



Mind the Gap! Predictive Flocking of Aerial Robot Swarm in Cluttered Environments

Giray Önür^{1,3}(✉), Ali Emre Turgut^{1,3}, and Erol Şahin^{2,3}

¹ Department of Mechanical Engineering, Middle East Technical University, Ankara, Turkey

girayo@metu.edu.tr

² Department of Computer Engineering, Middle East Technical University, Ankara, Turkey

³ Center for Robotics and Artificial Intelligence (ROMER), Middle East Technical University, Ankara, Turkey

Abstract. Flocking, coordinated movement of individuals, widely observed in animal societies, and it is commonly used to guide robot swarms in cluttered environments. In standard flocking models, robot swarms often use local interactions between the robots and obstacles to achieve safe collective motion using virtual forces. However, these models generally involve parameters that must be tuned specifically to the environmental layout to avoid collisions. In this paper, we propose a predictive flocking model that can perform safe collective motion in different environmental layouts without any need for parameter tuning. In the model, each robot constructs a search tree consisting of its predicted future states and utilizes a heuristic search to find the most promising future state to use as the next control input. Flocking performance of the model is compared against the standard flocking model in simulation in different environmental layouts, and it is validated indoors with a swarm of six quadcopters. The results show that more synchronized and robust flocking behavior can be achieved when robots use the predicted states rather than the current states of others.

1 Introduction

Flocking, the coherent motion of a group of individuals, is commonly encountered in animal societies [1] with schools of fish and flocks of birds demonstrating impressive examples of coordinated motion [10, 11]. Flocking has also been an interest in artificial systems. One of the earliest attempts to implement flocking in artificial systems is due to Reynolds [13]. In his model, flocking is modeled using repulsion, velocity alignment, and attraction behaviors. Repulsion ensures collision avoidance, velocity alignment maintains the coherent motion, and attraction keeps the flock together. Numerous flocking models have been proposed based on these interactions to describe various systems such as animal groups [5] and migrating cells [9].

In the virtual force model [16], these behaviors are generated via virtual forces exerted on the agents where the magnitude and direction of the forces depend on the local interactions between neighboring agents. The virtual force model is often used for implementing robot swarms in cluttered or confined environments due to its reactivity, and low computational complexity [7, 18]. However, these implementations frequently involve parameters that must be tuned specifically to the environmental layout. Consequently, sudden changes in the environmental layout may cause collisions or decrease flocking performance, which makes these models unreliable in real-world.

Recent studies [2, 6] claim that natural swarms, such as schools of fish or swarms of bats, use predicted future states of their neighbors rather than the current states. In accordance with this claim, in [3] it was demonstrated that the neural activity in bat's brain encodes non-local navigational information up to a few meters away from the bat's present state during both random exploration and goal-directed motion.

Inspired by the natural swarms, predictive flocking models based on distributed model predictive control (DMPC) have emerged since DMPC can handle actuation constraints of robots and optimize flocking performance [8, 14]. Previous work showed that aerial robot swarms using DMPC-based flocking model could perform safe collective motion with noisy sensor measurements in a cluttered environment [14] and in the presence of dynamic obstacles [8]. Although DMPC-based flocking models perform more robust collective motion compared to standard flocking models, their onboard implementation remains a challenge due to the computational complexity and the requirement of excessive communication between robots.

In this paper, we propose a novel flocking model based on a predictive search method that can perform safe collective motion in different environmental layouts without parameter tuning. In the proposed model, robots can sense obstacles and other robots within a limited range and only use the local information. Each robot constructs a search tree consisting of their possible future states and finds the trajectory that fulfills the flocking objectives by using a heuristic search algorithm to determine its next move.

2 Methodology

Consider a swarm consisting of N robots with a radius r_a moving in a 2D environment. Robots can sense obstacles and other robots within a limited range, r_s . At each time step, Δt , the next velocity of the robot is determined according to the updated information of the neighboring robots and obstacles. The number of neighbors and obstacles closest to robot i are indicated by M and O , respectively. Their values are limited to $M \leq M_{max}$ and $O \leq O_{max}$. The position and velocity of robot i are denoted by \mathbf{p}_i and \mathbf{v}_i . The position of the point closest to robot i on the boundary of obstacle k is indicated by \mathbf{o}_k . The distance between robot i and its j^{th} neighbor is denoted by $d_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\| - 2r_a$ whereas the distance between robot i and obstacle k is indicated by $d_{ik} = \|\mathbf{p}_i - \mathbf{o}_k\| - r_a$.

