World Journal of Engineering and Technology, 2021, 9, 155-172
https://www.scirp.org/journal/wjet
ISSN Online: 2331-4249
ISSN Print: 2331-4222

Review and Simulation of a Novel Formation Building Algorithm While Enabling Obstacle Avoidance

Michael Levkovsky

School of Electrical Engineering and Telecommunications, University of New South Wales, Kensington, Sydney, New South Wales, Australia
Email: ing.michael2014@outlook.com

How to cite this paper: Levkovsky, M. (2021) Review and Simulation of a Novel Formation Building Algorithm While Enabling Obstacle Avoidance. *World Journal of Engineering and Technology*, **9**, 155-172.
https://doi.org/10.4236/wjet.2021.91012

Received: November 15, 2020
Accepted: February 7, 2021
Published: February 10, 2021

Abstract

Technically, a group of more than two wheeled mobile robots working collectively towards a common goal are known as a multi-robot system. An increasing number of industries have implemented multi-robot systems to eliminate the risk of human injuries while working on hazardous tasks, and to improve productivity. Globally, engineers are continuously researching better, simple, and faster cooperative Control algorithms to provide a Control strategy where each agent in the robot formation can communicate effectively and achieve a consensus in their position, orientation and speed. This paper explores a novel Formation Building Algorithm and its global stability around a configuration vector. A simulation in MATLAB® was carried out to examine the performance of the Algorithm for two geometric formations and a fixed number of robots. In addition, an obstacle avoidance technique was presented assuming that all robots are equipped with range sensors. In particular, a uniform rounded obstacle is used to analyze the performance of the technique with the use of detailed geometric calculations.

Keywords

Consensus Variables, Adjacency Matrix, Decentralized Control, Formation Building Algorithm, Unicycle Robot

1. Introduction

Multiple studies show that novel and better multi-agent formation algorithms have been proposed in recent years. The idea behind developing control laws for large-scale systems is to provide an alternative for the control of a group of autonomous vehicles dynamically decoupled [1] [2], moving along a path, and preserving a geometric shape. The first step in understanding these algorithms is to know how to implement the basic motion dynamics found in previous research in biology [3].

The use of studies on basic motion dynamics in biology let us generate powerful algorithms to recreate geometric formations for groups of wheeled mobile vehicles [4]. Some animal species have shown that the analysis of the individual and group motion kinematics provides an initial idea on how to control large-scale systems [4] [5]. In robotics, that means that a number of robots collaborate to form geometric patterns, with a common orientation and speed. Although the journal paper under review provides a powerful insight of a distributed control law, many other approaches have been explored throughout the history, such as the leader-follower or neural network approach.

The goal of the research behind Formation Building Algorithms is to reduce the risk of hazardous tasks currently done by humans [6] [7]. In [1], particularly, the implementation of consensus variables, updates the position, linear velocity, and angular velocity with the information available from the adjacent robots or by the data transmitted from a robot neighbor at a time instant. In the end, the consensus variables algorithm, is designed to find a common speed and orientation to maintain a desired spatial formation. The angular and linear velocity value control input is calculated, with hard constraints applied on them. These constraints are introduced to improve the stability of nonlinear systems [1].

Different algorithms have been formulated for multi-agent systems exploring a vast number of control laws for a first and a second order approach. The first visible example of some of the control algorithms can be found in [8] or perhaps in [9] where the orientation of the robot changes in relation to the average heading of its adjacent robots in the system. In [9] the speed of each robot changes as in [10]; however, in this case, we use the average speed accordingly. Algorithms such as in [11] study the Lyapunov method in detail to understand the system stability, here the non-holonomic motion model and energy equations are used to formulate the control dynamics of the system. Some of them lie in the lack of analysis in real scenarios as in practice many physical and environmental constraints must be considered. Therefore, the best approach examines the use of distributed control laws with hard constraints in the system parameters.

The aim is to model a Formation Building Algorithm presented in the journal article [1]. Additionally, the document focuses on the system components identification, and the performance measures to be analyzed. The rationale behind the study and simulation of the Control Law is to provide a graphical representation of the geometrical formation, and to present an alternative to the Leader-Follower approach. A fictitious target defines the direction of the collective, where a linear and triangular formation can be used in the supply chain or mining industry, or in search and rescue activities. In addition to the Control Law, this document presents an obstacle avoidance technique that seeks to evade a rounded obstacle while maintaining the formation.

2. Literature Review

2.1. Formation Building Algorithm

To begin with, we take the spatial coordinates (x, y) in a global reference frame, and the orientation θ about the *x*-axis (**Figure 1**).

Let us take the linear speed $v_i(t)$ and the angular velocity $w_i(t)$ as the control inputs of the system, computed later with a novel distributed control technique based on the consensus between vehicles, and their adjacency in the reference frame. Such adjacency is calculated with the help of the Euclidean distance formula (**Figure 2**).

If the distance is less or equal to a reference range, they are said to be connected through an edge under the graph analysis [12]. Now, the kinematics of any unicycle robot are:

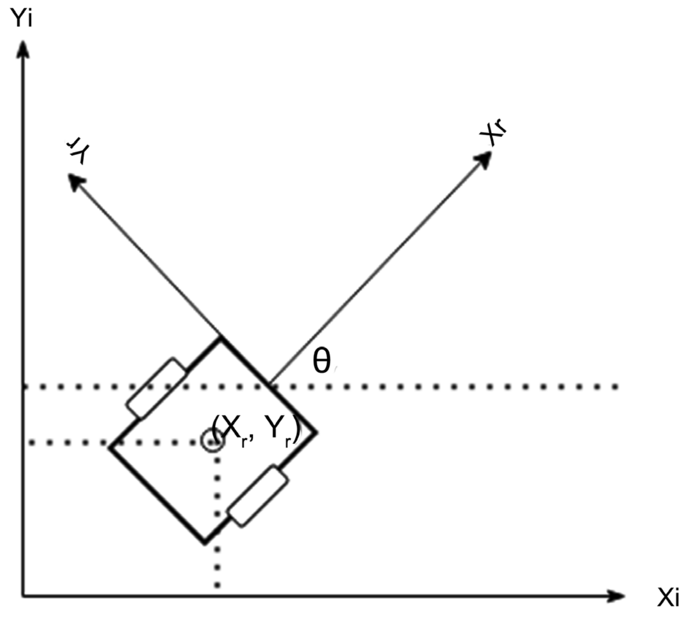

Figure 1. Mobile robot kinematics.

