Estudio del comportamiento de un algoritmo de swarm aggregation con un líder y evasión de obstáculos

Gabriel Álvarez
Estudiante del Máster Universitario de Ingeniería Informática
Universidad de Zaragoza
Zaragoza, España
781429@unizar.es

***Abstract*— Este documento se enfoca en la implementación de un algoritmo de swarm aggregation con un líder y evasión de obstáculos, y la experimentación de sus parámetros. Durante el estudio se pudieron conseguir resultados que corroboran lo que se puede deducir de las ecuaciones usadas en la implementación. Además, se pudo encontrar una configuración de parámetros con el comportamiento deseado. Finalmente, los resultados obtenidos durante el estudio fueron satisfactorios lo que implica que este algoritmo podría ser aplicado en un entorno más realista.**

I. Introducción

El estudio y desarrollo de comportamientos en sistemas de multi-robots ha sido de gran importancia en las últimas décadas. Se han producido grandes avances en esta área. En particular, este documento tiene como finalidad la implementación y experimentación del comportamiento de un algoritmo descentralizado de swarm aggregation con evasión de obstáculos.

El algoritmo implementado utiliza como base el descrito en el artículo "*A Swarm Aggregation Algorithm based on Local Interaction with Actuator Saturations and Integrated Obstacle Avoidance"* [1]. Durante el desarrollo de este proyecto se realizaron varios experimentos, con los diferentes parámetros presentes en el algoritmo, para evaluar el comportamiento del mismo.

Además, se agregó un robot líder, el cual puede moverse de manera circular o recta a través del mapa, con la finalidad de que el resto de los robots del enjambre lo siguieran. Y así, evaluar el algoritmo cuando el enjambre no es estático o no ha convergido a un punto en particular. Por otro lado, durante la implementación al realizar pruebas, y comparar con el entorno definido en el artículo [1] en el que está basado este documento, se tomó como dimensiones del mapa de 4 por 4 metros. En los siguientes puntos se describirán la implementación del algoritmo y se mostrarán los resultados obtenidos.

II. Implementación

El proyecto fue desarrollado usando la versión de *Python 3.7.4*, en conjunto con las librerías de *numpy*, para los cálculos matemáticos y operaciones de vectores y *pandas* para la generación de los ficheros con las distancias. También se utilizó la librería *matplotlib*, para mostrar las posiciones de los robots.

En la implementación de este algoritmo, se utilizó una matriz de adyacencia de un grafo no-dirigido, para tener la vecindad de cada robot, la cual es actualizada en cada instante de tiempo. En donde, se tendrá el valor 1 para una casilla de la matriz *ij* y *ji* si la distancia entre el *robot i* y el *robot j* es menor o igual a un cierto rango de visión *r*, en caso contrario el valor es 0 [1].

$$E(t) = \{\epsilon_{ij}(t)\}$$

Si *E(t)* es el conjunto de lados del grafo no-dirigido en el instante *t*. Entonces la expresión usada para actualizar la matriz de adyacencia sería la siguiente [1]:

$$\epsilon_{ij}(t) = 1 \Leftrightarrow \|x_i - x_j\| \leq r \ , \ sino \ \epsilon_{ij}(t) = 0$$

Además, se tiene la siguiente fórmula para actualizar las posiciones de los robots a través del tiempo [1].

$$\dot{x_i} = \frac{\sum_{j \in N_i(t)} \gamma_{ij} g(x_i - x_j)}{\sum_{j \in N_i(t)} \gamma_{ij}} \quad (1)$$

Donde *Ni(t)* representa al conjunto de los vecinos del *robot i* en el instante de tiempo *t* [1].

Las función gamma, llamada factor de peso [1]:

$$\gamma_{ij} = \frac{1}{\left\|x_i - x_j\right\|^{\alpha}} \ , \ con \ \alpha \geq 1$$

En donde, se puede observar que a medida que la distancia entre el *robot i* y el *robot j* decrece, el resultado de esta función crece. El valor del parámetro alfa, indicará qué tanto se quiere que la distancia entre ambos robots afecte los resultados. Para valores de distancia muy grandes, cuando alfa sea mayor a 1, hará que el valor de gamma sea aún más pequeño, y viceversa, para valores de distancia muy pequeños.