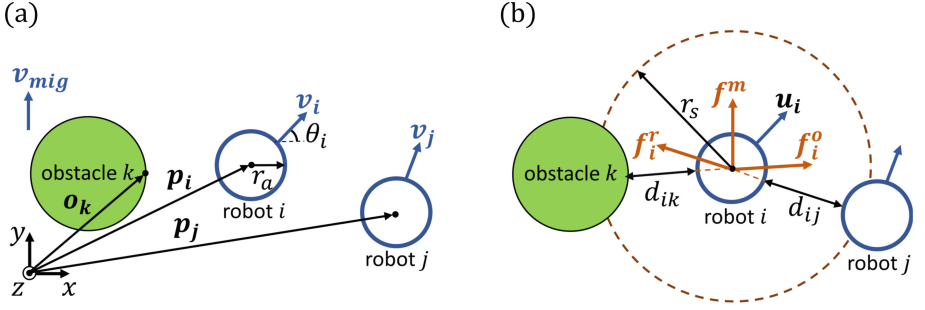


Fig. 1. Illustration of a robot swarm in a 2D environment with (a) robot's radius, heading angle, position and velocity vectors, (b) sensing range, heading, relative distances, and virtual forces.

v_{mig} is defined as the migration velocity to the robots. Each robot moves in the direction of its heading \mathbf{u}_i , with the heading angle θ_i calculated with respect to the frame of reference (Fig. 1). Unit vectors along the x, y, and z-axis of the frame of reference are denoted by $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$. The common frame of reference is used for ease of description. In the following models, a robot can calculate its next velocity using a local frame of reference and relative positions and velocities of others.

2.1 Standard Flocking Model (SFM)

We extended the Active Elastic Sheet (AES) model [4], which uses spring-like virtual interaction forces between robots, by adding an obstacle avoidance force to prevent robot-obstacle collision and a migration force to provide migration velocity. The total force acting on robot i is obtained as the sum of three forces as:

$$\mathbf{f}_i = \mathbf{f}_i^r + \mathbf{f}_i^o + \mathbf{f}_i^m \quad (1)$$

where \mathbf{f}_i^r is the inter-robot force, \mathbf{f}_i^o is the obstacle avoidance force and \mathbf{f}_i^m is the migration force (Fig. 1b).

The spring-like inter-robot force is calculated as:

$$\mathbf{f}_i^r = \sum_{j=1}^M k^r (d_{ij} - d_{eq}) \mathbf{u}_{ij} \quad (2)$$

where k^r is the inter-robot force gain, d_{eq} is the equilibrium distance and \mathbf{u}_{ij} is the unit vector directed from robot i to its j^{th} neighbor.

The obstacle avoidance force is calculated as:

$$\mathbf{f}_i^o = \sum_{k=1}^O -k^o \frac{1}{d_{ik}^2} \mathbf{u}_{ik} \quad (3)$$

where k^o is the obstacle avoidance force gain and \mathbf{u}_{ik} is the unit vector directed from robot i to obstacle k . In [7], it has been shown that as the robot approaches an obstacle, the obstacle avoidance force must increase significantly compared to the inter-robot force to avoid the robot-obstacle collision. Thus, \mathbf{f}_i^o designated as its magnitude increases exponentially as robot i gets closer to obstacle k , whereas the magnitude of \mathbf{f}_i^r increases linearly as two robots get closer to each other.

The migration force keeps the velocity of robots at the migration velocity and it is calculated as:

$$\mathbf{f}^m = k^m \mathbf{v}_{mig} \quad (4)$$

where k^m is the migration force gain.

