2024 Tenth Indian Control Conference (ICC)
9 - 11 December, 2024
IISER Bhopal, India

Leader-Driven Formation Control for Unmanned Ground Vehicles: Simulation and Practical Implementation

Ram Milan Kumar Verma, Swati Singh, Shashi Ranjan Kumar, and Hemendra Arya

Abstract— **This paper addresses the formation control problem in multi-agent systems with a central flock leader. Within this framework, each agent of the flock exchanges its position and velocity information with their neighbors. The navigation command for the entire group is determined by the leader's configuration, while followers adopt a lattice formation based on inter-agent communication, which is limited to the sensing range of an agent. The proposed algorithm enables the leader bot to position itself while other agents maintain the desired formation. The flock can also move with a variable velocity while being in formation. The effectiveness of the proposed leader-driven flocking algorithm is demonstrated using MATLAB® and Gazebo simulation environment. Finally, the real-time implementation of the proposed algorithm on the actual JetBots demonstrates its practical applicability.**

Index Terms— **Unmanned ground vehicles, mobile robots, networked autonomous vehicles, autonomous vehicles, flocking.**

I. Introduction

With technological advancement in the field of autonomy, unmanned ground vehicles (UGVs), also referred to as mobile robots, the interest in developing control strategies to enable the UGVs to achieve a common goal autonomously has increased significantly. One of the notable works amongst the early application was discussed in [1]. Nowadays, UGVs find many applications ranging from terrain mapping, construction site monitoring, agriculture, cleaning, and security to multi-agent coordinated tasks. Various topics in distributed control, such as consensus, swarm behavior, rendezvous, and synchronization of multi-agent systems, are closely related to formation control. Over the past decade, significant efforts have been made to address the challenges in distributed control caused by information constraints. The authors in [2] discussed cooperative object transportation using the leader-follower approach using UGV-UAVs in industrial areas. In their work, the authors presented a vision-based tracking algorithm in which the follower bots maintain a certain distance and bearing from the leader.

In [3], the authors discussed a scenario involving an earthquake-damaged building, which serves as the experimental setting to generate a three-dimensional (3D) map of the top three floors. This task was accomplished through the collaboration of UAVs and UGVs. In [4], the authors used a UAV that flew ahead of the UGV to provide geo-referenced 3D geometry to navigate safely, avoiding obstacles (holes and ditches). The experimental results were threefold: the first experiment used only UGV onboard sensors to navigate the area. In the second experiment, the UGV used a traversability map already captured by the UAV and its onboard sensors. The main scenario where the UAV flew ahead of the UGV was validated through simulations, but some experimentation was carried out with the UAV manually piloted. A follow-up of the previous work can be found in [5]. To overcome the UGV sensors' limitations in detecting negative obstacles (holes, cliffs, etc.), another approach can be found in [6]. Authors in [7] proposed formation control of mobile robots, assuming a tree sensing structure generally leading to cascade interconnection structures of the closed-loop systems. In [8], the authors proposed distributed nonlinear controllers for leader-following formation control of unicycle robots without global position measurements. Nonlinear small-gain design methods were also used to deal with the problem caused by the non-holonomic constraint of the unicycle robot and yield simple conditions for practical implementation. In [9], the authors proposed a control Lyapunov function-based approach to achieve the desired formation of ground vehicles using a unicycle model. The authors in [10] proposed a constructive design method for formation tracking control and collision avoidance of unicycle robots. In [11], the artificial potential-based approach was studied for flocking control of non-holonomic mobile robots. They used an artificial potential-based approach in the flocking control for multi-agent systems, which was focused on migration and waypoint navigation problems.

Although numerous works in the existing literature used the flocking approach to achieve coordination and collision avoidance of UGVs, they did not implement it on actual ground robots. In this work, we propose control strategies for flocking and waypoint navigation of a group of agents. A virtual leader represents either the stationary destination or a moving reference point on the line joining waypoints that are being traversed by the group. Following the approach outlined in [12], we revisit the design of gradient-based control laws within the artificial potential framework, which effectively addresses inter-agent collision avoidance and velocity consensus. The main contributions of the paper are thus summarized below:

- First, we develop a simple and effective flocking algorithm for the agents utilizing a double integrator

R. M. K. Verma, S. Singh, S. R. Kumar, and H. Arya are with the Department of Aerospace Engineering, Indian Institute of Technology Bombay, Powai, Mumbai 400076, India. (e-mails: rmverma@iitb.ac.in, swatisingh@aero.iitb.ac.in, srk@aero.iitb.ac.in and arya@aero.iitb.ac.in).