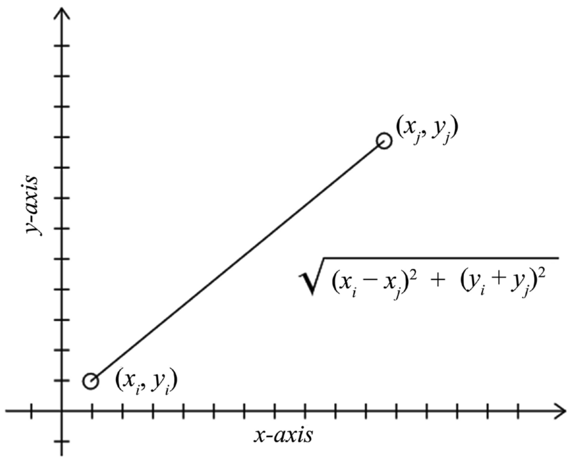

Figure 2. Euclidean distance between two nodes.

$$\begin{pmatrix} \dot{x}_i(t) \\ \dot{y}_i(t) \\ \dot{\theta}_i(t) \end{pmatrix} = \begin{pmatrix} v_i(t)\cos(\theta_i(t)) \\ v_i(t)\sin(\theta_i(t)) \\ w_i(t) \end{pmatrix} \tag{1}$$

Let $i = 1, 2, \cdots, n$ be the number of robots found in the multi-robot system, each with Cartesian coordinates $(x_i(t), y_i(t))$ with its heading measured counter-clockwise from the x-axis. As explained before the linear and angular speed control inputs are applied distributed to each of the agents of the group of wheeled mobile vehicles. The standard kinematics are not only used to describe the dynamical behavior of wheeled vehicles, but for many other artifacts, such as UAVs, smart missiles, etc. [13] [14] [15] [16] [17]. A set of initial constraints are defined for the Angular and linear speed to reduce the complexity and the instability risk:

$$\begin{aligned} &-w^{\max} \le w_i(t) \le w^{\max} \\ &V^m \le v_i(t) \le V^M \\ &w^{\max} > 0 \text{ and } 0 < V^m < V^M \end{aligned} \tag{2}$$

Each robot communicates at a discrete time $k = 1, 2, \cdots, n$. Based on the graph analysis we assume that each of the nodes that belongs to the group of robots is connected through a set of lines that forms an undirected graph. In [18], it is mentioned that each of the robots can communicate at a time instant if and only if one of them is inside an imaginary disc with radius r_c. Back in [1], it is said that all nodes connected forming a graph are grouped using the notation $\mathcal{N}_i(k)$. Moving to the consensus variable approach, a global linear speed, angular speed and robot orientation is found using the equations shown below, with initial conditions $\widetilde{x_i(0)}$, $\widetilde{y_i(0)}$, $\widetilde{\theta_i(0)}$ and $\widetilde{v_i(0)}$. This guarantees that all robots will move on a common reference system, with the same speed and orientation.

Some of the assumptions found in [1] are that the consensus angle satisfies the initial condition interval $[0, \pi)$, that the information shared between robots are the coordinates (x_j, y_j) and that the consensus variables $\widetilde{x}_j, \widetilde{y}_j, \widetilde{v}_j$ and $\widetilde{\theta}_j$, are calculated for all $j \in \mathcal{N}$ for $t \le 0$. In [1], the Formation Building Algorithm proposed to update the consensus variables is:

$$\begin{aligned} \widetilde{\theta}_i(k+1) &= \frac{\widetilde{\theta}_i(k) + \sum_{j\in\mathcal{N}_i(k)} \widetilde{\theta}_j(k)}{|1+\mathcal{N}_i(k)|} \\ \widetilde{x}_i(k+1) &= \frac{\left(\widetilde{x}_i(k) + x_i(k)\right) + \sum_{j\in\mathcal{N}_i(k)} \left(\widetilde{x}_j(k) + x_j(k)\right)}{|1+\mathcal{N}_i(k)|} - x_i(k+1) \\ \widetilde{y}_i(k+1) &= \frac{\left(\widetilde{y}_i(k) + y_i(k)\right) + \sum_{j\in\mathcal{N}_i(k)} \left(\widetilde{y}_j(k) + y_j(k)\right)}{|1+\mathcal{N}_i(k)|} - y_i(k+1) \\ \widetilde{\theta}_i(k+1) &= \frac{\widetilde{\theta}_i(k) + \sum_{j\in\mathcal{N}_i(k)} \widetilde{\theta}_j(k)}{|1+\mathcal{N}_i(k)|} \end{aligned} \tag{3}$$

If we analyze further the behavior of the equations when k tends to infinite, we may find that each of the values converges to a constant value $\widetilde{\theta_o}, \widetilde{v_o}, \widetilde{x_o}, \widetilde{y_o}$, such that:

$$\begin{aligned} &\lim_{k\to\infty} \widetilde{\theta_i}(k) = \widetilde{\theta_o} \\ &\lim_{k\to\infty} \widetilde{v_i}(k) = \widetilde{v_o} \\ &\lim_{k\to\infty} \left(\widetilde{x_i}(k) + x_i(k)\right) = \widetilde{X_o} \\ &\lim_{k\to\infty} \left(\widetilde{y_i}(k) + y_i(k)\right) = \widetilde{Y_o} \end{aligned} \tag{4}$$

This is shown in [1], and further analyzed in [8]. The journal article and this review proved that all robots will move parallel to the *x*-axis with a stabilizing control law with initial position $(x_i(0), y_i(0))$ and a geometrical configuration vector denoted by $\mathcal{C} = X_1, X_2, \cdots, X_n, Y_1, Y_2, \cdots, Y_n$ where X_n and Y_n are constants. Moreover, if the solution of the closed control loop with the robot kinematic equations in continuous time satisfies:

$$\begin{aligned} &\lim_{t\to\infty} \theta_i(t) = 0 \\ &\lim_{t\to\infty} v_i(t) = \widetilde{v_o} \\ &\lim_{t\to\infty} \left(x_i(t) - x_j(t)\right) = X_i - X_j \\ &\lim_{t\to\infty} \left(y_i(t) - y_j(t)\right) = Y_i - Y_j \end{aligned} \tag{5}$$

From the above definition one can say that all vehicles will have the same speed and orientation along the *x*-axis maintaining the formation configuration $\mathcal{C}$. The journal article [1] introduces a fictitious target whose location ahead of to the robot is defined by a constant *c*. This constant is calculated with the maximum linear and angular speed as $\frac{2V^M}{w^{\max}}$, known to all the robots.

Let us introduce a two-dimensional vector h_i, which defines the behavior and location of the fictitious target $\mathcal{T}$:

$$h_i(t) = \left(x_i(t) + \widetilde{x_i}(t)\right) + X_i + t\widetilde{v_i}(t) \tag{6}$$

The fictitious target coordinates x and y are calculated as:

$$\begin{aligned} g_i^x(t) &= \begin{cases} h_i(t) + c, & \text{if } x_i(t) \le h_i(t) \\ x_i(t) + c, & \text{if } x_i(t) > h_i(t) \end{cases} \\ g_i^y(t) &= \left(y_i(t) + \widetilde{y_i(t)}\right) + Y_i \end{aligned} \tag{7}$$

We form a vector that includes both x and y coordinates:

$$g_i = \begin{pmatrix} g_i^x \\ g_i^y \end{pmatrix} \tag{8}$$

In **Figure 3** below the vector $d_i(t)$ formed by the difference between the vector of each robot coordinates $z_i(t) = \begin{pmatrix} x_i(t) \\ y_i(t) \end{pmatrix}$ and the vector of the fictitious target g_i, is what we know as the distance ahead of the robot to the target at each instant of time: $d_i(t) = g_i(t) - z_i(t)$.