The linear speed of the robot is computed by projecting \mathbf{f}_i onto \mathbf{u}_i and multiplying it by the linear speed gain k^l as:

$$v_i = k^l (\mathbf{f}_i \cdot \mathbf{u}_i). \quad (5)$$

The angular speed of the robot is obtained by projecting \mathbf{f}_i onto the vector perpendicular to its heading and multiplying it by the angular speed gain k^a as:

$$\omega_i = k^a (\mathbf{f}_i \cdot \mathbf{u}_i^\perp). \quad (6)$$

The linear speed is bounded between v_{min} and v_{max} whereas the angular speed is bounded between $-\omega_{max}$ and ω_{max} . Then, the linear and angular velocities of robot i are obtained as:

$$\mathbf{v}_i = v_i \mathbf{u}_i, \quad (7)$$

$$\boldsymbol{\omega}_i = \omega_i \hat{\mathbf{z}}. \quad (8)$$

2.2 Predictive Flocking Model (PFM)

The predictive flocking model searches for a trajectory that fulfills the flocking objectives only using local information. Each robot constructs a search tree consisting of nodes that contain its possible future position and velocity states where levels of the search tree represent future time steps.

Let the heading angle and speed of a parent node at the c^{th} level of the search tree be denoted by θ^c and v^c , respectively. Then, the heading angle and speed values of its child nodes at the next level of the search tree are calculated as:

$$\theta_a^{c+1} = \theta^c + a \Delta \theta \quad a \in \mathbb{Z} : -A \leq a \leq A, \quad (9)$$

$$v_b^{c+1} = v^c + b \Delta v \quad b \in \mathbb{Z} : -B \leq b \leq B \quad (10)$$

where $\Delta \theta$ and Δv are the search step parameters, A and B are the parameters that determine the number of considered reachable heading angle and speed values at the next time step, respectively. Since the calculated heading angle

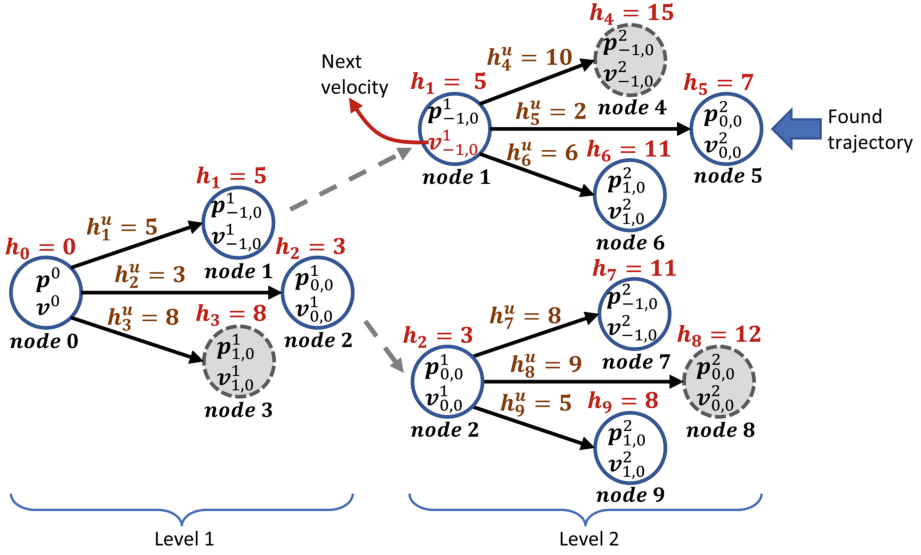


Fig. 2. Illustration of an example search tree for the parameters $\beta = 2$, $d = 2$, $A = 1$ and $B = 0$. p^0 and v^0 represent the current position and velocity of the robot. The gray nodes are pruned, and the remaining nodes are expanded. The next velocity of the robot is obtained as $v^1_{-1,0}$ which is the first velocity state of the found trajectory.

and speed values should satisfy the actuation constraints of the robot, the speed term v_b^{c+1} is bounded between v_{min} and v_{max} , and the parameters are selected as the constraints $A\Delta\theta \leq \omega_{max}\Delta t$ and $B\Delta v \leq a_{max}\Delta t$ are satisfied where ω_{max} and a_{max} are the maximum angular speed and the maximum acceleration of the robot. Then, the velocity and position states of the child nodes are calculated as:

$$v_{a,b}^{c+1} = v_b^{c+1} (\cos(\theta_a^{c+1})\hat{x} + \sin(\theta_a^{c+1})\hat{y}), \quad (11)$$

$$p_{a,b}^{c+1} = p^c + \Delta t v_{a,b}^{c+1} \quad (12)$$

where p^c is the position state of the parent node.

