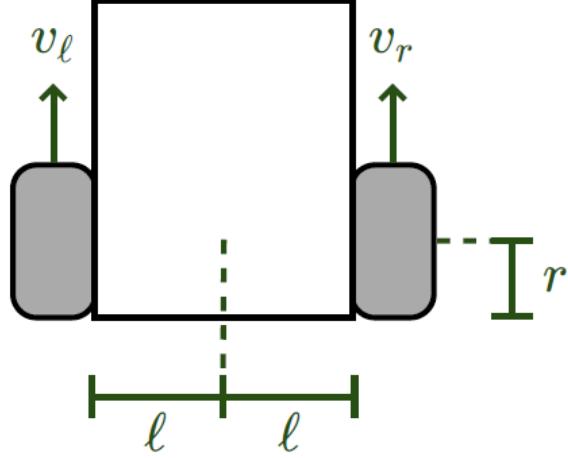


Figura 11: Modelo de uniciclo [20].

Adicional a esto, se puede calcular la velocidad controlada de la rueda derecha y la rueda izquierda con las ecuaciones (3) y (4), respectivamente.

$$\dot{\varphi}_{R,crtl} = \frac{v_{ctrl} + \ell w_{ctrl}}{r} \quad (3)$$

$$\dot{\varphi}_{L,crtl} = \frac{v_{ctrl} - \ell w_{ctrl}}{r} \quad (4)$$

### Otras ecuaciones matemáticas relevantes

Para mantener una correcta formación y distribución del enjambre, se necesitan herramientas como la ecuación de consenso y la función de tensión.

#### Ecuación de consenso

La ecuación 5 se utiliza para mantener la formación de los agentes en la posición asignada. Esta toma en cuenta el centro de masa de la formación y calcula la velocidad de cada agente para mantener la forma del grafo [8].

$$v_i(t) = \sum_{j \in N(i)} (x_j(t) - x_i(t)), i = 1, 2, \dots, n \quad (5)$$

Donde  $N(i)$  es el conjunto de unidades adyacentes de la unidad  $i$  en la red multi-agente.

Luego de añadir los pesos, se obtiene la ecuación 6:

$$\frac{\partial \varepsilon_{i,j}}{\partial x_i} = w_{i,j}(\|x_i - x_j\|)(x_i - x_j) \quad (6)$$

Donde de puede despejar el peso  $w_{i,j}$ .

### Funciones racionales para hallar la tensión

Para modificar la ecuación de consenso es necesario hallar la tensión utilizando funciones racionales que toman en cuenta las distancias deseadas y las distancias restringidas [8].

- Modelo 1: Combinación aditiva del control de formación y evasión de obstáculos.

$$\epsilon_{ij} = \frac{1}{2}(\|x_i - x_j\| - d_{ij})^2 + \frac{\|x_i - x_j\|^2}{\|x_i - x_j\| - r} \quad (7)$$

Donde,  $d_{ij}$  es la distancia entre los agentes  $i$  y  $j$ , y  $r$  es el radio de los agentes.

- Modelo 2: Combinación de control de formación, evasión de obstáculos y mantenimiento de la conectividad.

$$\epsilon_{ij} = \frac{(\|x_i - x_j\| - d_{ij})^2}{(\|x_i - x_j\| - r)(\|x_i - x_j\| - R)} \quad (8)$$

Donde,  $R$  es la distancia máxima que se pueden alejar los agentes sin salirse del rango del radar de los otros.

- Modelo 3: Combinación de control de formación y evasión de obstáculos.

$$\epsilon_{ij} = \frac{2(\|x_i - x_j\| - d_{ij})^2}{\|x_i - x_j\| - r} \quad (9)$$

- Modelo 4: Modelo dinámico con control de formación y evasión de obstáculos.

Se utiliza un modelo dinámico que al inicio, cuando los agentes comienzan en posiciones aleatorias, se utiliza únicamente el control para evitar colisiones. Luego, cuando los agentes están lo suficientemente cerca sin chocarse, se cambia al modelo de la ecuación 9 que toma en cuenta el control de la formación

- Modelo 5: Modelo dinámico con control de formación usando coseno hiperbólico y evasión de obstáculos.

$$\epsilon_{ij} = 0.01 \cosh(1.8\|x_i - x_j\| - 8.4) \quad (10)$$

Se utilizó el coseno hiperbólico ya que es una función “plana” que permite que el control de formación sea el que decida dónde deben posicionarse los agentes en caso de tener un mínimo erróneo en una distancia de 0.