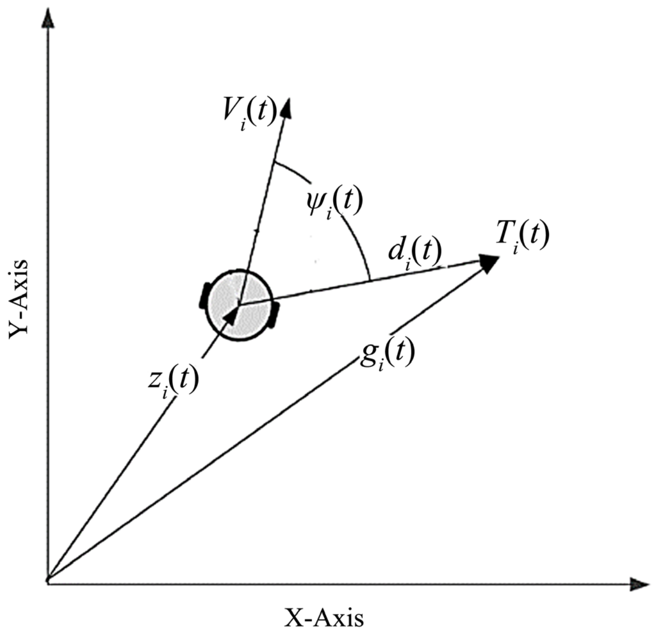

Figure 3. Geometric analysis of the vehicle and fictitious target [18].

The **Decentralized Control Law** defined in [1] depends on the relative position of the robot x and the vector $h_i(t)$ for all $i = 1, 2, \cdots, n$. The control input $w_i(t)$ depends on the maximum angular speed and the sign of the angle between the vector $V_i(t)$ formed by $\dot{x}, \dot{y}$, and the distance $d_i(t)$:

$$v_i(t) = \begin{cases} V^M, & \text{if } x_i(t) \le h_i(t) \\ V^m, & \text{if } x_i(t) > h_i(t) \end{cases} \tag{9}$$
$$w_i(t) = w^{\max} \mathcal{F}(V_i(t), d_i(t))$$

Here: $V_i(t) = \begin{pmatrix} \dot{x}_i(t) \\ \dot{y}_i(t) \end{pmatrix}$, and the Function $\mathcal{F}(V_i(t), d_i(t))$ is defined by the sign of the angle between $V_i(t)$ and $d_i(t)$.

2.2. Obstacle Avoidance Technique

Another of the challenges presented is the presence of obstacles in addition to the distributed control law described in the previous chapter. At this point, a fix circular obstacle is placed with coordinates (X_o, Y_o) and according to [19] no robot has information about the obstacle, therefore, to detect it, all robots have a built-in range sensor with 180˚ field of view to measure the proximity to the surface of the circular landmark.

The sensor detects an obstacle at certain distance r_s, and with a calculation the robot determines the range and angle to the landmark.

The robot collects all the information from the sensors and proceeds to evaluate the actions required to avoid the obstacle, in other words, it finds the route away from the collision path. The algorithm applied comes from an extensive analysis performed in documents such as [13] and [19]. As shown in **Figure 4**

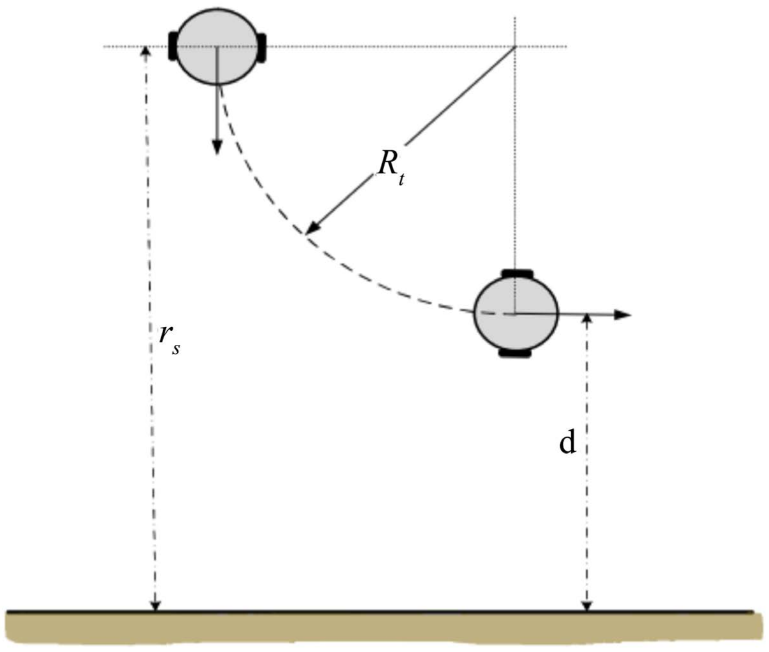

Figure 4. Sensor range detection [18].

the robot detects an obstacle once inside the range of the sensor. The robot adjusts its heading and speed to maintain a distance from the surface of the obstacle. Taking R_t as the turning radius and d the distance from the obstacle when the robot is moving parallel to the obstacle, one can calculate maximum rotation and minimum distance boundaries, respectively.

$$\mathcal{R}_t^{\max} = \frac{V^M}{w^{\max}}$$

$$d_{\min} = r_s - \mathcal{R}_t^{\max}$$

Assumptions are considered for this problem. Firstly, we can assume that the robot displacement is parallel and along the circumference perimeter. Secondly, we assume that the obstacle is considerably big to represent the surface of the obstacle as a flat long rectangle. At this point, the sensor detects at a distance r_s the obstacle and taking as reference the center of the robot there will be an angle between the heading of the vehicle and the ray that the sensor emits. This difference is considered the desired avoidance angle and can be calculated as:

$$\varphi_o = \sin^{-1}\left(\frac{d_o}{r_s}\right) \tag{10}$$

Where, d_o is the farthest point that the sensor can detect from the central point of reference of the robot coordinates. If there is a curved obstacle the robot should keep a constant distance from the surface of the landmark as explained before; however, in this case, the sensor detects that the angle of avoidance φ is greater than the desired angle φ_o, with a difference:

$$\Delta\phi = \left|\varphi - \varphi_o\right| \tag{11}$$

The angular rate of change is what the robot needs to correct to preserve the desire distance from the obstacle. In **Figure 5**, we can see in detail the geometrical analysis when the robot detects a rounded obstacle, maintaining a constant distance d_o. The angle between the heading of the robot and the virtual representation of the sensor ray for a flat surface is φ_o defined as the desired avoidance angle. At this point as the surface is not flat but convex the robot needs to rotate $\Delta\phi$ towards the obstacle to preserve a distance d_o from it.