This work was supported in part by the Ministry of Electronics and Information Technology, India, via Capacity building for Human Resource Development in Unmanned Aircraft Systems (Drone and related technology) - Guidance, Navigation, and Control Algorithms and Simulations (Reference No: L-14011/29/2021-HRD).

kinematic of the JetBot.

- We propose an algorithm that ensures that the agents' relative positions are stationary within the group. Our control strategy ensures that the group's velocities achieve consensus, prevent inter-agent collisions, and lead the group's configuration toward a local minimum of the artificial potential field.
- Unlike the existing strategies, which consider the centroid of the flock to generate the navigation command, the proposed strategy only requires information about the leader states and desired trajectory or waypoints to generate the navigation command.
- With this proposed strategy, the leader can be present at the desired position or trajectory while the teammates can be spread around it in a lattice structure.
- Most of the existing works assume the bot velocity to be constant (for example [13], [14]). However, In our work, we consider it to be time-varying, which brings our strategy closer to practice and more suitable for real-time applications.

The remainder of the paper is structured as follows: In Section II, we introduce the robot kinematic model used in this work, followed by formulating the problem statement and some of the preliminaries. We present the leader-based flocking algorithm and waypoint/trajectory tracking in Section III. This is followed by performance studies in the MATLAB® and implementation of the proposed approach in Gazebo environments in Section IV. Finally, we implement our algorithm on actual JetBots, and the results are discussed in Section IV-C, followed by concluding remarks.

II. Background and Problem Description

In this section, we present the kinematic model of the ground bots, necessary preliminaries, and the problem statement.

Consider the scenario where a collective system of N mobile robots are randomly placed in close proximity to each other on a planar surface, and there is one leader bot amongst all of them. Before we formally present the problem, let us first put forward the equations governing the motion of bots under the applied control inputs. To define the system's state, we utilize two reference frames: the inertial frame OXY and the body frame ox_by_b, fixed to the bot as shown in Fig. 1. The x_b-axis of the bot extends from aft to forward, while y_b points towards the left. The bot's orientation, or heading, is characterized by the angle between the X-axis and the x_b-axis. The kinematics of the ground robot in the 2D plane are described by the following equations:

$$\dot{x}_i = v_i \cos\theta_i, \tag{1a}$$
$$\dot{y}_i = v_i \sin\theta_i, \tag{1b}$$
$$\dot{\theta}_i = \omega_i, \tag{1c}$$
$$\dot{v}_i = a_i, \tag{1d}$$

where the subscript i represents i^{th} agent. $q_i = \begin{bmatrix} x_i & y_i \end{bmatrix} \in \mathbb{R}^2$, and $\theta_i \in \mathbb{R}$ are the position in the 2D plane and heading of the bot with respect to the world frame OXY as shown in Fig. 1. The symbols $v_i \in \mathbb{R}$, $\omega_i \in \mathbb{R}$, and a_i denote the linear velocity, angular velocity, and the linear acceleration of the bot. The bot's motion is controlled by commanding

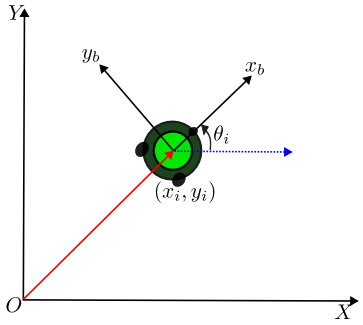


Fig. 1: Reference frame associated with the bot's motion.

v_i and ω_i. We now state some assumptions which we have taken in our work.

Assumption 1. v_i is taken to be along the body axis x_b, and there is no skidding.

Assumption 2. The initial position of the agents is such that there is no fragmentation of the flock. That is, the agents are within the interaction range of other agents so that they can maintain the lattice form.

Assumption 3. The members of the flock stay closer to the leader or other members of the group as the limit on sensing capabilities dictates the interaction range of the bots.