- Modelo 6: Modelo dinámico con control de formación usando coseno hiperbólico y evasión de obstáculos incluyendo límites de velocidad.

Esta modificación no afecta directamente a la ecuación. Se realiza después de esta y consiste en incluir un límite de velocidad al modelo de la ecuación 10. Por lo tanto, si la velocidad obtenida con el modelo es mayor al límite establecido, solo se tomará en cuenta la dirección.

## Herramientas de software

Para realizar los cálculos, el control y las pruebas con los agentes robóticos, se necesita utilizar software especializado como los siguientes:

### Matlab

Matlab es una plataforma de programación desarrollada por MathWorks con su propio lenguaje especializado para aplicaciones de ingeniería, análisis de datos, generación de algoritmos, entre otras [21].

Para el presente trabajo, se utiliza Matlab como la fuente de procesamiento de datos y generación de trayectorias para el algoritmo de sincronización y control de formaciones.

### Entorno de simulación Webots

Webots es un software de código abierto creado por Cyberbotics [22]. Esta plataforma se centra en la simulación de escenarios que permiten replicar situaciones reales con distintos tipos de robots y objetos tal como se muestra en la Figura 12. Varios de los modelos disponibles en Webots están calibrados para comportarse como el modelo real y el usuario puede modificar los parámetros para garantizar una simulación realista.

Webots también cuenta con la implementación de controladores en lenguajes como C, C++, Matlab, Python, Java, ROS, o con API.

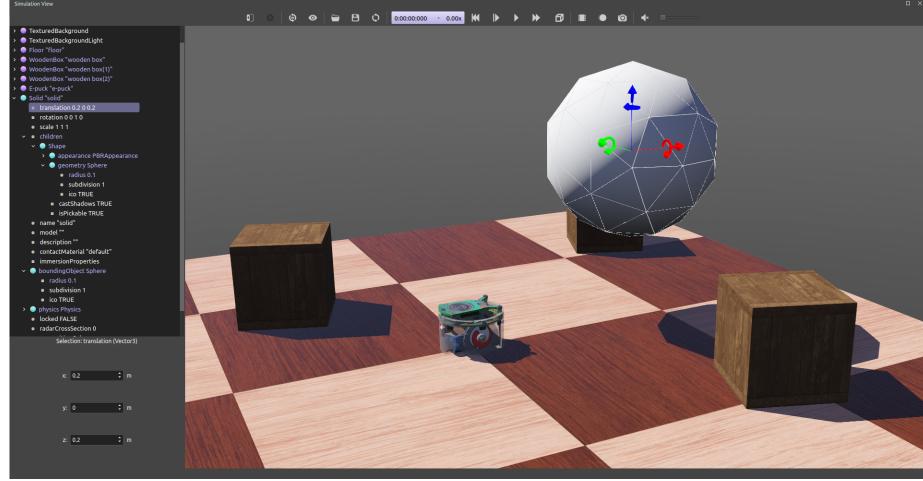


Figura 12: Entorno de simulación en Webots [22].

## Paralelismo computacional

Al momento de trabajar con algoritmos y problemas complejos, se requiere utilizar métodos de optimización de software para agilizar el procesamiento de cálculos y la ejecución de tareas. Para esto, se utilizan herramientas como el paralelismo computacional. Consiste en dividir las tareas y asignarlas a cada núcleo del procesador para ejecutarlas simultáneamente [23]. Esto reduce considerablemente el tiempo de procesamiento. Además es importante considerar las dependencias entre las tareas como:

- Dependencia de control de secuencia: Es el orden secuencial clásico de los algoritmos secuenciales.
- Dependencia de comunicación: Es cuando una tarea depende de la información que envíe otra tarea.

En la Figura 13 se observa un ejemplo donde puede existir tiempos muertos ya que la tarea 3 depende de la información enviada por la tarea 1 y la tarea 4 depende de la información enviada por la tarea 3.