Again, we can consider the existence of a fictitious target $\mathcal{T}$, located at a distance to the robot denoted by r_s and an angle $\Delta\phi$ which becomes the rate of change necessary to keep a distance d_o from the obstacle while avoiding it. Given some equalities and according to **Figure 6** one can see that the distance d and d''

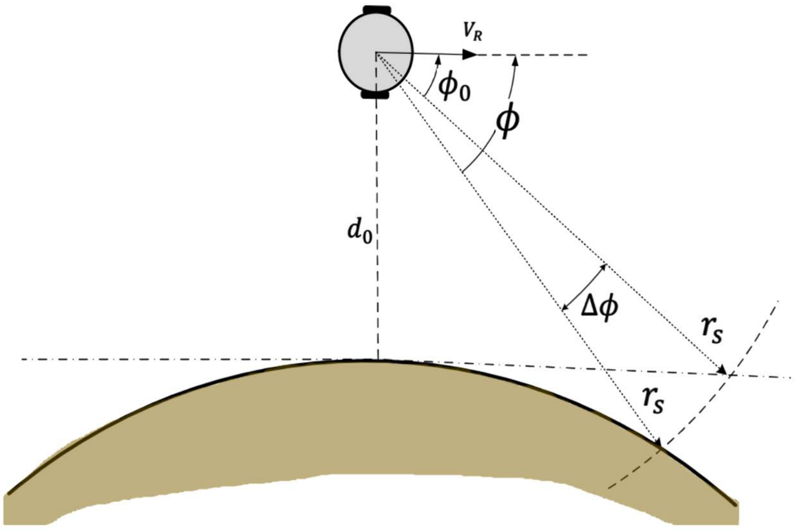

Figure 5. Round obstacle [18].

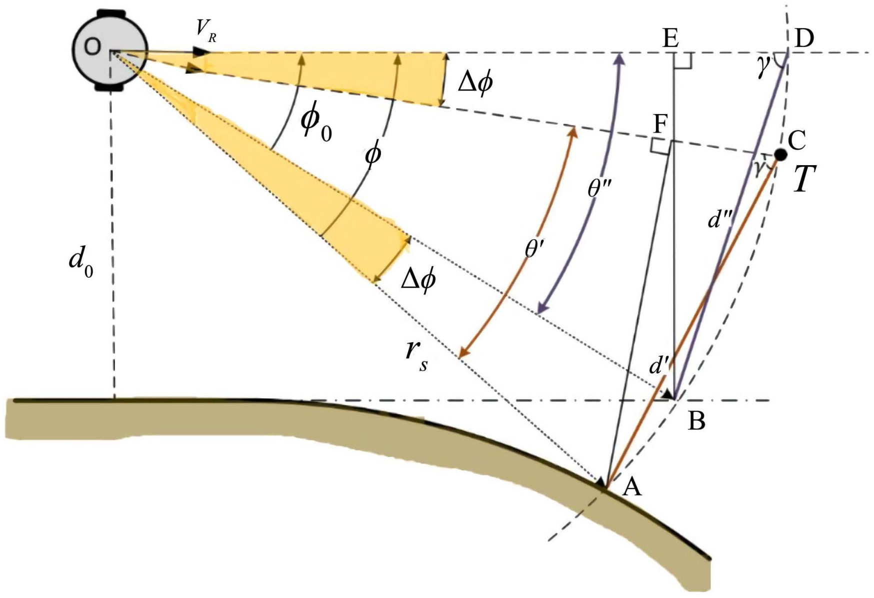

Figure 6. Geometric representation on a rounded surface [18].

are equal since the angles projected θ' and θ'' are equal in value as well. Furthermore, the segment $\overline{AB}$ as a result of the union of the respective points is equal to the segment $\overline{CD}$. The angle formed by the points $\measuredangle ODB = \measuredangle OCD = \gamma$ that in turn shows the equality of the triangles formed by the points *BED* and *AFC*. Therefore, the distances represented by the segment *AF*, which is the distance from the vehicle to the obstacle is equal to the segment $BE = d_o$ [18].

If *C* is the point where the fictitious target is located, one can determine that the robot will maintain a constant distance from the obstacle. If the obstacle has a sudden change in the shape, we can see that the fictitious target will change its location to maintain a fixed distance from the obstacle and readjust robot heading (**Figure 7**).

In that order, the robot needs to rotate $\Delta\phi$ degrees to maintain the distance d_o, for both scenarios: the flat and convex surfaces. The proposed control rule, like the one described in the previous chapter, is designed to calculate the linear and angular speed necessary to move the robot away from the collision course around the obstacle [18]:

$$\begin{aligned} v_i(t) &= V^M \\ w_i(t) &= w^{\max}\operatorname{sign}(\psi_i(t)) \\ \psi_i(t) &= \begin{cases} 1 & \text{if } \theta < \theta_o \\ 0 & \text{if } \theta = \theta_o \\ -1 & \text{if } \theta > \theta_o \end{cases} \end{aligned} \tag{12}$$

The sign of the angle and the constant maximum angular velocity are provided by the control rule described in the previous chapter.

3. Simulation Results and Discussion

3.1. Formation Building Algorithm

Numerical simulation results are presented for the control law and consensus

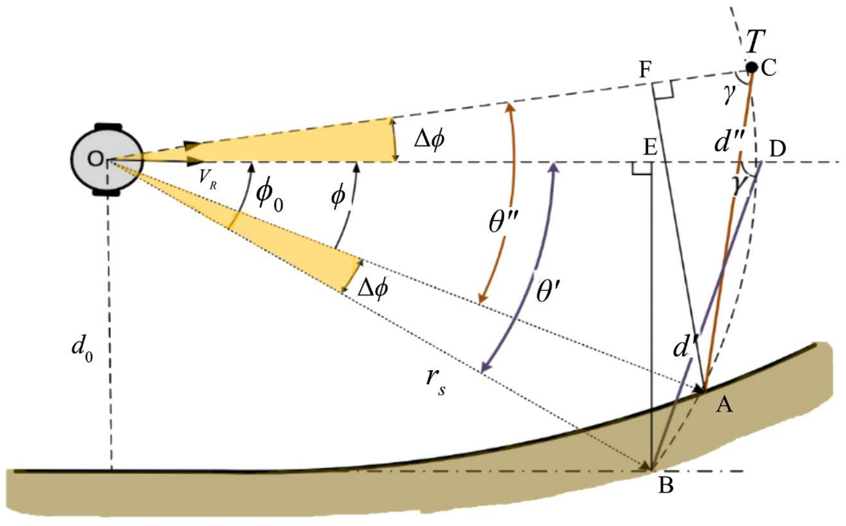

Figure 7. Sudden change in shape [18].

variables algorithm under analysis (3) and (9). MATLAB® a well-known programming environment and numerical analysis tool was used to simulate and interpret the Formation Building Algorithm presented in [1]. The number of robots n is adjustable and can be changed to a large-scale system. For simplicity, the number of agents in the simulation is kept as five. The configuration parameters $\mathcal{C}$ and maximum and minimum control inputs $w_i(t)$ and $v_i(t)$ were defined along with the time span t, control variable c and, communication range r_c (Table 1).

Thus,

$$-2 \le w_i(t) \le 2$$
$$0.2\ \text{m/s} \le v_i(t) \le 1.5\ \text{m/s}$$
$$w^{\max} > 0 \ \text{ and } \ 0 < V^m < V^M$$

The formation configuration vector C was defined for both shapes as: (Table 2).

Simulations for the Line and Delta formation are shown in Figure 8 and Figure 9 by applying the configuration parameters and constraints to each node in the system.

Table 1. Control Input constraints and configuration settings.

Parameter	Configuration settings				
	t	V^M	V^m	$c = \frac{2V^M}{w^{\max}}$	r_c
value	50 s	1.5 m/s	0.2 m/s	1.5	2 m

Table 2. Configuration vector.