Por otro lado, tenemos la función de interacción *g(.)*, la cual viene definida por la fórmula [1]:

$$g(y) = \frac{y}{\|y\|}\left[(g_r(\|y\|) - g_a(\|y\|))\right], \quad y \in \mathbb{R}^d \quad (2)$$

$$con\ g_a(.) : \mathbb{R} \to \mathbb{R}\ \ y\ \ g_r(.) : \mathbb{R} \to \mathbb{R}$$

Donde, *y* es un vector entre el *robot i* y el *robot j*. Las funciones *gr* y *ga*, son las funciones de repulsión y atracción, respectivamente. La cuales vienen definidas por las siguientes fórmulas [1]:

$$g_a(\|y\|) = a(1 - \Phi(\|y\|)), \quad (3)$$
$$g_r(\|y\|) = b(\Phi(\|y\|)),$$

$$Donde\ a,\ b > 0 \text{ son valores constantes y } \Phi(.) : \mathbb{R} \to \mathbb{R}$$

Finalmente, la función fi utilizada fue la descrita en el artículo, la cual tiene la característica de que es monotónica, el valor del límite cuando la distancia tiende a 0, es 1. Y cuando cuando la distancia tiende a infinito, el valor del límite es 0 [1]:

$$exp(-\frac{\|y\|^{\beta}}{c})\ con\ \beta \geq 1\ y\ c > 0$$

Una vez definidas estas ecuaciones, se procedió a realizar el algoritmo, el cual consistía en:

1. Inicializar un arreglo de posiciones (x, y) de los robots de manera aleatoria, en este vector se incluye al robot líder en la posición n, donde n es el número de robots del enjambre. De manera que, los primeros *n* elementos del arreglo de posiciones tienen las posiciones de los robots del enjambre, seguido por la posición del robot líder.
2. Inicializar los obstáculos, los cuales también forman parte del arreglo de posiciones y empiezan desde la posición *n+1*. Los obstáculos mantienen una posición fija durante la ejecución del programa, a diferencia del resto de los robots.
3. Un bucle infinito en donde se imprimen las posiciones de los vectores, usando *matplotlib*. Se actualiza la matriz de adyacencia y se calculan las nuevas posiciones de los robots del enjambre usando la ecuación 1, descrita anteriormente. Además, se actualiza la posición del robot líder usando una función para generar el movimiento circular o recto, dependiendo del parámetro de entrada del programa.

Luego, se procedió a realizar experimentos variando los parámetros del algoritmo, los cuales son:

- ➔ α: para la función gamma.
- ➔ a: para la función de atracción.
- ➔ b: para la función de repulsión.
- ➔ β: para la función fi.
- ➔ c: para la función fi.

En el siguiente apartado se muestran los resultados obtenidos.

III. Resultados

Se tienen los parámetros fijos del tamaño del mapa, así como la atracción del robot líder y la repulsión de los obstáculos como se describe en la siguiente tabla:

TABLA I
PARÁMETROS FIJOS DEL EXPERIMENTO

Constante de atracción del robot líder	4 veces mayor al usado por los robots del enjambre.
Constante de repulsión de los obstáculos	4 veces mayor al usado por los robots del enjambre.
Tamaño del mapa	4x4, con (x, y) ϵ [-2, 2] x [-2, 2]
Número de iteraciones	300
Rango de visión	10 (Todos los nodos son vecinos entre sí)
β(fi)	4.0
c(fi)	0.02

El tamaño del mapa fue tomado utilizando como referencia al mapa del artículo [1] en donde se asumió que tenía unas dimensiones de alrededor de 4x4 metros. Por otro lado, se utilizaron constantes de atracción y repulsión particulares para el robot líder y los obstáculos, respectivamente. Con la finalidad de que los robots del enjambre tuviesen una mayor atracción al robot líder y evitaran los obstáculos.