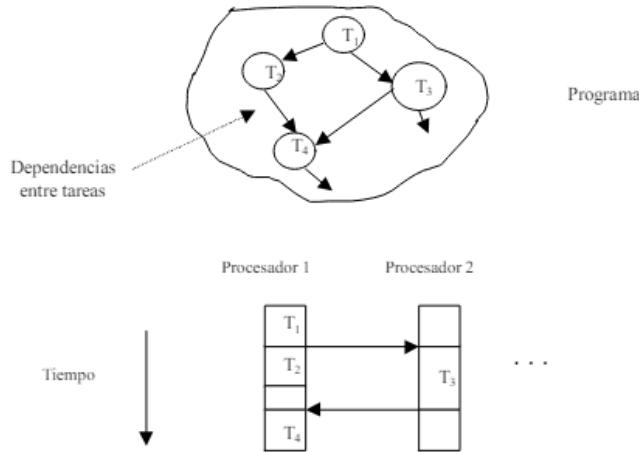


Figura 13: Ejemplo de dependencia de tareas [23].

## Ecosistema Robotat

El Robotat es una plataforma que cuenta con herramientas y sistemas de captura de movimiento útiles para experimentar con la robótica de enjambre. A continuación se describen las herramientas más importantes.

### Mesa de pruebas

Es una plataforma de acero y plycem con un espacio útil de  $5 \times 5 \times 3$  m, capaz de soportar cargas puntuales de hasta dos toneladas.

### Sistema de captura de movimiento OptiTrack

Es un sistema de captura de movimiento que consta de 6 cámaras OptiTrack Prime<sup>x</sup> 41 al rededor de la mesa de pruebas. Estas son cámaras de alta precisión y baja latencia que permiten realizar experimentos en tiempo real. En la Figura 14 se muestra el modelo disponible en la Universidad del Valle de Guatemala que tiene las siguientes características [24]:

- Resolución de 4.1 mega píxeles (MP).
- Precisión de  $\pm 0.10$  mm.
- Errores rotacionales menores a 0.5 grados.
- Lentes de 12 mm.
- Tasa de refresco nativa de 180 FPS con un máximo de hasta 250 FPS.

- Distancia de captura de hasta 100 pies desde la cámara al marcador.
- Rango de captura de hasta 290,000 pies cúbicos por cámara para marcadores pasivos
- Rango de captura de hasta 1,000,000 pies cúbicos por cámara para marcadores activos.



Figura 14: Cámara de captura de movimiento Prime<sup>x</sup> 41 de OptiTrack [24].

### Comunicación del Robotat

La plataforma del Robotat utiliza un protocolo de comunicación TCP para transmitir la información de las cámaras OptiTrack al servidor principal del laboratorio. Este servidor de Python envía los datos a través de Wi-Fi en una red local. A esta red local, se puede acceder con una computadora para extraer la información que se necesite, además las plataformas robóticas también se pueden conectar a la red local para recibir instrucciones.

### Hardware

Para poner a prueba el algoritmo de sincronización y control de formaciones en un ambiente físico, se necesita de una plataforma robótica móvil que en este proyecto será el Pololu 3pi+.

### Plataforma móvil Pololu 3pi+ modificado

La plataforma móvil a utilizar es una modificación basada en el Pololu 3pi+ 32U4 OLED Robot [25]. Tiene un diámetro de 97 mm y altura de 36 mm, además, cuenta con lo siguiente:

- Procesador ATmega34U4 MCU @ 16 MHz.
- Sensores de línea y de choque frontales.
- Encoders de doble cuadratura para control de posición y velocidad en lazo cerrado.
- IMU (acelerómetro de 3 ejes, magnetómetro y giroscopio).
- Pantalla OLED integrada.