Parameter	Line shape	Delta shape
value	$\mathcal{C} = [0\ -2; 0\ -1.5; 0\ -1; 0\ -0.5; 0\ \ 0]$	$\mathcal{C} = [0\ -1.5; 1\ -1; 2\ -0.5; 1\ \ 0; 0\ \ 0.5]$

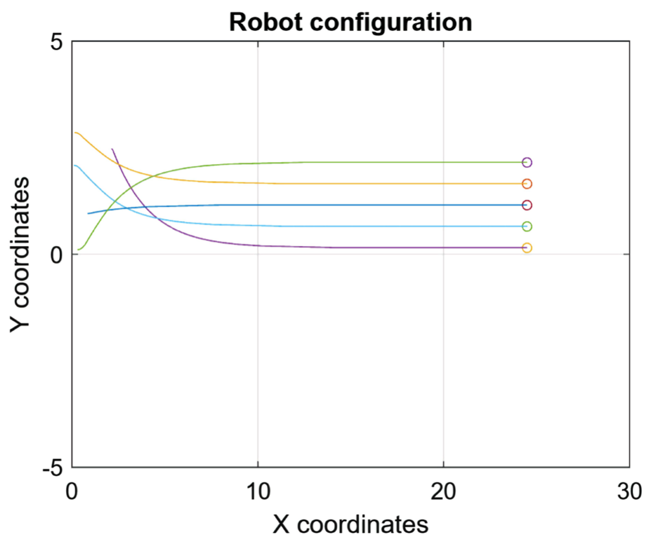

Figure 8. Line formation.

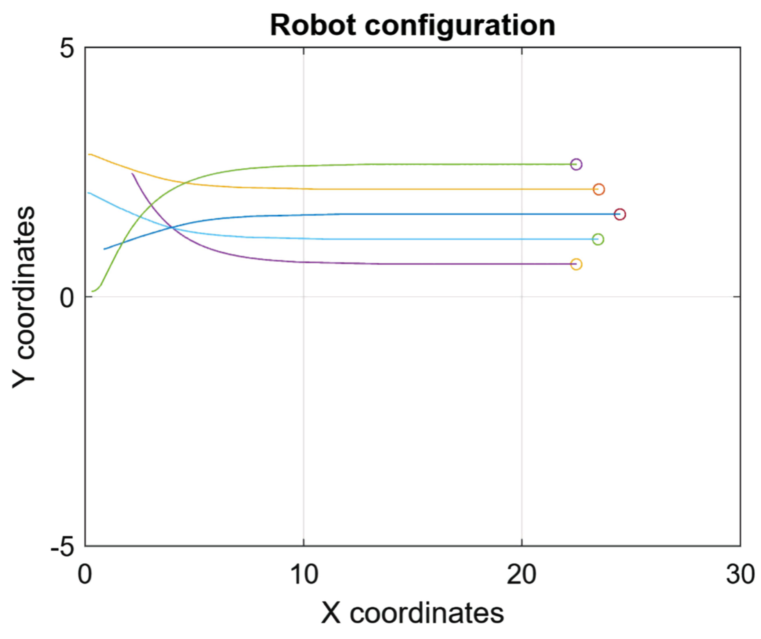

Figure 9. Delta formation.

According to [1], the distributed Control Law satisfies that $\lim_{t\to\infty}\theta_i(t)=0$. The heading of all robots is zero, which means that all the ground vehicles are moving in the same direction parallel to the *x*-axis, maintaining the same speed and the shape of the formation defined by the configuration vector $\mathcal{C}$ (**Figure 10**).

In Lemma 3.1 [1], following the assumption that there exists time intervals which are bounded and non-empty where the union of each vertex in the coordinate plane is connected through a set of lines or edges that forms an undirected graph, and that the initial condition of the heading of the vehicle is bounded to $\theta(0)\in[0,\pi)$ for robot *i*, we can say that eventually the limit of the sum of the pose of each robot and its respective consensus variable when the discrete time k tends to infinite is a constant value X_o and Y_o for the X and Y coordinates, respectively, as shown in **Figure 11** and **Figure 12** (**Table 3**).

The same is seen in the consensus heading and speed of the robot that converges to a constant value when the discrete time k tends to infinite as seen in **Figure 13** and **Figure 14** (**Table 4**).

3.2. Obstacle Avoidance

Now, for obstacle avoidance, we assume that the robot is equipped with a set of range sensors. Those sensors help the robot to determine the distance from a fixed round obstacle placed at (X_o,Y_o) with a radius r_o. According to the obstacle avoidance algorithm the robot maintains a desired angle, which comes from the relation between the heading of the vehicle and the projected line of the sensor range, given that the robot should maintain a desired distance from the obstacle. We implemented a set of equations for the intersection points between the obstacle surface and the radial detection range of the sensor (**Figure 15**).

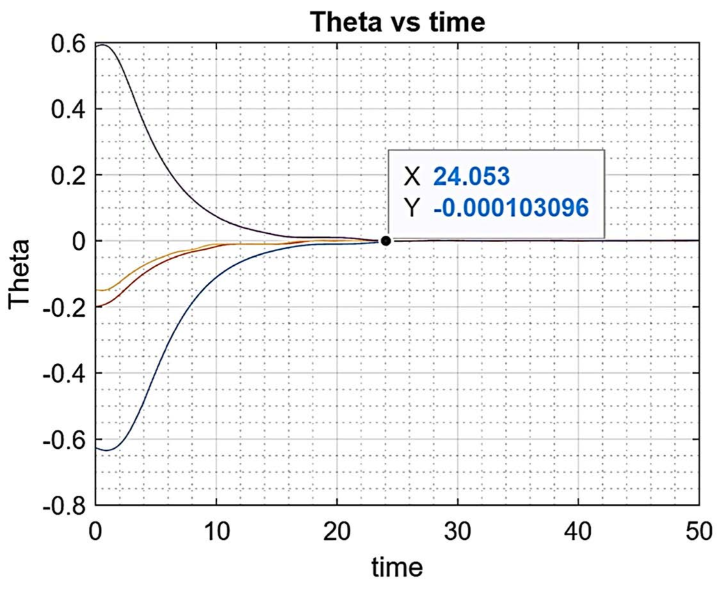

Figure 10. Limit of theta when time tends to infinity.

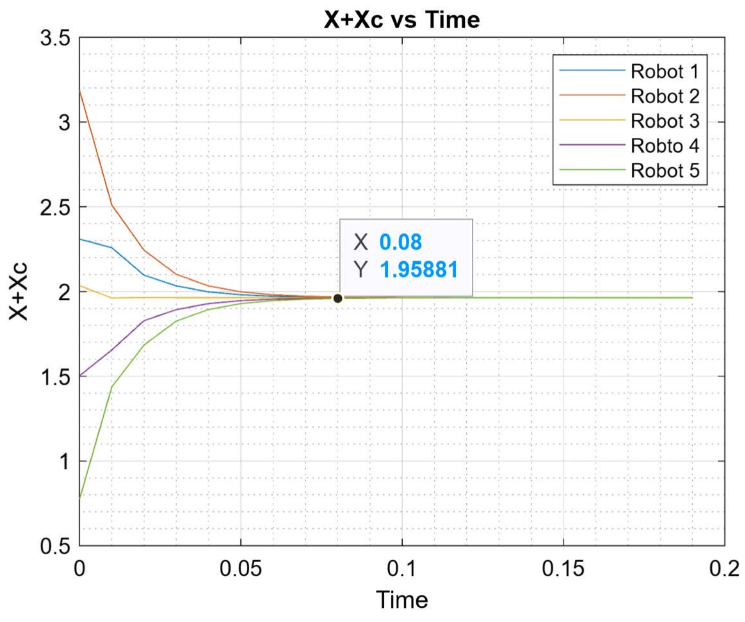

Figure 11. $\lim_{k\to\infty}\left(\widetilde{x_i}(k)+x_i(k)\right)$.