TABLA II
PARÁMETROS GENERALES DEL EXPERIMENTO

Número de robots	5, 10 y 25
Número de obstáculos	0, 5, 10
Tipo de movimiento del robot líder	Rectilíneo, circular
Velocidad del robot líder	0.005, 0.05 y 0.5
Velocidad angular del robot líder	0.5, 1.0 y 2.0

Se variaron el número de robots del enjambre y las velocidades del robot líder, pudiéndose notar como la influencia del robot líder disminuye a medida que la cantidad de robots del enjambre aumenta, lo que podría ser solucionado aumentando la constante de atracción del robot líder o disminuyendo su velocidad. Finalmente, se implementó un movimiento rectilíneo, pero al descubrir que el robot líder se salía del mapa, se cambió a un movimiento circular, para mantener al robot líder dentro del mapa.

TABLA III
PARÁMETROS DEL MÉTODO

	α(gamma)	**a(attr)**	**b(repul)**
Buena	1.0	4.0	0.4
Regular	1.0	0.4	0.4
Mala	1.0	0.4	4.0

Para la búsqueda de los parámetros óptimos del algoritmo, se fijaron el número de robots a 10, el tipo de movimiento del robot líder a circular, su velocidad angular a 1.0 y el número de obstáculos a 10.

A. Configuración buena, constante de repulsión menor a la constante de atracción.

Para la configuración buena se utilizaron las constantes de atracción y repulsión definidas en el artículo [1], debido a que el mapa tiene dimensiones similares. A continuación se muestran unas figuras, demostrando el comportamiento del algoritmo para estos parámetros

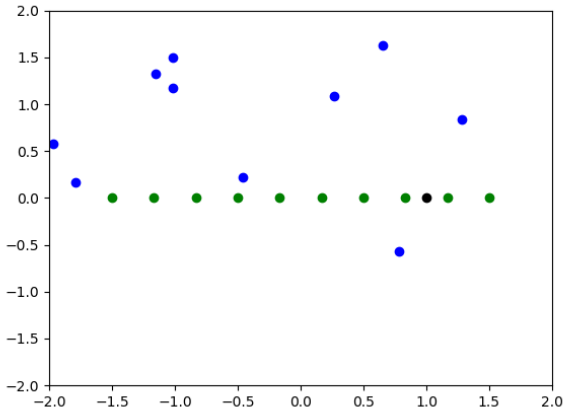

Fig 1. Etapa inicial, configuración buena.

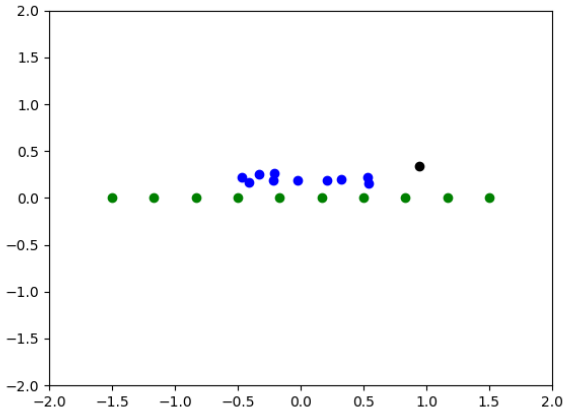

Fig 2. Etapa media, configuración buena.

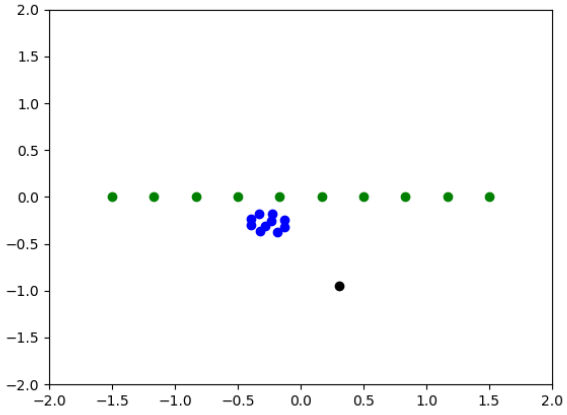

Fig 3. Etapa final, configuración buena.

Nota: Los robots del enjambre están representados por el color ***azul***, el robot líder por el color ***negro*** y los obstáculos por el color ***verde.***

Se puede observar en las diferentes etapas como el enjambre tiende a agruparse primero, para luego empezar a seguir al robot líder.