The positions of the neighboring robots at the c^{th} level of the search tree are predicted assuming their velocities remain the same as:

$$p_j^c = p_j^0 + c\Delta t v_j^0 \quad (13)$$

where p_j^0 and v_j^0 are the current position and velocity of the j^{th} neighboring robot.

The cost values of each node are calculated using the flocking heuristic functions. For the i^{th} node at the c^{th} level of the search tree, p_i and v_i are taken as the position and velocity states of the i^{th} node whereas p_j taken as the predicted position of the j^{th} neighboring robot at the c^{th} level of the search tree.

For consistency with SFM, the inter-robot heuristic function is designated in the form of the spring potential energy function as:

$$h_i^r = \sum_{j=1}^M w^r (d_{ij} - d_{eq})^2 \quad (14)$$

where w^r is the inter-robot heuristic coefficient.

Similarly, the obstacle avoidance heuristic function is taken as the potential function of the obstacle avoidance force as:

$$h_i^o = \sum_{k=1}^O w^o \frac{1}{d_{ik}} \quad (15)$$

where w^o is the obstacle avoidance heuristic coefficient.

The migration heuristic is taken as the combination of the speed and direction heuristic functions. The speed heuristic maintains the speed of the robots at migration speed, and the direction heuristic moves robots towards the migration direction. The speed and direction heuristics are calculated as:

$$h_i^s = w^s \left| \|\mathbf{v}_{mig}\| - \|\mathbf{v}_i\| \right|, \quad (16)$$

$$h_i^d = w^d \left(1 - \frac{\mathbf{v}_{mig} \cdot \mathbf{v}_i}{\|\mathbf{v}_{mig}\| \|\mathbf{v}_i\|} \right) \quad (17)$$

where w^s and w^d are the speed and direction heuristic coefficients, respectively.

The sum of the inter-robot, obstacle avoidance, speed, and direction heuristics is calculated as:

$$h_i^u = h_i^r + h_i^o + h_i^s + h_i^d. \quad (18)$$

The heuristic cost of the i^{th} node is obtained as:

$$h_i = h_i^u + h_i^p \quad (19)$$

where h_i^p is the heuristic cost of the parent node of the i^{th} node.

To find a trajectory that meets the flocking objectives, each robot utilizes a beam-search algorithm that expands only the β number of nodes with the lowest heuristic cost at each level of the search tree and prunes the remaining nodes to reduce the required time and memory for the search. The total number of levels in the search tree, which determines the total number of predicted future steps, is called the depth, d . The starting node of the search tree is represented as node 0, whose states are the current position and velocity of the robot, and its heuristic cost is taken as $h_0 = 0$ since its value does not affect the search result. The trajectory of the node with the smallest heuristic cost at the d^{th} level of the search tree is taken as the found trajectory. Each robot repeats the search process within a short time interval, Δt , and takes its next velocity command as the first velocity state of the found trajectory, as illustrated in Fig. 2.