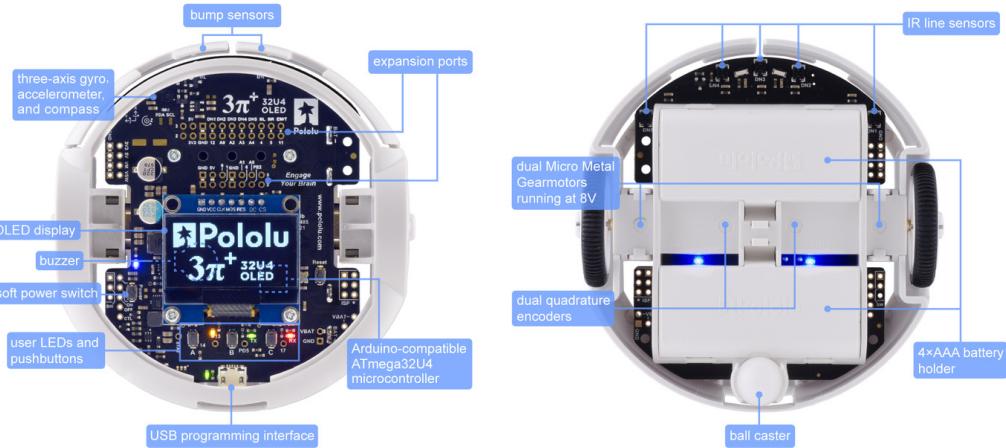


Figura 15: Sensores y componentes del Pololu 3pi+ 32U4 OLED Robot [25].

A esta plataforma móvil se le adaptó un ESP32 ya que el microcontrolador original tiene poca capacidad de procesamiento. Esto permitió tener control de capa superior e inferior, además de incorporar una red de comunicación WiFi con el Robotat para controlar los agentes.

## Metodología

La metodología que se llevará a cabo para cumplir con los objetivos establecidos es la siguiente:

### Simulaciones de la fase previa

En esta etapa se tomará el trabajo realizado anteriormente. Se ejecutarán las simulaciones en Webots con el código planteado y se replicarán algunos de los escenarios para ejecutar las trayectorias del algoritmo de sincronización y control de formaciones.

### Ajuste de la fase previa para su ejecución en el Robotat

Luego, se realizará el ajuste del algoritmo para su ejecución en el entorno del Robotat utilizando plataformas móviles como los Pololu 3Pi+ para verificar su correcto funcionamiento. Para esto es necesario realizar una calibración de los marcadores disponibles en el Robotat y comprobar la conexión de los Pololu 3Pi+ con el servidor.

### Replicar escenarios de la fase previa

Una vez comprobada la conectividad de los Pololu 3Pi+ con el servidor del Robotat, se ejecutará el algoritmo con algunos de los escenarios realizados en la fase previa en el entorno

del Robotat para verificar su correcto funcionamiento.

## Identificación de deficiencias y puntos de mejora

Una vez levantada la fase previa, se realizará un análisis de la implementación del algoritmo en cuanto a código, lenguaje de programación y métodos de comunicación para identificar las deficiencias y puntos de mejora. Luego se optimizará el algoritmo original tomando en cuenta las mejoras encontradas.

## Validación del algoritmo optimizado en escenarios previos

Una vez encontrados los puntos de mejora, se implementarán progresivamente y se realizarán pruebas utilizando los mismos escenarios que en el levantamiento de la fase previa para ejecutar las trayectorias y comparar el rendimiento de la nueva implementación del algoritmo.

## Validación del algoritmo optimizado en escenarios con obstáculos móviles

Una vez se haya obtenido una mejora del algoritmo en cuanto a tiempos de ejecución, se adaptará para evaluar su rendimiento en escenarios con obstáculos móviles en un ambiente controlado.

## Cronograma de actividades

Actividad principal	Subtarea	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
		03-jun	10-jun	17-jun	24-jun	01-jul	08-jul	15-jul	22-jul	29-jul	05-agosto	12-agosto	19-agosto	26-agosto	02-sept	09-sept	16-sept	23-sept	30-sept
Levantamiento de la fase previa	Entendimiento de funcionamiento del algoritmo																		
	Replicar las simulaciones realizadas en etapas anteriores																		
	Realizar pruebas de conexión y familiarizarse con los Pololu 3P+T																		
	Replicar las simulaciones en físico utilizando el Robotat																		
Optimización del algoritmo	Identificar deficiencias y puntos de mejoras en la implementación del algoritmo																		
	Investigar métodos de optimización para la implementación del algoritmo																		
	Aplicar las mejoras identificadas al algoritmo																		
	Validar el algoritmo mejorado utilizando los mismos escenarios que en las simulaciones anteriores																		
Validar el rendimiento del algoritmo en escenarios con obstáculos móviles	Comparar el rendimiento del algoritmo anterior y el optimizado																		
	Crear nuevos escenarios utilizando obstáculos móviles																		
Escritura de tesis	Ejecutar el algoritmo optimizado en los escenarios con obstáculos móviles																		
	Escritura y preparación de tesis																		

Figura 16: Cronograma de actividades.