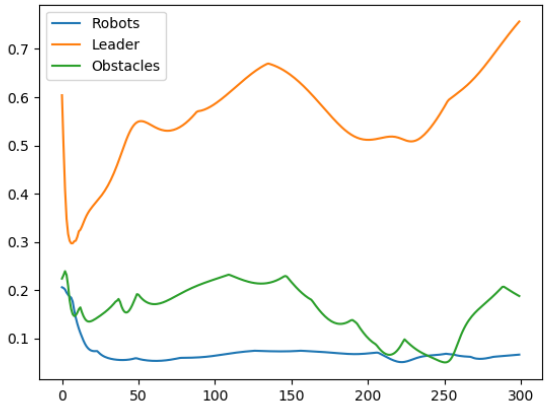

Fig 4. Distancias mínimas entre robots del enjambre, líder y obstáculos para cada iteración del ciclo, configuración buena.

En la figura anterior, se puede observar como las distancias entre los robots del enjambre, representado por la línea azul, disminuye y llega a un punto de equilibrio en el cual no hay una mayor variación. Por otro lado, se tiene que la distancia mínima a los obstáculos nunca llega a ser 0, por lo que se puede decir, que efectivamente, los robots del enjambre están evitandolos. Finalmente, la distancia al robot líder, presenta un aumento debido a que el robot líder se mueve a una velocidad suficientemente rápida como para que los robots del enjambre no se acerquen demasiado.

B. Configuración regular, constante de repulsión igual a la de constante de atracción.

A continuación se muestran unas figuras, demostrando el comportamiento del algoritmo para estos parámetros:

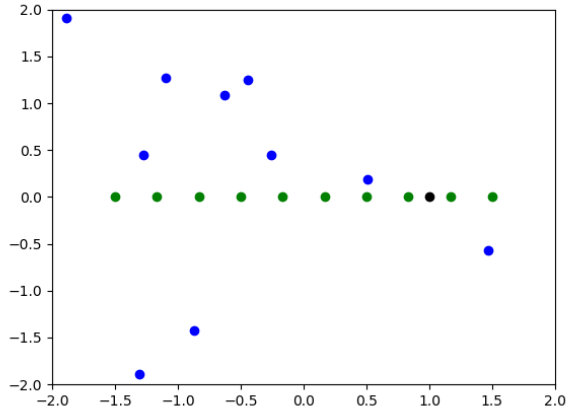

Fig 5. Etapa inicial, configuración regular.

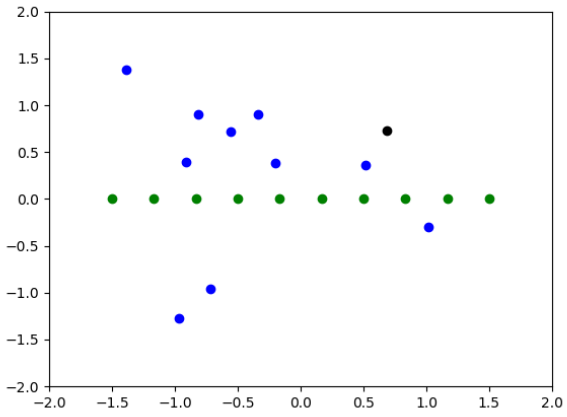

Fig 6. Etapa media, configuración regular.

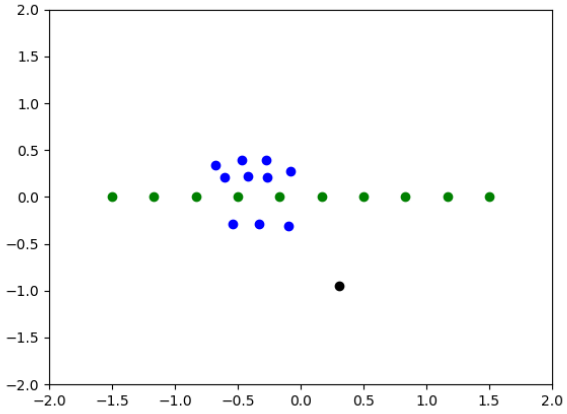

Fig 7. Etapa final, configuración regular.