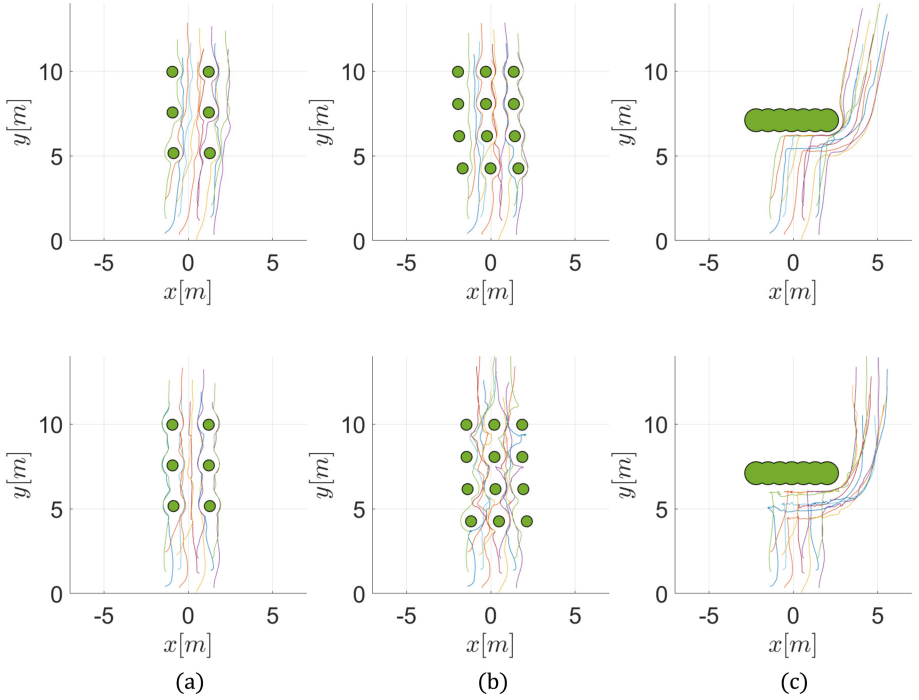


Fig. 3. Simulated trajectories of (top) PFM and (bottom) SFM in (a) the first, (b) second, and (c) third environment. Videos are available at <https://tinyurl.com/Giray22>.

3 Experimental Setup

We prepared three test environments consisting of a $L \times L$ rectangular arena with cylindrical obstacles to compare SFM and PFM in kinematic simulations. The first environment has low obstacle density, the second environment has high obstacle density, and the third environment contains a wall consisting of intertwined obstacles (Fig. 3).

2D Gaussian noise ($\mu = 0, \sigma_n = 0.02$) is added to the positions and velocities of the robots to test the robustness of the models. At the beginning of each test, robots are randomly placed in the environment. The tests are completed when all robots cross the finish line at y_f , and the maximum time allowed to complete a test is limited to t_{max} . Each test is repeated 10 times.

To test the applicability of PFM on real robots, we used a swarm of six Crazyflie 2.1 quadcopters¹ in an indoor flight arena populated with obstacles and communicated with them using the Crazyswarm platform [12]. Positions of the quadcopters are tracked using the Vicon motion capture system, and the velocity commands are computed on a single computer in different threads to mimic the decentralized behavior.

¹ <https://www.bitcraze.io/products/crazyflie-2-1/>.

Table 1. Parameters of PFM and SFM used in simulation and real robot tests.

Parameter	w^r	w^o	w^s	w^d	d_{eq}	$\Delta\theta$	Δv	β	d	A	B	k^r	k^o	k^m	k^l	k^a
Unit	$1/m^2$	m	s/m	–	m	rad	m/s	–	–	–	–	$1/m$	m^2	s/m	m/s	rad/s
Value (Sim.)	50	1.5	2	2	1	0.1	0.02	2	2	2	3	25	1.5	10	0.1	0.1
Value (Real)	25	3	4	2	1	0.15	0.1	2	2	2	1	–	–	–	–	–

Table 2. Common parameters that are used in both simulation and real robot tests.

Parameter	r_a	r_s	v_{min}	v_{max}	ω_{max}	a_{max}	N	M_{max}	O_{max}	Δt	t_{max}	L	y_f	v_{mig}
Unit	m	m	m/s	m/s	rad/s	m/s^2	–	–	–	s	s	m	m	m/s
Value (Sim.)	0.07	2	0.05	2	6	2	12	4	2	0.05	150	14	10.5	$1\hat{y}$
Value (Real)	0.07	2	0.05	2	6	2	6	4	2	0.05	150	8	7.5	$0.5\hat{y}$

The parameters used in simulation and real robot tests are provided in Tables 1 and 2.

The order, speed error, and proximity metrics are introduced based on previous work [15,17] to evaluate the performance of the flocking models. The order metric measures the coherence of the heading directions of the neighboring robots:

$$\Theta_{ord} = \sum_{i=1}^N \sum_{j=1}^M \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{NM \|\mathbf{v}_i\| \|\mathbf{v}_j\|}. \tag{20}$$

Θ_{ord} becomes 1 when the neighboring robots are perfectly aligned, and it becomes -1 in case of complete disorder.