# Índice preliminar

Prefacio

Lista de figuras

Lista de cuadros

Resumen

Abstract

1. Introducción

2. Antecedentes

    2.1 Robótica de enjambre

    2.2 Robótica de enjambre inspirada en peces

    2.3 Robotarium de Georgia Tech

    2.4 Robotat de la Universidad del Valle de Guatemala

    2.5 Validación de algoritmos de algoritmos de *Particle Swarm Optimization* (PSO) y *Ant Colony Optimization* (ACO)

    2.6 Algoritmo de sincronización y control de sistemas de robots multiagente para misiones de búsqueda

    2.7 Validación de un algoritmo de inteligencia de enjambre enfocado en sincronización y control de formaciones de sistemas robóticos multi-agente en un entorno físico

3. Justificación

4. Objetivos

    4.1 Objetivo general

    4.2 Objetivos específicos

5. Alcance

6. Marco teórico

    6.1 Conceptos fundamentales en robótica de enjambre

        6.1.1 Enjambre

        6.1.2 Agente

        6.1.3 Formaciones

        6.1.4 Control centralizado y descentralizado

    6.2 Conceptos básicos en Teoría de grafos

        6.2.1 Teoría de grafos

        6.2.2 Vértices y aristas

        6.2.3 Tipos de grafos

        6.2.4 Matrices asociadas a un grafo

- 6.3 Control de la formación
  - 6.3.1 Grafo de formación
  - 6.3.2 Construcción de un grafo mínimamente rígido
  - 6.3.3 Grafo para la red de comunicación
- 6.4 Teoría de control
  - 6.4.1 Control de formación
- 6.5 Cinemática de robots diferenciales con ruedas
- 6.6 Otras ecuaciones matemáticas relevantes
  - 6.6.1 Ecuación de consenso
  - 6.6.2 Funciones racionales para hallar la tensión
- 6.7 Herramientas de software
  - 6.7.1 Matlab
  - 6.7.2 Entorno de simulación Webots
  - 6.7.3 Paralelismo computacional
- 6.8 Ecosistema Robotat
  - 6.8.1 Mesa de pruebas
  - 6.8.2 Sistema de captura de movimiento OptiTrack
  - 6.8.3 Comunicación del Robotat
- 6.9 Hardware
  - 6.9.1 Plataforma móvil Pololu 3pi+ modificado
- 7. Optimización del algoritmo de sincronización y control de formaciones
- 8. Comparación de rendimiento del algoritmo optimizado en escenarios similares a los probados anteriormente
- 9. Validación de rendimiento del algoritmo optimizado en un ambiente controlado con obstáculos móviles
- 10. Conclusiones
- 11. Recomendaciones
- 12. Bibliografía
- 13. Anexos

## Referencias

- [1] M. Ekelhof y G. Persi, *Robótica de enjambre*, 2023. dirección: [https://unidir.org/wp-content/uploads/2023/05/UNIDIR\\_Swarms\\_SinglePages\\_web\\_SP.pdf](https://unidir.org/wp-content/uploads/2023/05/UNIDIR_Swarms_SinglePages_web_SP.pdf).
- [2] R. Solís, *Enjambres de robots y sus aplicaciones en la exploración y comunicación*, 2019. dirección: <http://hdl.handle.net/2238/13120>.
- [3] L. Burrows, *Robotic swarm swims like a school of fish*, 2021. dirección: <https://wyss.harvard.edu/news/robotic-swarm-swims-like-a-school-of-fish/>.