Nota: Los robots del enjambre están representados por el color ***azul***, el robot líder por el color ***negro*** y los obstáculos por el color ***verde.***

Se puede observar en las diferentes etapas como el enjambre tiende a agruparse primero, para luego empezar a seguir al robot líder, como en el caso anterior. Sin embargo, hay una mayor separación en los robots del enjambre.

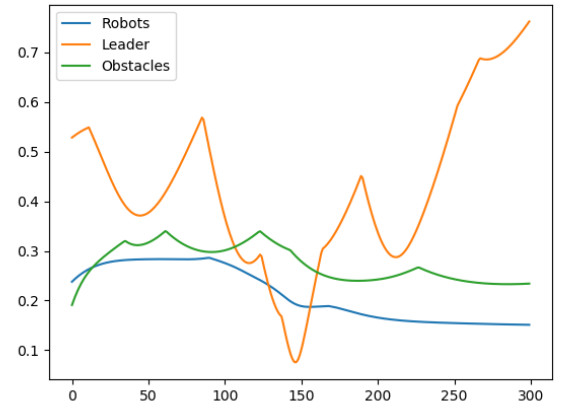

Fig 8. Distancias mínimas entre robots del enjambre, líder y obstáculos para cada iteración del ciclo, configuración regular.

En la figura anterior, se puede observar como las distancias entre los robots del enjambre, representado por la línea azul, disminuye de una manera menos pronunciada, que el caso anterior. Por otro lado, al igual que el caso anterior, los obstáculos son evadidos de manera satisfactoria. Por último, al igual que en el caso anterior, el robot líder se mueve a una velocidad que hace que se separe eventualmente.

C. Configuración mala, constante de repulsión mayor a la constante de atracción.

A continuación se muestran unas figuras, demostrando el comportamiento del algoritmo para estos parámetros:

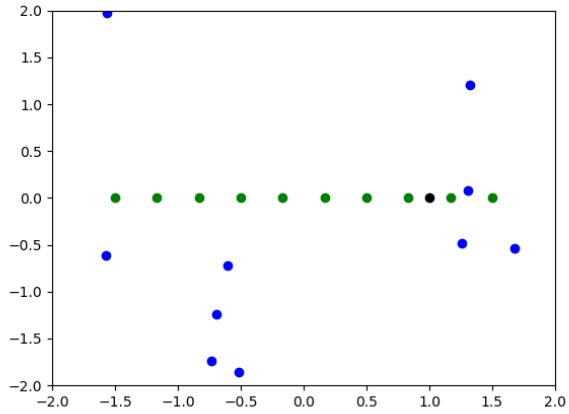

Fig 9. Etapa inicial, configuración mala.

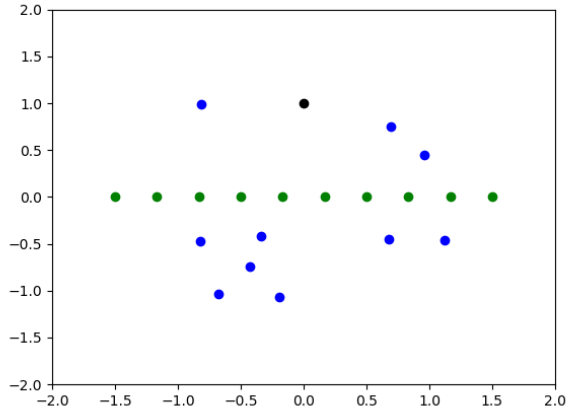

Fig 10. Etapa media, configuración mala.

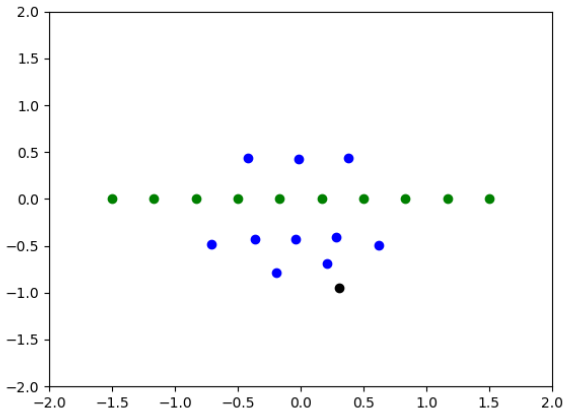

Fig 11. Etapa final, configuración mala.