The speed error metric measures the normalized mean difference between the speed of the robots and the desired migration speed:

$$E_{spd} = \sum_{i=1}^N \frac{|\|\mathbf{v}_{mig}\| - \|\mathbf{v}_i\||}{N \|\mathbf{v}_{mig}\|}. \tag{21}$$

E_{spd} becomes 0 when the robots’ speed is equal to the migration speed, whereas its value gets larger when the difference between robots’ speed and migration speed increases.

The proximity metric measures the normalized mean distance between the robots and their neighbors:

$$D_{prox} = \sum_{i=1}^N \sum_{j=1}^M \frac{d_{ij}}{NM d_{eq}}. \tag{22}$$

D_{prox} becomes 1 when the mean inter-robot distance is equal to the equilibrium distance. Its value decreases as the robots get closer to each other and increases as they move away from each other.

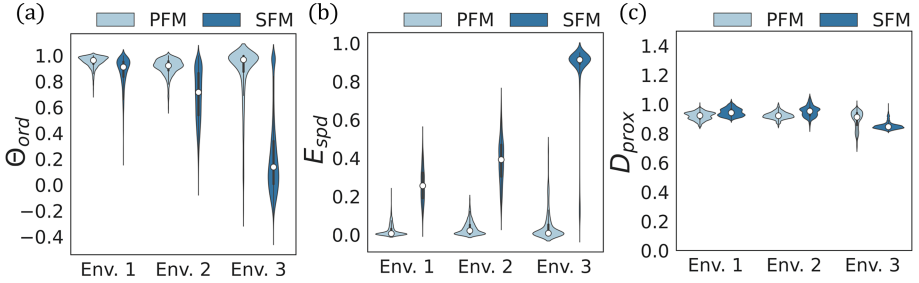


Fig. 4. (a) The order, (b) speed error, and (c) proximity metric values of PFM and SFM for the 10 times repeated tests in three different environments. Colored regions illustrate the data distribution, and white dots within the colored regions represent the median values.

4 Results and Discussion

The metric values of simulation tests for PFM and SFM in three different environments are depicted in Fig. 4 as violin plots. In violin plots, the first 0.5 seconds of each test are excluded in order to eliminate the initial transient period. PFM has order values close to 1 in all environments, whereas the order values of SFM decreased significantly in the second and third environments compared to the first one due to the change in environmental conditions (Fig. 4a). The effect of environmental change is further evident in the speed error metric; PFM has almost zero speed error values, whereas the SFM has significantly large speed error values in all environments (Fig. 4b). In the third environment, SFM only completed half of the tests within the maximum allowed time, t_{max} , and completed the remaining ones 3 to 6 times slower than PFM because of the highly reduced speed of the robots around the obstacle boundary. The performance of PFM and SFM is close in terms of cohesion; both models kept the proximity values close to 1 in all environments (Fig. 4c), and no collisions were observed throughout the tests.

For a more detailed comparison of PFM and SFM, trajectories of robots in three environments with the same initial positions are given in Fig. 3. While both models have smooth trajectories in the first environment (Fig. 3a), the trajectories of SFM are distorted near obstacle boundaries in the second and third environments (Fig. 3b, c). The order, normalized speed, and normalized inter-robot distance plots for both models in the second environment are given in Fig. 5. PFM completed the test faster than SFM by performing more ordered motion (Fig. 5a) and tracking the migration speed much better (Fig. 5b) while maintaining the mean inter-robot distance close to the equilibrium distance (Fig. 5c).

The results of the real robot test are given in Fig. 6. Similar to simulation tests, PFM completed the real robot test with nearly perfect order values (Fig. 6b) by tracking the migration speed with small errors (Fig. 6c) while keeping the mean inter-robot distance around the equilibrium distance (Fig. 6d).