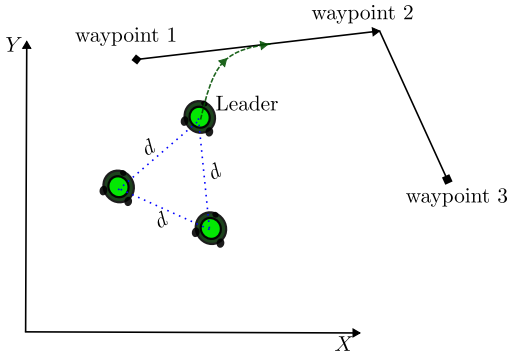


Fig. 2: Schematic of waypoint followed by a flock of bots with the leader.

Problem 1. *The objective is to develop a formation control algorithm where multiple ground bots (say N) maintain an equal distance from their neighbors and move along with the flock leader without collision, as shown in Fig. 2 while traversing the predefined waypoints or desired trajectory. Mathematically, the objective is to find v_i and ω_i so that the bot positions maintain*

$$\left\| q_j - q_i \right\| = d \quad \forall\ j \in N_i(q),$$

to form a lattice structure while simultaneously following the leader. Here, N_i denotes the neighbors of the i^{th} agent.

Before proceeding with the controller design, we briefly discuss some of the terminologies that aid in the development.

A. Flocking

Flocking is a phenomenon of collective behavior in which many agents interact with their neighbors based on limited information about them. The three rules of flocking [15] by Reynolds are following:

1) Flock centering: attempt to be close to flock-mates; also called as *cohesion*.
2) Collision Avoidance: attempt not to go too close to the flock-mates to avoid a collision; also called *separation*.
3) Velocity consensus: trying to move with the same velocity as their neighboring flock-mates; also called *alignment*.

Reynolds's rules are subject to broad interpretation, and there are several ways to implement them. In this work, we present the algorithm to enforce the agents to be equidistant from each other, forming a lattice structure. There is one designated leader of the flock; the navigation command is generated solely based on the state of the leader and the desired trajectory to be followed.

B. Algebraic graph theory

The interaction topology of a networked dynamical system is modeled via graphs. Throughout this work, we will assume that the graphs are connected and undirected.

Definition 1. A graph is an ordered pair $\mathcal{G} = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V}$ denotes the node set of $\mathcal{G}$ with $\mathbb{V} = \{v_1, v_2, v_3, ..., v_n\}$, and $\mathbb{E} \subseteq \mathbb{V} \times \mathbb{V}$ denotes the edge set of $\mathcal{G}$ with $\mathbb{E} \subset \{\{v_i, v_j\} | v_i, v_j \in \mathbb{V} \text{ and } i \neq j\}$, corresponding to the communication links among the agents [16].

Definition 2. Nodes $v_i, v_j \in \mathbb{V}$ are adjacent if $v_i, v_j \in \mathbb{E}$.

Definition 3. For an undirected graph with the vertex set $\mathbb{V}$, we have the adjacency matrix, $\mathcal{A}(\mathcal{G})$, given by

$$\mathcal{A}(\mathcal{G}) : a_{ij} := \begin{cases} 1 & \text{if } \{v_i, v_j\} \in \mathbb{E} \\ 0 & \text{otherwise.} \end{cases}$$

Definition 4. The incidence matrix for a graph with rows and columns indexed by the vertices and the edges, respectively, is given as

$$\mathcal{E}(\mathcal{G})_{ik} := \begin{cases} 1 & \text{if an edge } e_k \text{ emanates from node } v_i; \\ -1 & \text{if an edge } e_k \text{ terminates at node } v_i; \\ 0 & \text{otherwise.} \end{cases}$$

III. Controller Design

In this section, we present the modified form flocking algorithm inspired by [12]. One should note that the validation of the algorithm on actual ground robots is equally important. Therefore, we have also validated the algorithm using ROS/Gazebo and implemented it on actual ground robots.

A. Flocking Algorithm

The objective in Problem 1 is achieved by designing control with three essential features: cohesion, separation, and velocity matching. Let q_i and v_i denote the position and velocity of the agent in 2D plane for i^{th} agent.