Nota: Los robots del enjambre están representados por el color ***azul***, el robot líder por el color ***negro*** y los obstáculos por el color ***verde.***

Se puede observar en las diferentes etapas como el enjambre tiene una tendencia a agruparse, pero las distancias entre los robots son alrededor de 3 veces mayores al caso anterior, de manera que el comportamiento del enjambre no se ve con tanta claridad, al haber tanta separación.

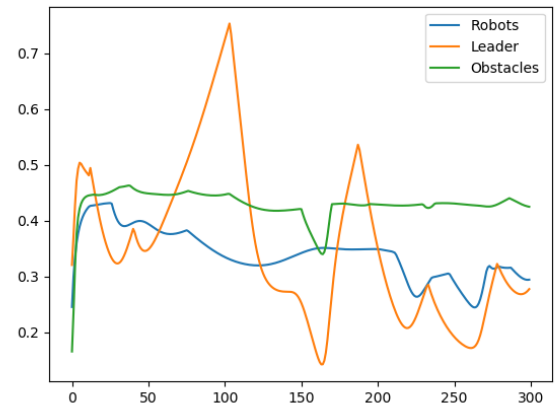

Fig 12. Distancias mínimas entre robots del enjambre, líder y obstáculos para cada iteración del ciclo, configuración mala.

En la figura anterior, se puede observar como las distancias entre los robots del enjambre, representado por la línea azul, disminuye poco. Por otro lado, se sigue manteniendo la evasión de obstáculos ya que la distancia mínima nunca llega a 0. Finalmente, la distancia al robot líder, presenta una variación que viene dada debido a las distancias que hay entre los robots del enjambre, los cuales están más dispersos en el mapa, y es por esto que el gráfico pareciera mostrar una menor distancia con el líder en la última iteración a diferencia de los casos anteriores.

D. Variar el parámetro α sobre la mejor configuración.

Después de encontrar una configuración adecuada, se procedió a variar el parámetro alfa de la ecuación gamma, con los valores de 1.0, 2.0 y 3.0. En donde, se pudo observar que los robots del enjambre no seguían al robot líder ya que este parámetro aumenta la importancia a las distancias cortas.

A continuación se muestran un gráfico de las distancias mínimas y máximas para los diferentes valores del parámetro alfa.

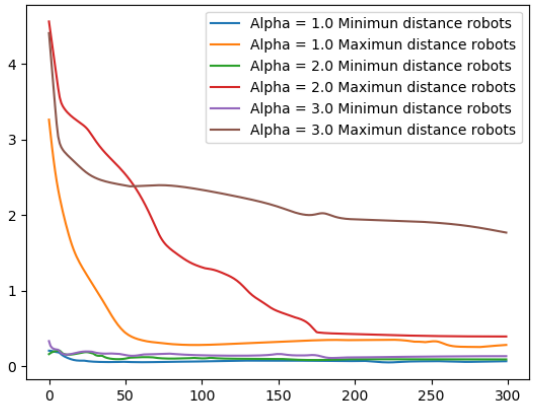

Fig 13. Distancias mínimas y máximas para la configuración buena y parámetro alfa igual a 1.0, 2.0 y 3.0.

Se puede observar cómo a medida que se aumenta el alfa, hay una mayor tendencia a que se generen subgrupos de enjambres, si los robots están lo suficientemente separados. Para un alfa de 3.0 la distancia máxima al final de la ejecución es mucho mayor a su respectiva distancia mínima, lo que indica que hay al menos dos subgrupos del enjambre. Esto demuestra la hipótesis de que aumentar el alfa, aumenta la preferencia por distancias cortas, y por consiguiente, la formación de subgrupos.

