Voronoi-based Geometric Distributed Fleet Control of a Multi-Robot System

Sylvain Bertrand, Ioannis Sarras, Alexandre Eudes, Julien Marzat

Voronoi-based Geometric Distributed Fleet Control of a Multi-Robot System*

Sylvain Bertrand[1], Ioannis Sarras[1], Alexandre Eudes[1] and Julien Marzat[1]

***Abstract*— A new distributed algorithm is presented for waypoint navigation of a multi-robot system. The proposed two-level architecture (reference generator and local controller) exploits Voronoi partitioning and purely geometric considerations to distributively generate references for each robot in order to ensure collision avoidance and convergence of the fleet to the waypoint. Flexibility in the obtained formation pattern is made possible by the algorithm, by not pre-fixing as usually done its geometric form. In addition, the gain tuning is easy and the setting allows to naturally obtain certain formation patterns and adjust the rigidity of the fleet. Moreover the distributed nature of the algorithm also allows robustness to online modification of the number of vehicles (in the fleet or within range of communication), also addressing the 2-robot scenario. Field experiments on ground mobile robots are provided to illustrate the performance of the algorithm.**

SUPPLEMENTARY MATERIAL

A video of the experiments is available at https://tinyurl.com/VoronoiFleet.

I. INTRODUCTION

The last decades have seen an explosion of research activity in the area of cooperative control of multiple unmanned ground/aerial/underwater vehicles (UXVs) [3]–[6]. While initially a large portion of the literature assumed the availability of global team knowledge, leading to the so-called centralized control schemes, the communication and environmental conditions (relative measurements, obstacles, GPS unavailability, limited and dynamic inter-agent communication) during realistic UXV mission scenarios have increased the interest in exploiting only local interactions that lead to distributed control laws. This distributed nature allows for an increased autonomy and robustness and enables building larger fleets of robotic vehicles.

In the literature, there exists a large number of methods that are concerned with distributed control of robotic vehicles, see [3]–[9] and references therein. Existing methods for fleet control can be largely classified in three main categories: 1) Leader-following; 2) Behavioral rules: "nearest neighbors", virtual agents; 3) Virtual structure: implicit (attraction / repulsion) or explicit. Each of these classes of methods has its relative advantages and disadvantages. However, a large number of the existing approaches in distributed fleet control share a common characteristic. The designed algorithms require that the geometric formation of the fleet is (quasi-)explicitly defined through fixed, desired relative positions or distances to be attained [10], [11], or a mix of them [12]. Although this is a natural approach for tackling the problem at hand, it restricts the relative motions of the agents, as well as of the global formation, and does not allow much flexibility and adaptability when dealing with uncertain, dynamic and cluttered environments. For the above reasons such type of formations are usually called rigid [3]. Some exceptions are the works in [1], [2], [8]. The solution of [8] ensures that all vehicles stay inside a region, with a minimum distance between neighbors, whose shape can be assigned by modifying the associated potential function. However, the gain selection for practical deployment of the approach is not easy. Other works have used Voronoi tesselations and their properties to define partitions of the space for motion coordination of robots. More widely used for deployment, allocation and coverage control tasks [5], [9], [13]–[16], Voronoi partitioning has been more rarely exploited for distributed displacement control of fleet of robotic vehicles. The works in [1], [2], [17] are notable exceptions that provide such algorithms. However, the obtained solutions depend on navigation functions to generate a reference point (centroid of the Voronoi cell) to be tracked by each agent and can be difficult to compute and tune in practice or suffer from differentiability issues. In these works, collision avoidance between the vehicles is addressed thanks to the Voronoi partitioning of the space. Moreover, no explicit repulsion behavior between the vehicles is defined that would enable to account for their size for example and guarantee collision avoidance in practice.

This paper builds upon the main principles introduced in these previous works, namely distributed computation by each robot of a reference point to be tracked inside its own cell of a Voronoi partition. The novelty is the way the reference point is computed. Instead of using navigation functions which mainly rely on potentials defined from distances to the objective, the new approach proposed in this paper is based on pure geometric constructions relying on directions to the objective. It enables more intuitive and direct motion to the waypoints, especially when also introducing collision avoidance behaviors between the vehicles. The second contribution of this paper is indeed the possibility to account for both attractive behavior to a waypoint and repulsive behaviors with respect to other vehicles in the construction of the reference point to be tracked. Its geometric nature and the reduced number of parameters make this algorithm easily tunable and interpretable for practical applications. Furthermore, compromise between attraction

*This work was supported by Direction Générale de l'Armement (DGA)

[1]All authors are with Université Paris-Saclay, ONERA, Traitement de l'Information et Systèmes, 91123, Palaiseau, France. {sylvain.bertrand, ioannis.sarras, alexandre.eudes, julien.marzat}@onera.fr

and repulsion can be easily adjusted to allow more flexibility or rigidity in the fleet. Finally, another contribution with respect to previous works is that the proposed algorithm adresses the limit case of "2-robot formations", which can naturally appear in practice in case of system failures or loss of communication.