- *Gradient-based term*: This term of the control input is responsible for attracting/repelling the neighboring bots to maintain d distance using the minimum artificial potential energy concept.

$$u_{i,\text{grad}} = \sum_{j=1}^{n} \phi_\alpha \left(||q_j - q_i||_\sigma \right) \boldsymbol{n}_{ij}, \tag{2}$$

where the terms $\phi_\alpha(z)$, $\phi(z)$ and $\boldsymbol{n}_{ij}$ are described by

$$\phi_\alpha(z) = \rho_h(z/\|r\|_\sigma)\phi(z - \|d\|_\sigma), \tag{3}$$

$$\phi(z) = \frac{1}{2}\left[(a+b)\frac{(z+c)}{\sqrt{1+(z+c)^2}} + (a-b) \right], \tag{4}$$

$$\boldsymbol{n}_{ij} = \nabla \left\| q_j - q_i \right\|_\sigma = \frac{(q_j - q_i)}{1 + \epsilon \left\| (q_j - q_i) \right\|_\sigma}. \tag{5}$$

The notation $\|\cdot\|_\sigma$ used in the equations, called as $\sigma -$ norm, is defined as

$$\|z\|_\sigma = \frac{1}{\epsilon}\left[\sqrt{1 + \epsilon \|z\|^2} - 1 \right]. \tag{6}$$

In Equation (3), the $\rho_h(z)$ called bump function is defined as

$$\rho_h(z) = \begin{cases} 1, & \text{for } z \in [0, h) \\ \frac{1}{2}\left[1 + \cos\left(\pi \frac{z-h}{1-h} \right) \right], & \text{for } z \in [h, 1] \\ 0, & \text{otherwise} \end{cases}, \tag{7}$$

where $h \in (0, 1)$ is a tunable parameter that controls how smoothly the function diminishes from 1 to 0. The Equation (4) is uneven sigmoidal function with parameters a, b, c satisfying $0 < a \leq b$, $c = (a-b)/2\sqrt{ab}$. The $\phi_\alpha(z)$ function in Equation (3) can be integrated from $\|d\|_\sigma$ till z to get the potential at z.

- *Consensus term*: This term is responsible for forcing all the agents to reach a velocity consensus, which is necessary to maintain the lattice structure as the flock migrates.

$$u_{i,\text{con}} = \sum_{j=1}^{n} a_{ij}(q_i, q_j)(p_j - p_i), \tag{8}$$

where

$$a_{ij}(q_i, q_j) = \rho_h\left(\frac{||q_j - q_i||_\sigma}{\|r\|_\sigma} \right) \in [0, 1], \quad j \neq i. \tag{9}$$

- *Navigation term*: This term guides the flock to travel to designated waypoints or any arbitrary reference path or trajectory. In this paper, we propose that one of the agents in the flock be the leader. The navigation term of the control input is decided based only on the states of

the leader and communicated to all the other members. Hence, the proposed navigation command is given by

$$u_{i,\text{nav}} = u_l = -c_1(q_l - q_r) - c_2(p_l - p_r), \tag{10}$$

where c_1 and c_2 are gains for position and velocity errors, respectively; subscript l stands for the leader.

Hence, the final control law can be obtained as the weighted sum of the three components

$$u_i = k_{\text{grad}}\, u_{i,\text{grad}} + k_{\text{con}}\, u_{i,\text{con}} + u_l. \tag{11}$$

Now, we need to transform $u_i = \begin{bmatrix} u_{xi} & u_{yi} \end{bmatrix} \in \mathbb{R}^2$ to v_i and ω_i, which can be obtained with the help of feedback linearization as shown in [8], and are given by

$$\begin{bmatrix} a_i \\ \omega_i \end{bmatrix} = \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i / v_i & \cos\theta_i / v_i \end{bmatrix} \begin{bmatrix} u_{xi} \\ u_{yi} \end{bmatrix}. \tag{12}$$

Now, v_i can be obtained by performing integration of linear acceleration, that is, $v_i = \int a_i dt$ over a specified time interval.