- [4] J. Maderer, *Robotarium: A Robotics Lab Accessible to All*, 2017. dirección: <https://news.gatech.edu/archive/features/robotarium-robotics-lab-accessible-all.shtml>.
- [5] P. Barrera, *16 datos sobre el Robotat*, 2023. dirección: <https://noticias.uvg.edu.gt/datos-robotat-habitat-robotica-cit-116/>.
- [6] P. Barrera, *Cuando la física granular y la robótica se apoyan una a otra*, 2022. dirección: <https://noticias.uvg.edu.gt/robotat-robotario-fisica-granular-robotica/>.
- [7] J. Menéndez, “Validación de los algoritmos de robótica de enjambre Particle Swarm Optimization y Ant Colony Optimization con sistemas robóticos físicos en el ecosistema Robotat,” Tesis de licenciatura, Universidad Del Valle de Guatemala, 2023.
- [8] A. Peña, “Algoritmo de sincronización y control de sistemas de robots multi-agente para misiones de búsqueda,” Tesis de licenciatura, Universidad Del Valle de Guatemala, 2019.
- [9] J. Rodríguez, “Validación de un algoritmo de inteligencia de enjambre enfocado en sincronización y control de formaciones de sistemas robóticos multi-agente en un entorno físico,” Tesis de licenciatura, Universidad Del Valle de Guatemala, 2023.
- [10] S. Chatterjee, *Part 1; Swarm Robots: Definition, System and Cycles*, 2017. dirección: <https://csoham.com/2017/08/09/part-1-swarm-robots-definition-system-and-cycles/>.
- [11] Leotronics, *Interacción de los enjambres y robótica de grupo en la resolución de diversas tareas*. dirección: <https://leotronics.eu/es/nuestro-blog/interaccion-de-los-enjambres-y-robotica-de-grupo-en-la-resolucion-de-diversas-tareas> (visitado 2024).
- [12] B. Moreno y J. Hernández, “Control centralizado y descentralizado de edificaciones mediante acristalamientos activos,” *Revista de Investigación “Pensamiento Matemático”*, vol. 7, n.º 1, págs. 19-38, 2017.
- [13] V. Noble, *Teoría de grafos*, 2022. dirección: [https://rpubs.com/Yelky99/Tgrafos\\_pmai](https://rpubs.com/Yelky99/Tgrafos_pmai).
- [14] A. Ruiz-Ruano y J. López, “Modelos Gráficos y Redes en Psicología,” *Revista de Historia de la Psicología*, vol. 41, n.º 4, págs. 24-33, 2020.
- [15] F. Franco, “Aspectos algebraicos en Teoría de Grafos,” Tesis de licenciatura, Universidad de Sevilla, España, 2016.
- [16] L. Crespo, *Matroides y rigidez de grafos*, 2020. dirección: <http://hdl.handle.net/10902/20492>.
- [17] L. Krick, “Application of Graph Rigidity in Formation Control of Multi-Robot Networks,” Tesis de maestría, Universidad de Toronto, 2007.
- [18] M. Mesbahi y M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, 2010, ISBN: 9780691140612. dirección: [https://www.ebook.de/de/product/10047947/mehran\\_mesbahi\\_magnus\\_egerstedt\\_graph\\_theoretic\\_methods\\_in\\_multiagent\\_networks.html](https://www.ebook.de/de/product/10047947/mehran_mesbahi_magnus_egerstedt_graph_theoretic_methods_in_multiagent_networks.html).
- [19] N. Nise, *Control system engineering*. Benjamin/Cummings, 1995.
- [20] M. Zea, *Lectura 12: Control de robots móviles con ruedas*. (visitado 2024).

- [21] MathWorks, *MATLAB - El lenguaje del cálculo técnico*. dirección: <https://la.mathworks.com/products/matlab.html> (visitado 2024).
- [22] Cyberbotics, *Webots*. dirección: <https://cyberbotics.com/> (visitado 2024).
- [23] J. Aguilar y E. Leiss, *Introducción a la Computación Paralela*. Universidad de los andes Mérida-Venezuela, 2004, ISBN: 980-12-0752-3. dirección: [https://gc.scalahed.com/recursos/files/r161r/w25041w/introduccionalacomputacionparalela\\_S5.pdf](https://gc.scalahed.com/recursos/files/r161r/w25041w/introduccionalacomputacionparalela_S5.pdf).
- [24] OptiTrack, *Primex 41*. dirección: <https://optitrack.com/cameras/primex-41/> (visitado 2024).
- [25] Pololu, *3pi+ 32U4 OLED Robot - Standard Edition (30:1 MP Motors), Assembled*. dirección: <https://www.pololu.com/product/4975> (visitado 2024).