Table 3. Limit of $\widetilde{x_i}(k)+x_i(k)$ and $\widetilde{y_i}(k)+y_i(k)$.

Parameter	$\lim_{k\to\infty}\left(\widetilde{x_i}(k)+x_i(k)\right)=\widetilde{X_o}$	$\lim_{k\to\infty}\left(\widetilde{y_i}(k)+y_i(k)\right)=\widetilde{Y_o}$
value	1.95881	2.06918

Table 4. Limit of $\widetilde{\theta_i(k)}$ and $\widetilde{v_i(k)}$.

Parameter	$\lim_{k\to\infty}\widetilde{\theta_i(k)}=\widetilde{\theta_o}$	$\lim_{k\to\infty}\widetilde{v_i(k)}=\widetilde{V_o}$
value	1.52412 rads	0.388873 m/s

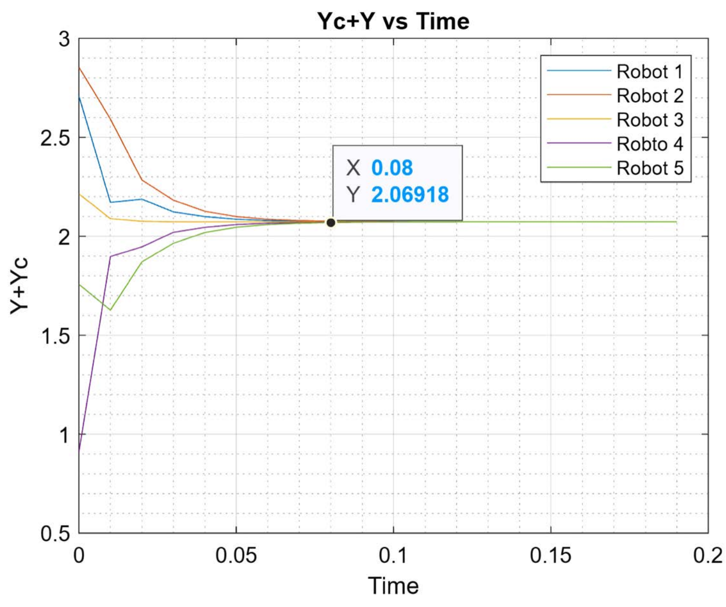

Figure 12. $\lim_{k \to \infty}\left(\widetilde{y_i}(k)+y_i(k)\right)$.

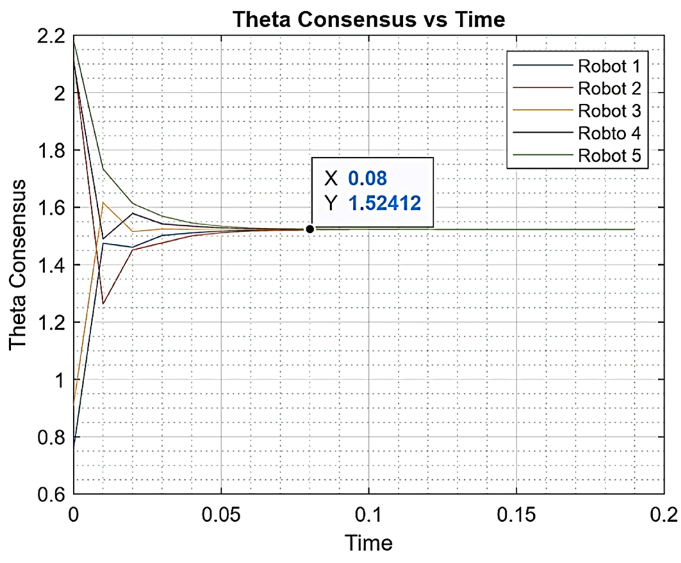

Figure 13. $\lim_{k \to \infty}\widetilde{\theta_i(k)}=\widetilde{\theta_o}$.

The first of the equations finds the range of the radial sensor about the robots' instant position (x_r, y_r):

$$r_s^2=(X-x_r)^2+(Y-y_r)^2 \tag{13}$$

The surface of the obstacle is defined by:

$$r_o^2=(X-x_o)^2+(Y-y_o)^2 \tag{14}$$

By solving these two Equations (13) and (14), we find the intersection points, where (x_r, y_r), and (x_o, y_o) are the robot and obstacle coordinates, respectively. Moreover, by expanding and subtracting them we get:

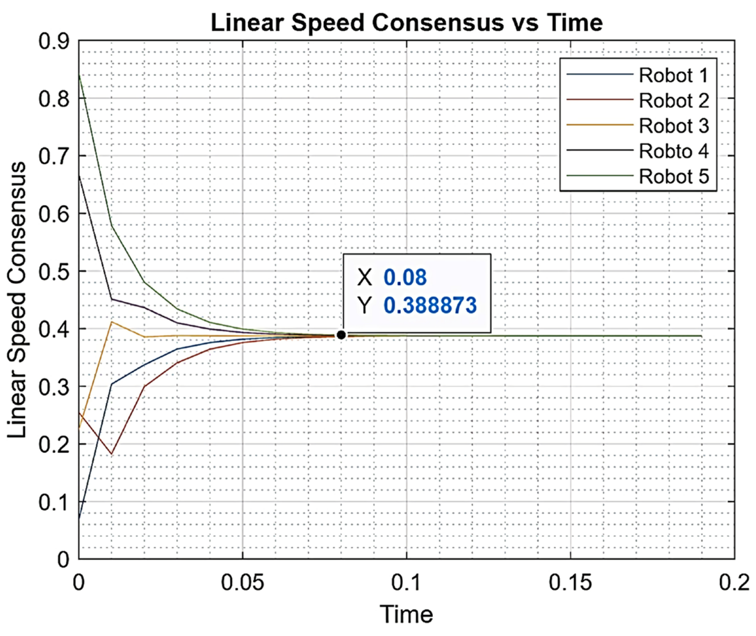

Figure 14. $\lim_{k\to\infty}\widetilde{\theta_i(k)}=\widetilde{\theta}_o$.