IV. Performance Studies

In this section, we present the simulation results and results from the actual implementation on JetBots to validate the proposed leader-based formation control algorithm. The reference trajectory is generated for simulation by assuming a virtual agent moving with constant velocity along the line joining the given waypoints. The simulation and controller parameters are tabulated in Table I. The flocking distance (d)

Potential function	a	5
Potential function	b	15
Bump function parameter	h	0.6
Lattice ratio	k	1.2
$\sigma-$norm parameter	ϵ	0.1
Position gain for u_{nav}	c_1	0.5
Velocity gain for u_{nav}	c_2	1.4
Gradient gain	k_{grad}	5
Velocity consensus gain	k_{con}	50

TABLE I: Controller-related parameters.

the bots should maintain between them is 1m. The interaction range (r) of the bots, which is dependent on the sensing capability, is taken to be 1.2m. The navigation velocity with the bots should move for trajectory tracking is taken to be 0.05 m/s.

A. Implementation in MATLAB®

First, we simulated the proposed algorithm for 3 mobile robots. Here, we have assumed bot B2 as our leader. The initial positions and headings are chosen randomly using `rand` command of MATLAB®, such that they are inside the interaction range of each other. The waypoints given are $(10, 5)$and $(20, 10)$. A trajectory is generated on the line joining the leader bot to the waypoints. To follow the reference trajectory, small intermediate points are generated using a navigation speed of 0.05 m/s. A flock is formed in Fig. 4b, while the trajectory is given in Fig. 4a. As we have made bot B2 our leader, it is seen from Fig. 4a that all the bots follow bot B2 while maintaining a safe distance. Fig. 4

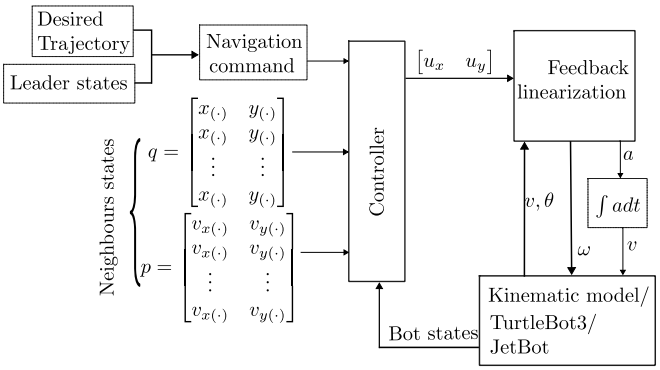


Fig. 3: Schematic of the simulation framework.

shows the motion of three bots, maintaining an equal distance from each other and following the leader. The final pattern in Fig. 4b is not a lattice because bot B1 and B3 are not within the interaction range of each other. It is seen from Fig. 4f that, indeed, the bot achieves consensus in velocity. The negative velocity means that the bot is going backward. Also, Fig. 4c shows that initially, high acceleration is required, but it drops to zero when the formation is formed.

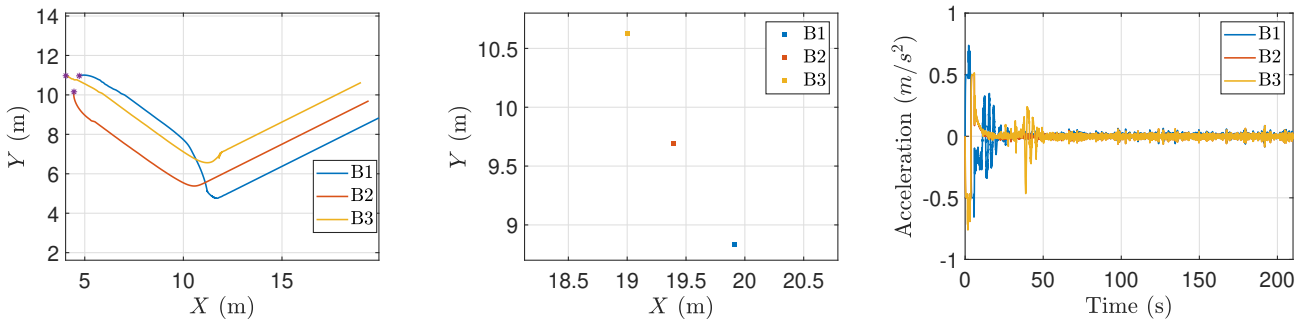
B. Implementation in ROS-Gazebo

We use a standard framework based on Robot Operating System (ROS) and Gazebo. TurtleBot3 has been used for the simulation because its model is available and is very similar to JetBot, which we used for validation.