E. Aumentar el número de robots sobre la mejor configuración.

Aumentar el número de robots no parece indicar un comportamiento demasiado diferente al obtenido con 10 robots. A continuación, se muestra la figura de distancias mínimas.

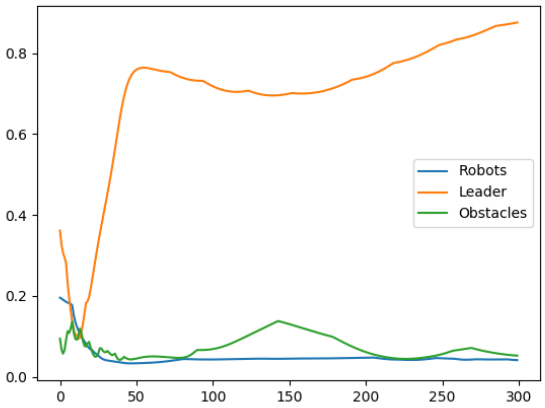

Fig 14. Distancias mínimas entre robots del enjambre, líder y obstáculos para cada iteración del ciclo, configuración buena con 25 robots.

En donde, se puede observar que hubo agrupamiento completo del enjambre, y además no hubo colisiones con los obstáculos debido a que la distancia mínima nunca fue 0. Por otro lado, la distancia con el robot líder no parece indicar una diferencia relevante con la configuración de 10 robots.

F. La mejor configuración, sin obstáculos.

Eliminar los obstáculos tuvo como consecuencia que los robots del enjambre siguieran al robot líder a una menor distancia. A continuación, se muestra la figura de distancias mínimas.

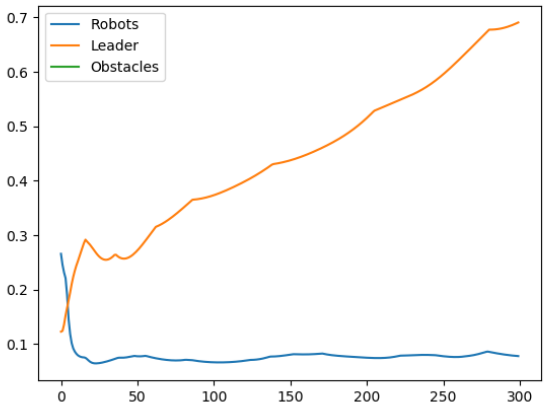

Fig 15. Distancias mínimas entre robots del enjambre, líder y obstáculos para cada iteración del ciclo, configuración buena sin obstáculos.

En donde, se puede observar que las distancias con el robot líder aumentaron debido a su velocidad, pero no hubo una mayor variación en esta, debido a que, al no haber obstáculos, no hay la influencia de los mismos en el comportamiento. Cabe destacar que los obstáculos, al igual que el resto de los elementos del experimento, poseen constante de atracción.

IV. Conclusiones

Se puede concluir que los resultados fueron satisfactorios, se pudo realizar una exploración de los parámetros y cómo afectan estos, al comportamiento del algoritmo, se pudo conseguir una configuración de parámetros adecuada en la que el comportamiento del algoritmo mostró los mejores resultados, además se pudo observar como efectivamente no había colisiones entre los robots del enjambre y los obstáculos.

V. Futuras Mejoras

- ➔ Experimentar un con el resto de los parámetros del experimento, como los de la función fi y los de velocidad.
- ➔ Probar con diferentes dimensiones del mapa.
- ➔ Usar la otra función generalizada propuesta en el artículo [1].
- ➔ Llevar este estudio a una versión multinodo usando ROS e integrar esta versión con Gazebo, de manera que se pueda visualizar el comportamiento de los robots en un entorno más realista.
- ➔ Agregar evasión de obstáculos al robot líder, ya que en un entorno real el robot líder no debería chocar con estos.

VI. Referencias

[1] A. Leccese, A. Gasparri, A. Priolo, G. Oriolo and G. Ulivi, "A swarm aggregation algorithm based on local interaction with actuator saturations and integrated obstacle avoidance," 2013 IEEE International Conference on Robotics and Automation, 2013, pp. 1865-1870, doi: 10.1109/ICRA.2013.6630823.