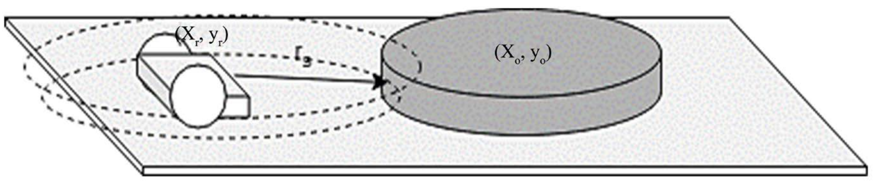

Figure 15. Robot sensor range and obstacle representation.

$$-2X(x_r-x_o)-2Y(y_r-y_o)=(r_s^2-r_o^2)-(x_r^2-x_o^2)-(y_r^2-y_o^2) \tag{15}$$

Solving 15 for Y we get:

$$Y=-X\frac{x_r-x_o}{y_r-y_o}+\frac{-(r_s^2-r_o^2)+(x_r^2-x_o^2)+(y_r^2-y_o^2)}{2(y_r-y_o)} \tag{16}$$

Substituting Equation (16) in Equation (15), we get the equation for the intersection point in X only:

$$\begin{aligned}&\left[1+\frac{(x_r-x_o)^2}{(y_r-y_o)^2}\right]+\left[\left(-2x_o+\frac{2y_o(x_r-x_o)}{(y_r-y_o)}\right)\right.\\&\left.+\left((x_r-x_o)\left(\frac{(r_s^2-r_o^2)+(x_r^2-x_o^2)+(y_r^2-y_o^2)}{(y_r-y_o)^2}\right)\right)\right]X\\&+x_o^2+y_o^2-r_o^2+\left(\frac{-(r_s^2-r_o^2)+(x_r^2-x_o^2)+(y_r^2-y_o^2)}{2(y_r-y_o)}\right)^2\\&+y_o\frac{(r_s^2-r_o^2)-(x_r^2-x_o^2)-(y_r^2-y_o^2)}{(y_r-y_o)}=0\end{aligned} \tag{17}$$

with the use of the function "roots" implemented in MATLAB® the polynomial Equation (17) can be solved. The solution indicates whether the sensor detects an obstacle or not. If the solution is real it means there is a detection, while it does not if it is conjugated. The value obtained should be always positive ahead in front of the robot to tell them that a control input needs to be applied to avoid the obstacle maintaining a fixed distance d_o from it. The robot rotates counter-clockwise if its position is below the obstacle, and clockwise if the robot is located above the obstacle (Figure 16).

We can see that for both the response of the Distributed Control Law together with Formation Building Algorithm and the obstacle avoidance technique is considerably rapid and adjusts the parameters in a few seconds. The results let us claim that the system is globally stable with random initial conditions around the configuration vector *C*. In addition, it is said that the limit of the difference in the position between robots, when the time tends to infinite is equal to the difference in the values contained in the formation configuration vector *C*. For illustrative purposes we present the plot for robot 1 and 2 (Figure 17 and Figure 18) (Table 5):

$$\lim_{t\to\infty}\left(x_i(t)-x_j(t)\right)=X_i-X_j$$
$$\lim_{t\to\infty}\left(y_i(t)-y_j(t)\right)=Y_i-Y_j$$

Table 5. Limit of $x_i(t)-x_j(t)$ and $y_i(t)-y_j(t)$.

Theorem	$\lim_{t\to\infty}\left(x_i(t)-x_j(t)\right)=X_i-X_j$	$\lim_{t\to\infty}\left(y_i(t)-y_j(t)\right)=Y_i-Y_j$
value	~0	~−0.5

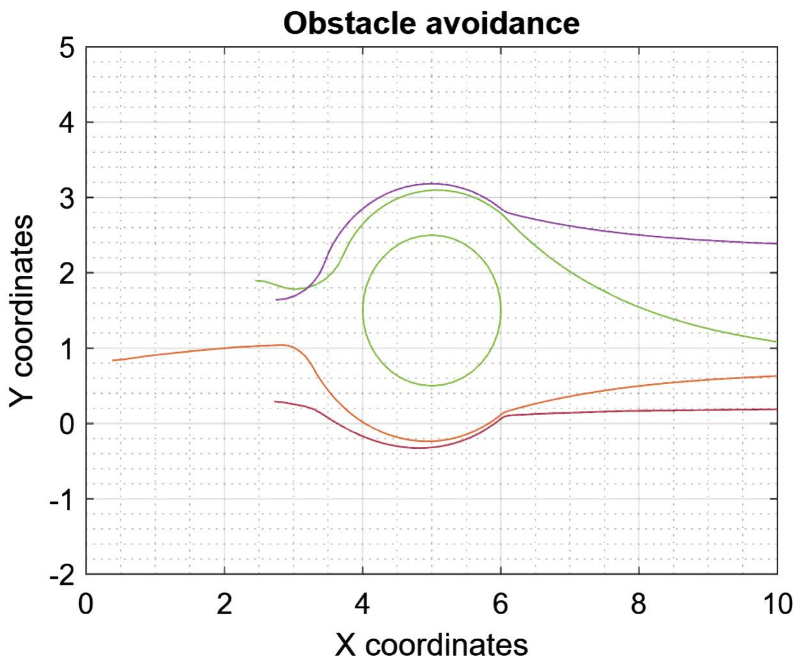

Figure 16. Obstacle avoidance simulation output.

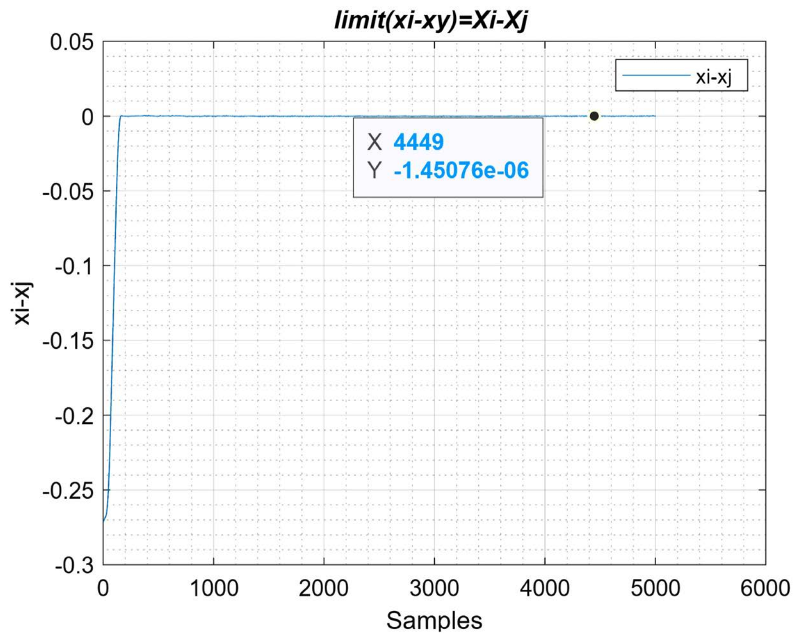

Figure 17. $\lim_{t\to\infty}\left(x_i(t)-x_j(t)\right)=X_i-X_j$.