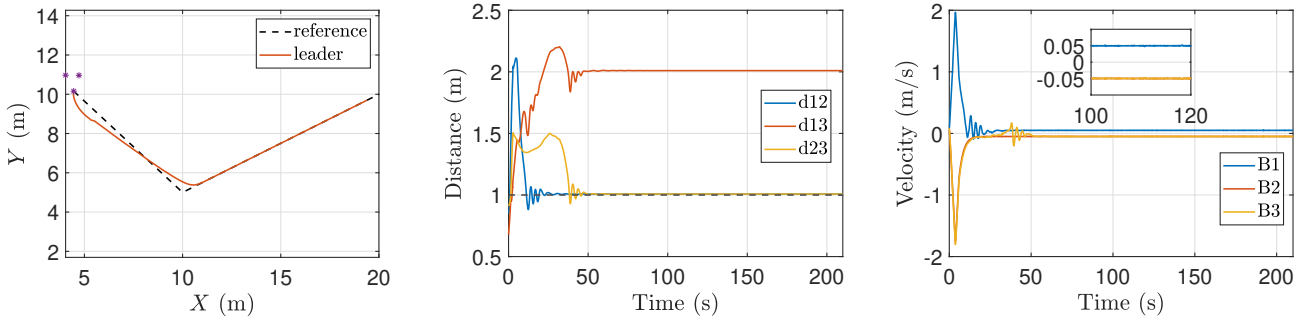
1) Setting up Environment: We need to install `ROS Noetic`, `Gazebo 11`, TurtleBot3 Simulation package from ROBOTIS (`https://emanual.robotis.com/docs/en/platform/turtlebot3/simulation/`), and other dependencies. A `catkin` workspace also needs to be set up and `build`, which contains all the scripts for the ROS nodes performing the necessary computations.

2) Simulation setup in ROS-Gazebo: Three TurtleBots are placed in an empty world in Gazebo as shown in Fig. 6a. The same algorithm is coded using Python programming language and put inside the catkin workspace already created.

Since the topic names for all bots are the same, we created different namespaces for each bot to associate the data with the bot publishing it. Each bot publishes its odometry data, which contains the position, orientation, and linear and angular speeds. Now, the algorithm subscribes to the odometry data from the bots and computes the control as per Equation (11) and transforms to the v_i and ω_i using Equation (12). Now, these commanded linear and angular velocities can be published on the `cmd_vel` topic, which will make the TurtleBots in Gazebo move as commanded. The Fig. 5 shows the results from Gazebo. The initial configuration (x, y, θ) of the three bots is $(1, 1, 1.57)$, $(1, -1, 1.57)$ and $(-1, -1, 0)$, see Fig. 6a. Now, the bots are commanded to go to the waypoint $(10, 10)$. As seen from the Fig. 5, the bots follow the leader while maintaining the formation as evident from Fig. 5b and reach the destination. The Fig. 6b shows the final pattern.

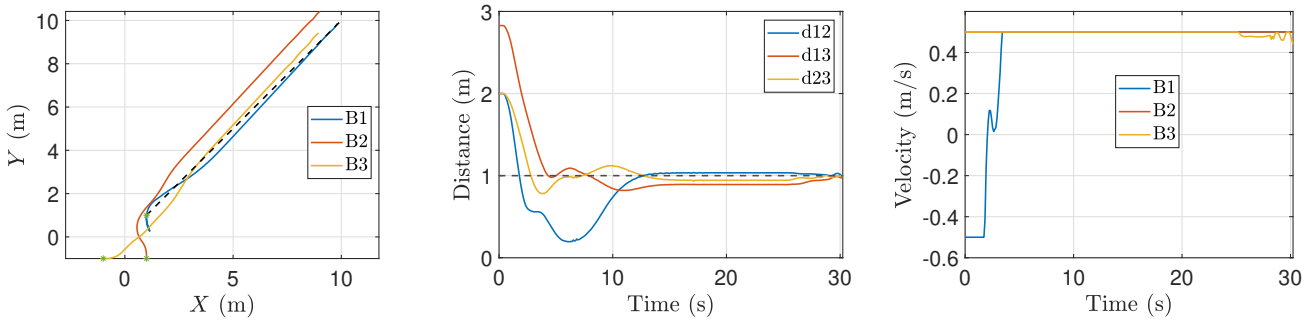


(a) Trajectories of all bots. (b) Final pattern formed by bots. (c) acceleration of bots.