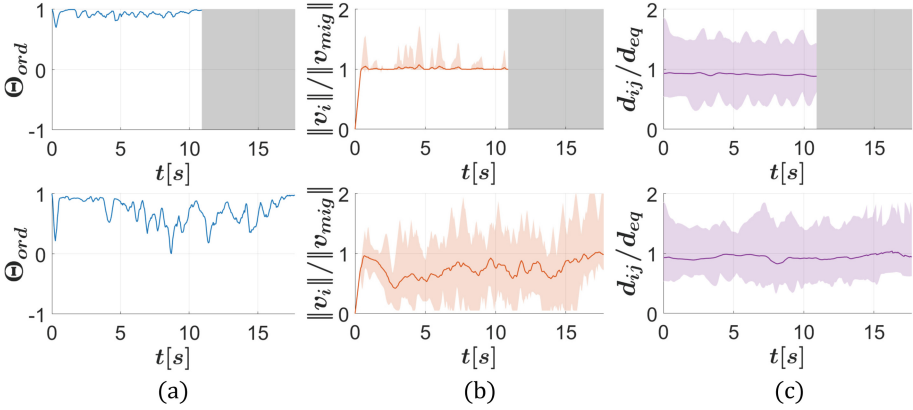


Fig. 5. (a) The order, (b) normalized speed, and (c) normalized inter-robot distance of (top) PFM and (bottom) SFM in the second environment. The plots are grey-scaled from the time PFM completes the task. In (b) and (c), solid lines are the mean values of the normalized speed and normalized inter-robot distance, whereas shades represent the maximum and minimum values. (Color figure online)

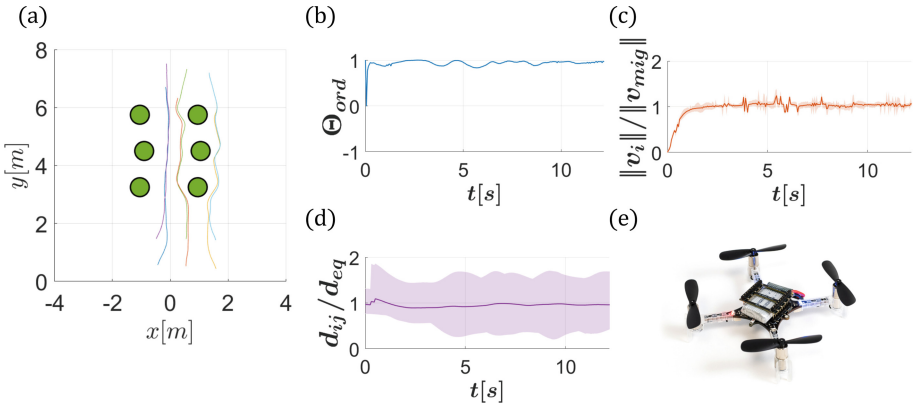


Fig. 6. (a) Trajectories of the quadcopters, (b) the order, (c) normalized speed, (d) normalized inter-robot distance of PFM in a real robot test, and (e) Crazyflie 2.1 quadcopter. In (c) and (d), solid lines are the mean values of the normalized speed and normalized inter-robot distance, whereas shades represent the maximum and minimum values. Video is available at <https://tinyurl.com/Giray22>.

Results of the tests showed that SFM might lead to oscillatory flocking behavior in cluttered environments due to its reactivity, and this behavior has a negative impact on the coherence of the robot swarm. Moreover, it is observed that the short-sightedness of SFM may cause robots to get stuck in obstacles and swarm speed to slow down drastically. On the other hand, it has been shown that PFM can achieve coherent flocking motion at desired migration speed in different environments by allowing the robot swarm to move according to predicted future states.

In this work, it is assumed that robots can only sense the closest point on the boundary of an obstacle. Since planning long-range trajectories without knowing the exact positions and shapes of the obstacles may be misleading, the depth of the search tree is kept small. Furthermore, it is observed that increasing the value of β does not improve the flocking behavior significantly for small depth values. Therefore, small β and d values are used in both simulation and real robot tests, reducing the required time for the trajectory search. Planning short-range trajectories also let robots use the constant velocity assumption for their neighbors, simplifying the prediction process and reducing the computational cost of PFM.