This paper is organized as follows. Section II presents the problem setting and the general control architecture. A description of the algorithm is provided in Section III while its main properties are discussed in Section IV. The efficiency of the approach is illustrated in Section V by means of field experiments on mobile ground robots. Concluding remarks and perspective for future work are given at the end of the paper.

II. Problem definition and control architecture

A. Problem definition and main notations

The problem under consideration consists in driving a fleet of N robots to a waypoint, and by extension in performing waypoint navigation. The position of the waypoint to be reached by the fleet is denoted by $p^* \in \mathbb{R}^2$.

It is assumed that each vehicle is able to estimate it own position with respect to a common global fixed reference frame, into which p^* is defined, and to broadcast it to all other vehicles within communication range. The position of Robot i will be denoted by $p_i \in \mathbb{R}^2$ and the set of its neighbor robot positions by $\mathcal{N}_i = \{p_j \,|\, j = 1..N, j \neq i, \|p_i - p_j\| \leq r_{com}\}$ where $r_{com} > 0$ is the communication range assumed to be constant. The number of neighbors of Robot i will be referred to as $N_i = Card\,\{\mathcal{N}_i\}$.

B. Control Architecture

The control architecture adopted for the robots in this paper is the one suggested in [1], [2], where each Robot i has a high-level control layer in charge of generating a reference position p_i^* to be tracked by its low-level controller. A fully distributed approach is considered in the sense that this control architecture is embedded on board of each vehicle. The algorithm proposed in this paper concerns the high level reference position generator making use of Voronoi partitions. The low-level controller is described in Section V-A.

III. Algorithm Description

The algorithm can be divided into 5 main steps which are:

- *A: Creation of mirror neighbors*
- *B: Creation of Voronoi partition*
- *C: Computation of attraction point to waypoint*
- *D: Computation of repulsion point(s) for collision avoidance*
- *E: Computation of reference point to be tracked*

Note that steps A and B are similar to the ones introduced in [1], whereas steps C to E are specific to the geometric nature of the new proposed approach.

The main steps of the algorithm, as run by each Robot i of the fleet, are described below by considering the general case $N_i > 1$. Special cases $N_i = 0$ and $N_i = 1$ are addressed in Section III-F.

A. Creation of mirror neighbors

The mechanism proposed in [1] is used to generate mirror neighbors that will ensure that the Voronoi cell of Robot i is bounded and help control the expansion of the fleet. Let us first define the placement operator $\Psi : \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^2$ by

$$\Psi(p_i, p_j, d) = \begin{cases} p_i - d\dfrac{p_j - p_i}{\|p_j - p_i\|} & \text{if} \quad p_i \neq p_j \\ p_i \quad \text{otherwise} \end{cases} \tag{1}$$

If Robot i is not in the convex hull of $\mathcal{N}_i$, then N_i mirror neighbors are defined with positions computed as

$$m_i^j = \Psi(p_i, n_i^j, d_{mir}), n_i^j \in \mathcal{N}_i, j = 1, \ldots, N_i \tag{2}$$

where $n_i^j \in \mathbb{R}^2$ and $d_{mir} > 0$ respectively denote the position of neighbor j and the distance of placement of the mirror neighbor with respect to Robot i (see Figure 1). The set of all mirror neighbors for Robot i will be denoted by $\mathcal{M}_i = \left\{\Psi(p_i, n_i^j, d_{mir}) \,|\, n_i^j \in \mathcal{N}_i\right\}$.

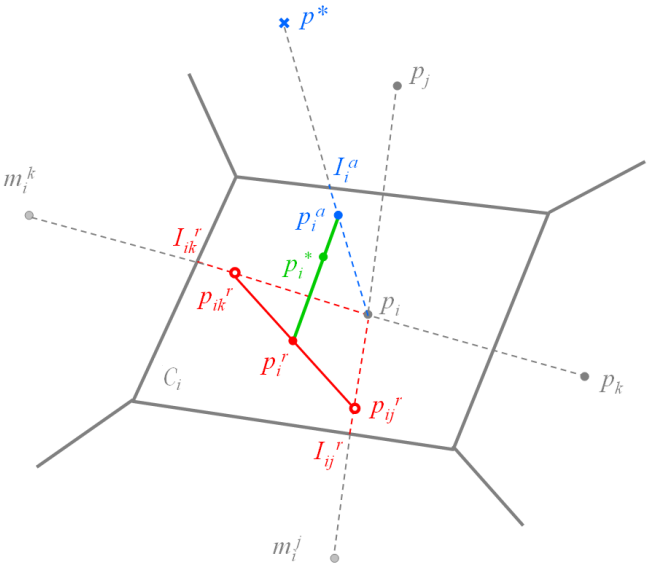


Fig. 1: Main notations used in the algorithm run by Robot i. Illustration for two neighbor Robots j and k leading to repulsion behaviors.