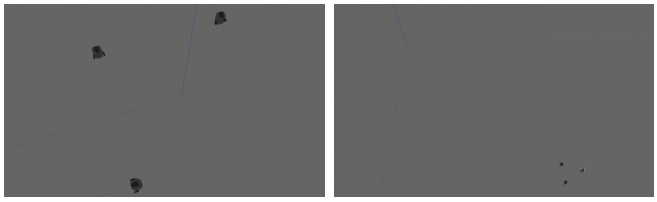
(d) Reference tracking by the leader bot. (e) Distance between the bots. (f) Velocity of bots.

Fig. 4: Figures demonstrating the trajectories, pattern formed, and control inputs for 3-Bots in MATLAB® simulation.



(a) Trajectories of all bots in Gazebo. (b) Distance between the bots in Gazebo. (c) Velocity of bots in Gazebo.

Fig. 5: Figures demonstrating the waypoint-following, interagent distance, and velocity for 3-Bots in Gazebo.



(a) Initial position. (b) Final position.

Fig. 6: Initial and final position of Turtlebots in Gazebo environment; The flock reaches $(10, 10)$. The $x - y - z$ axes are indicated in red-green-blue colors.

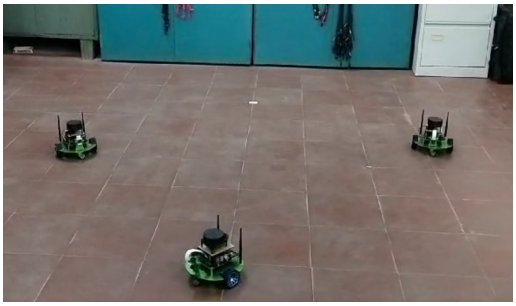
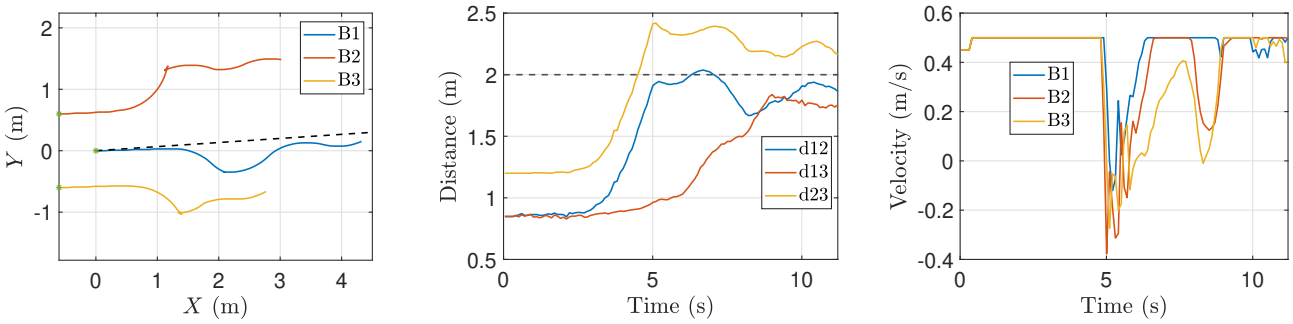


Fig. 7: Initial positioning of the JetBots.

C. Implementation on JetBots

JetBot is an open-source robot based on NVIDIA Jetson Nano https://jetbot.org/master/. It is affordable, easy to set up, and easily programmable. All the bots need to be connected to the same network. Either one of the bots or a ground station computer can be the ROS MASTER. The



(a) Trajectories of JetBots. (b) Distance between the JetBots. (c) Velocity of JetBots.

Fig. 8: Figures demonstrating the waypoint-following, interagent distance, and velocity for 3 actual JetBots.

algorithm coded in Python as used to command TurtleBots in Gazebo is put inside the `catkin` workspace. The same workspace is loaded on each JetBot with minimal modification. The JetBots used for the implementation are shown in Fig. 7. The initial placement (x, y, θ) of the three JetBots are $(0, 0, 0)$, $(0.6, -0.6, 0)$ and $(-0.6, -0.6, 0)$, respectively. Fig. 8a shows that the leader bot B1 tries to go towards the given waypoint $(4.5, 0.3)$ while the other bots follow along in formation, maintaining a distance as indicated in Fig. 8b; the legend d12 represents the distance between the bots B1 and B2. The Velocity of the bots hits the saturation limit as the waypoint is farther initially, and then it comes down as seen in Fig. 8c. Their velocity varies as they try to go towards the waypoint and maintain the formation.