5 Conclusion

In this study, we proposed a novel search-based predictive flocking model (PFM) that only depends on local information of the neighboring robots and the environment. We compared PFM with the virtual force-based standard flocking model (SFM) in different environments based on order, speed error, and proximity metrics. Results showed that the proposed PFM could perform successful flocking behavior despite environmental differences, unlike SFM. We tested the applicability of PFM on real robots with a swarm of six quadcopters in a cluttered flight arena and validated that PFM works successfully with robot swarms as in simulation. Future work will include the dynamic obstacle avoidance and the application of the search-based prediction model for other collective behaviors such as aggregation and foraging.

Acknowledgements. This work was partially supported by the EU H2020-FET RoboRoyale (964492).

References

1. Camazine, S., Deneubourg, J.L., Franks, N.R., Sneyd, J., Theraula, G., Bonabeau, E.: Self-Organization in Biological Systems. Princeton University Press, Princeton (2020)
2. Couzin, I.D.: Synchronization: the key to effective communication in animal collectives. *Trends Cogn. Sci.* **22**(10), 844–846 (2018)
3. Dotson, N.M., Yartsev, M.M.: Nonlocal spatiotemporal representation in the hippocampus of freely flying bats. *Science* **373**(6551), 242–247 (2021)
4. Ferrante, E., Turgut, A.E., Dorigo, M., Huepe, C.: Elasticity-based mechanism for the collective motion of self-propelled particles with springlike interactions: a model system for natural and artificial swarms. *Phys. Rev. Lett.* **111**(26), 268302 (2013)
5. Fine, B.T., Shell, D.A.: Unifying microscopic flocking motion models for virtual, robotic, and biological flock members. *Auton. Robot.* **35**(2), 195–219 (2013)
6. Kong, Z., et al.: Perceptual modalities guiding bat flight in a native habitat. *Sci. Rep.* **6**(1), 1–10 (2016)

7. Liu, Z., Turgut, A.E., Lennox, B., Arvin, F.: Self-organised flocking of robotic swarm in cluttered environments. In: Fox, C., Gao, J., Ghalamzan Esfahani, A., Saaaj, M., Hanheide, M., Parsons, S. (eds.) TAROS 2021. LNCS (LNAI), vol. 13054, pp. 126–135. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-89177-0_13
8. Lyu, Y., Hu, J., Chen, B.M., Zhao, C., Pan, Q.: Multivehicle flocking with collision avoidance via distributed model predictive control. *IEEE Trans. Cybern.* **51**(5), 2651–2662 (2019)
9. Méhes, E., Vicsek, T.: Collective motion of cells: from experiments to models. *Integr. Biol.* **6**(9), 831–854 (2014)
10. Okubo, A.: Dynamical aspects of animal grouping: swarms, schools, flocks, and herds. *Adv. Biophys.* **22**, 1–94 (1986)
11. Parrish, J.K., Viscido, S.V., Grunbaum, D.: Self-organized fish schools: an examination of emergent properties. *Biol. Bull.* **202**(3), 296–305 (2002)
12. Preiss, J.A., Honig, W., Sukhatme, G.S., Ayanian, N.: Crazyswarm: a large nanoquadcopter swarm. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 3299–3304. IEEE (2017)
13. Reynolds, C.W.: Flocks, herds and schools: a distributed behavioral model. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, pp. 25–34 (1987)
14. Soria, E., Schiano, F., Floreano, D.: Distributed predictive drone swarms in cluttered environments. *IEEE Robot. Autom. Lett.* **7**(1), 73–80 (2021)
15. Soria, E., Schiano, F., Floreano, D.: Predictive control of aerial swarms in cluttered environments. *Nat. Mach. Intell.* **3**(6), 545–554 (2021)
16. Spears, W.M., Spears, D.F., Hamann, J.C., Heil, R.: Distributed, physics-based control of swarms of vehicles. *Auton. Robot.* **17**(2), 137–162 (2004)
17. Vászárhelyi, G., Virágh, C., Somorjai, G., Nepusz, T., Eiben, A.E., Vicsek, T.: Optimized flocking of autonomous drones in confined environments. *Sci. Robot.* **3**(20), eaat3536 (2018)
18. Virágh, C., et al.: Flocking algorithm for autonomous flying robots. *Bioinspir. Biomim.* **9**(2), 025012 (2014)