B. Creation of Voronoi partition

Robot i computes its own Voronoi partition using the set of points $\{p_i\} \cup \mathcal{N}_i \cup \mathcal{M}_i$. From this Voronoi partition, only the edges and vertices that correspond to the partition where Robot i belongs are kept. This Voronoi *cell* of Robot i is denoted by $\mathcal{C}_i = (\mathcal{V}_i, \mathcal{E}_i)$, where $\mathcal{V}_i$ and $\mathcal{E}_i$ are the sets of vertices and edges of the cell. This cell defines the space in which the reference position p_i^* to be tracked by Robot i is placed, as defined by the following steps.

C. Computation of attraction point to waypoint

If the waypoint to be reached is located inside the Voronoi cell $\mathcal{C}_i$ of Robot i, then the attraction point is simply defined as $p_i^a = p^*$. If not, p_i^a will be placed inside $\mathcal{C}_i$ along the line of sight between Robot i and the waypoint, as defined by the following procedure.

Let us denote by I_i^a the intersection point of the geometrical segment $\overline{p_i\, p^*}$ with edges of $\mathcal{C}_i$ (see Figure 1) and by $d_{I,i}^a = \|I_i^a - p_i\|$ its distance to Robot i. The attraction point for Robot i is finally defined as

$$p_i^a = \Psi(p_i, I_i^a, -d_i^a) \tag{3}$$

with the distance

$$d_i^a = \min(d_{max}^a, \lambda^a d_{I,i}^a) \tag{4}$$

and where $0 < \lambda^a < 1$ and $d_{max}^a > 0$ are two tuning parameters used to set the position of the attraction point p_i^a on the segment $\overline{p_i\, I_i^a}$ and to limit its distance to Robot i.

D. Computation of repulsion points for collision avoidance

For collision avoidance between robots, repulsion points are defined for Robot i to make it deviate from other robots which are too close. The set of neighbor robots with collision risk with respect to Robot i is defined by

$$\mathcal{N}_i^{col} = \{p_j \in \mathcal{N}_i \mid \|p_i - p_j\| \leq \sigma d_{col}\} \tag{5}$$

where $d_{col} > 0$ ($d_{col} < d_{mir}$) is used to define a distance threshold representing a collision risk and where $\sigma \geq 1$ is used as a smoothing factor to account for some margin in the collision test. Let $N_i^{col} = Card\left\{\mathcal{N}_i^{col}\right\}$ be the number of neighbor robots of Robot i with collision risk.

If $N_i^{col} = 0$, the remaining part of Step D is skipped. If not, for each point $p_j \in \mathcal{N}_i^{col}$ a repulsion point p_{ij}^r is defined as follows. Let us consider the two intersection points of the geometrical line $(p_i p_j)$ with edges of the Voronoi cell $\mathcal{C}_i$ of Robot i. We denote by I_{ij}^r the intersection point such that the dot product $\overrightarrow{p_i p_j}.\overrightarrow{p_i I_{ij}^r}$ is negative, i.e. I_{ij}^r is located on the edge of $\mathcal{C}_i$ opposite to p_j with respect to p_i (see Figure 1). One also denotes $d_{I,ij}^r = \left\|I_{ij}^r - p_i\right\|$. The repulsion point for Robot i to avoid collision with Robot j is then defined by

$$p_{ij}^r = \Psi(p_i, I_{ij}^r, -d_{ij}^r) \tag{6}$$

with the distance

$$d_{ij}^r = \min(d_{max}^r, \lambda^r d_{I,ij}^r). \tag{7}$$

As in Step C, $0 < \lambda^r < 1$ is used to set the position of the repulsion point p_{ij}^r on the segment $\overline{p_i\, I_{ij}^r}$, and $d_{max}^r > 0$ to limit its distance to Robot i.

Following this procedure, one repulsion point is computed by Robot i for each robot with collision risk. A global repulsion point is deduced for Robot i by

$$p_i^r = \frac{1}{N_i^{col}} \sum_{j=1}^{N_i^{col}} p_{ij}^r. \tag{8}$$

Since all the p_{ij}^r are located inside the Voronoi cell $\mathcal{C}_i$, so does the global repulsion point p_i^r. Note that, by relation (8), p_i^r is computed as a mean of the p_{ij}^r. A weighted mean could also be used for example to give more influence to repulsion points corresponding to the closest robots.