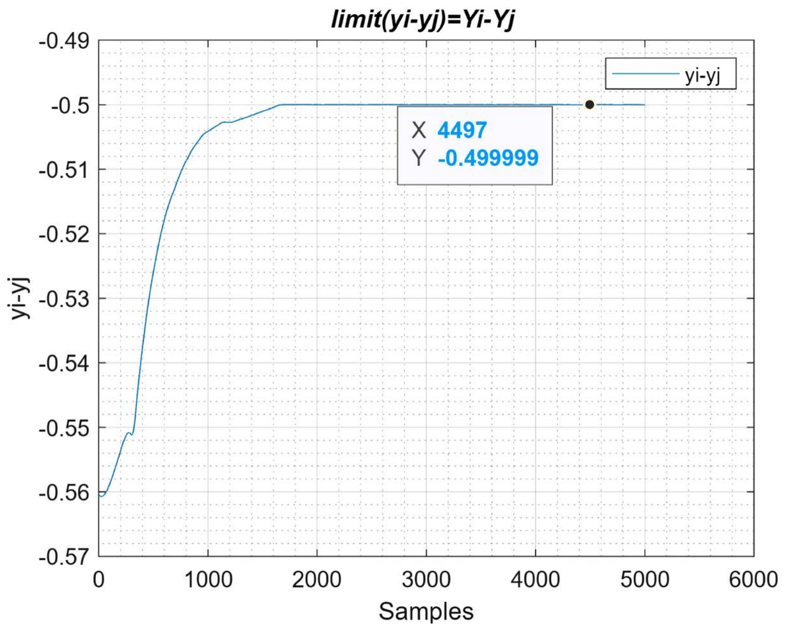

Figure 18. $\lim_{t\to\infty}\left(y_i(t)-y_j(t)\right)=Y_i-Y_j$.

4. Conclusions

To conclude, we proved through simulation that the Formation Building Algorithm (3), and Control Law (9) is Globally stable for any initial conditions around a configuration vector *C*.

The document presents the simulation for two different configurations a line and arrow-head formation. The Formation Building Algorithm applies a Distributed Control Law, where there exists a fictitious target, always located ahead far from the robot position.

The fictitious target position depends on the value of the constant variable c. If c has a small value, the system reaches faster the target position resembling a pure pursuit control technique. The Distributed Control Law applies constraints to both the angular and linear velocity, and unlike other algorithms, the one under review does not have any visible leaders in the formation.

We found and aligned with the journal paper, that all the robots reach an agreement in their position, heading and speed based on local information from other robots under the graph analysis. Also, that the control rule satisfies all theorems and assumptions.

On the other hand, the simulation of the obstacle avoidance algorithm, showed us that the robot maintains a desired distance to the obstacle while applying the required control inputs in order to change its orientation with the use of advance geometric analysis.

The group of robots start at a random initial position, apply the Consensus Algorithm to build the spatial formation, those that detect the obstacle change their path to avoid collision and once they have passed the obstacle, all robots form again the shape, always parallel to the x-axis. The robots are assumed to be equipped with linear range sensors that detect an obstacle at certain distance r_s. Further mathematical proof of the algorithms can be found in the core paper under analysis.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Savkin, A.V., Wang, C., Barazandeh, A., Xi, Z. and Nguyen, H.T. (2016) Distributed Formation Building Algorithms for Groups of Wheeled Mobile Robots. *Robotics and Autonomous Systems*, **75**, 463-474. https://doi.org/10.1016/j.robot.2015.08.006

[2] Müller, M.A., Schürmann, B. and Allgöwer, F. (2012) Robust Cooperative Control of Dynamically Decoupled Systems via Distributed MPC. *IFAC Proceedings Volumes*, **45**, 412-417. https://doi.org/10.3182/20120823-5-NL-3013.00007

[3] Bode, N.W., Wood, A.J. and Franks, D.W. (2010) Social Networks and Models for Collective Motion in Animals. *Behavioral Ecology and Sociobiology*, **65**, 117-130. https://doi.org/10.1007/s00265-010-1111-0

[4] Mach, R. and Schweitzer, F. (2003) Multi-Agent Model of Biological Swarming. *European Conference on Artificial Life*, Dortmund, 14-17 September 2003, 810-820. https://doi.org/10.1007/978-3-540-39432-7_87

[5] Flierl, G., Grünbaum, D., Levins, S. and Olson, D. (1999) From Individuals to Aggregations: The Interplay between Behavior and Physics. *Journal of Theoretical Biology*, **196**, 397-454. https://doi.org/10.1006/jtbi.1998.0842

[6] Jeremic, A. and Nehorai, A. (1998) Design of Chemical Sensor Arrays for Monitoring Disposal Sites on the Ocean Floor. *IEEE Journal of Oceanic Engineering*, **23**, 334-343. https://doi.org/10.1109/48.725229

[7] Cassinis, R., Bianco, G., Cavagnini, A. and Ransenigo, P. (1999) Strategies for Navigation of Robot Swarms to Be Used in Landmines Detection. 1999 *Third European Workshop on Advanced Mobile Robots*, Zurich, 6-8 September 1999, 211-218.

[8] Bakule, L. (2008) Decentralized Control: An Overview. *Annual Reviews in Control*, **32**, 87-98. https://doi.org/10.1016/j.arcontrol.2008.03.004

[9] Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I. and Shochet, O. (1995) Novel Type of Phase Transition in a System of Self-Driven Particles. *Physical Review Letters*, **75**, 1226-1229. https://doi.org/10.1103/PhysRevLett.75.1226

[10] Jadbabaie, A., Lin, J. and Morse, A. (2003) Correction to Coordination of Groups of Mobile Autonomous Agents Using Nearest Neighbor Rules. *IEEE Transactions on Automatic Control*, **48**, 1675-1675. https://doi.org/10.1109/TAC.2003.817537

[11] Ogren, P., Egerstedt, M. and Hu, X. (n.d.) A Control Lyapunov Function Approach to Multi-Agent Coordination. *Proceedings of the* 40*th IEEE Conference on Decision and Control*, Orlando, 4-7 December 2001, 847-851.

[12] Roy, A. (2010) Minimal Euclidean Representations of Graphs. *Discrete Mathematics*, **310**, 727-733. https://doi.org/10.1016/j.disc.2009.09.005

[13] Manchester, I.R. and Savkin, A.V. (2006) Circular-Navigation-Guidance Law for Precision Missile/Target Engagements. *Journal of Guidance*, *Control*, *and Dynamics*, **29**, 314-320. https://doi.org/10.2514/1.13275

[14] Fierro, R. and Lewis, F. (1998) Control of a Nonholonomic Mobile Robot Using Neural Networks. *IEEE Transactions on Neural Networks*, **9**, 589-600. https://doi.org/10.1109/72.701173

[15] Desai, J., Ostrowski, J. and Kumar, V. (2001) Modeling and Control of Formations of Nonholonomic Mobile Robots. *IEEE Transactions on Robotics and Automation*, **17**, 905-908. https://doi.org/10.1109/70.976023

[16] Yang, J. and Kim, J. (1999) Sliding Mode Control for Trajectory Tracking of Nonholonomic Wheeled Mobile Robots. *IEEE Transactions on Robotics and Automation*, **15**, 578-587. https://doi.org/10.1109/70.768190

[17] Barfoot, T. and Clark, C. (2004) Motion Planning for Formations of Mobile Robots. *Robotics and Autonomous Systems*, **46**, 65-78. https://doi.org/10.1016/j.robot.2003.11.004

[18] Barazandeh, A. (2016) Decentralized Autonomous Navigation Strategies for Multi-Robot Search and Rescue. Systems and Control.

[19] Savkin, A.V. and Wang, C. (2014) Seeking a Path through the Crowd: Robot Navigation in Unknown Dynamic Environments with Moving Obstacles Based on an Integrated Environment Representation. *Robotics and Autonomous Systems*, **62**, 1568-1580. https://doi.org/10.1016/j.robot.2014.05.006