V. Conclusion

This paper proposes a formation control algorithm to achieve waypoint navigation and trajectory tracking for unmanned ground vehicles. The algorithm was designed using the artificial potential field technique. To achieve the objectives of waypoint navigation or trajectory tracking, we adopted the leader-follower approach. One agent was designated as the leader, while the other agents followed, either forming a lattice structure or maintaining a predefined inter-agent distance to avoid collisions. It was observed that the proposed algorithm performed well in both the MATLAB® and ROS/Gazebo environments, as well as on actual mobile robots (JetBots). Future work aims to design and implement a formation control algorithm based solely on relative measurements and involving multiple mobile robots.

VI. Acknowledgment

The authors would like to sincerely thank Mr. Karthickeyan V. and Mr. Jash Desai for assembling and setting up the JetBots for the experiment.

References

[1] D. W. Gage, *UGV History 101: A Brief History of Unmanned Ground Vehicle (UGV) Development Efforts*. Naval Ocean Systems Center San Diego, CA, USA, 1995.

[2] E. H. C. Harik, F. Guérin, F. Guinand, J.-F. Brethé, and H. Pelvillain, "UAV-UGV cooperation for objects transportation in an industrial area," in *IEEE International Conference on Industrial Technology*, 2015, pp. 547–552.

[3] N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, K. Ohno, E. Takeuchi, and S. Tadokoro, "Collaborative mapping of an earthquake-damaged building via ground and aerial robots," *Journal of Field Robotics*, vol. 29, no. 5, pp. 832–841, 2012.

[4] T. Stentz, A. Kelly, H. Herman, P. Rander, O. Amidi, and R. Mandelbaum, "Integrated air/ground vehicle system for semi-autonomous off-road navigation," in *AUVSI Symposium, Unmaned Systems*. Carnegie Mellon University, 2002.

[5] A. Stentz, A. Kelly, P. Rander, H. Herman, O. Amidi, R. Mandelbaum, G. Salgian, and J. Pedersen, "Real-time, multi-perspective perception for unmanned ground vehicles," in *Proceedings of Unmanned Systems Symposium*. Carnegie Mellon University, 2003.

[6] B. Sofman, J. A. Bagnell, A. Stentz, and N. Vandapel, "Terrain classification from aerial data to support ground vehicle navigation," 2006.

[7] W. E. Dixon, D. M. Dawson, E. Zergeroglu, and A. Behal, *Nonlinear control of wheeled mobile robots*. Springer, 2001, vol. 175.

[8] T. Liu and Z.-P. Jiang, "Distributed formation control of nonholonomic mobile robots without global position measurements," *Automatica*, vol. 49, no. 2, pp. 592–600, 2013.

[9] P. Ogren, M. Egerstedt, and X. Hu, "A control Lyapunov function approach to multi-agent coordination," vol. 2, pp. 1150–1155, 2001.

[10] Z. Qu, J. Wang, and R. A. Hull, "Cooperative control of dynamical systems with application to autonomous vehicles," *IEEE Transactions on Automatic Control*, vol. 53, no. 4, pp. 894–911, 2008.

[11] Q. Li and Z.-P. Jiang, "Flocking control of multi-agent systems with application to nonholonomic multi-robots," *Kybernetika*, vol. 45, no. 1, pp. 84–100, 2009.

[12] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401–420, 2006.

[13] L. Consolini, F. Morbidi, D. Prattichizzo, and M. Tosques, "Leader–follower formation control of nonholonomic mobile robots with input constraints," *Automatica*, vol. 44, no. 5, pp. 1343–1349, 2008.

[14] T. Nguyen and H. M. La, "Distributed formation control of nonholonomic mobile robots by bounded feedback in the presence of obstacles," in *IEEE International Conference on Real-time Computing and Robotics*. IEEE, 2017, pp. 206–211.

[15] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, 1987, pp. 25–34.

[16] C. Godsil and G. F. Royle, *Algebraic Graph Theory*. Springer Science & Business Media, 2001, vol. 207.