E. Computation of reference point to be tracked

Finally, the reference point to be tracked by Robot i is

$$p_i^* = \begin{cases} p_i^a & \text{if} \quad N_i^{col} = 0 \\ \beta p_i^a + (1-\beta) p_i^r & \text{else} \end{cases} \tag{9}$$

where the coefficient $\beta \in [0, 1]$ defines the trade-off between pure attraction to the waypoint ($\beta = 1$) and pure repulsion from the other robots ($\beta = 0$). It can be chosen constant to impose a desired trade-off or as a function of the smallest distance to collision to get a smoother transition between attraction and repulsion behaviors. In the second case, a possible choice is

$$\beta(d_{min}) = \frac{1 - \beta_{min}}{\sigma^2} \left(\frac{d_{min}}{d_{col}}\right)^2 + \beta_{min} \tag{10}$$

with $d_{min} = \min\left\{\|p_i - p_j\| \mid p_j \in \mathcal{N}_i^{col}\right\}$ and $\beta_{min} \in [0, 1]$ a desired lower bound on β.

Remark 1: If $d_{min} = \sigma d_{col}$, relation (10) results in $\beta = 1$ and continuity to pure attraction behavior as imposed by (9) for $N_i^{col} = 0$ (i.e. $d_{min} > \sigma d_{col}$) is ensured.

Remark 2: If in addition one would like to impose a specific value β_{max} to β for $d_{min} = d_{col}$, one can set the smoothing parameter $\sigma = \sqrt{(1 - \beta_{min})/(\beta_{max} - \beta_{min})}$.

As previously mentioned, this description of the algorithm corresponds to the case $N_i > 1$. Special cases corresponding to $N_i = 0$ or $N_i = 1$ are addressed in the next section.

F. Special cases: $N_i = 0$ or $N_i = 1$

During the mission, the number of neighbors of a robot may change, temporarily or definitively, eg. due to loss of communication links, loss of robots, etc. If at a given instant, Robot i has zero or one neighbor, the following additional step is performed before the standard algorithm.

If $N_i = 0$, Robot i has no neighbors. In this case, its reference position to be tracked is simply set to the waypoint, that is $p_i^* = p^*$ and Steps A to E of the algorithm are skipped.

If $N_i = 1$, Robot i has only one neighbor. In this case, the Voronoi cell obtained in Step B will be degenerated. To avoid this problem, if $N_i = 1$, two virtual robots are defined as follows and their positions are added to $\mathcal{N}_i$.

Let $(u_{ij}, u_{ij}^{\perp})$ denote the local reference frame attached to Robot i such that u_{ij} is the unit vector directed from Robot i to its neighbor Robot j, i.e. $u_{ij} = (p_j - p_i)/d_{ij}$ with $d_{ij} = \|p_j - p_i\|$ and $p_j \in \mathcal{N}_i$. The second unit vector $u_{ij}^{\perp}$ completes the orthonormal basis (see Figure 2).

The positions of the two virtual robots are defined by:

$$V_{ij}^1 = p_i + \frac{d_{ij}}{2} u_{ij} + d_{mir} u_{ij}^{\perp} \tag{11}$$

$$V_{ij}^2 = p_i + \frac{d_{ij}}{2} u_{ij} - d_{mir} u_{ij}^{\perp} \tag{12}$$

They are artificially added to the neighborhood of Robot i, that is $\mathcal{N}_i \cup \{V_{ij}^1, V_{ij}^2\} \rightarrow \mathcal{N}_i$, and Steps A to E of the algorithm are executed by Robot i.

Remark 3: By construction, these two virtual robots are located at a distance greater than d_{col} and will not be involved in collision risk.

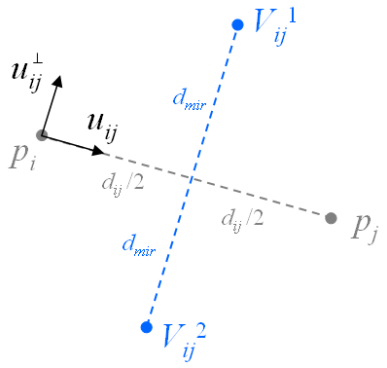


Fig. 2: Positioning of virtual robots in case of $N_i = 1$.

IV. Main properties of the algorithm

This section provides a discussion on the main properties of the algorithm that are of interest for practical applications.

A. Safety regions

Voronoi cells can be viewed as "safety regions" in the sense that if each robot performs a trajectory within its cell to reach its reference point to be tracked, and if the size of the cell is large enough, collisions between the vehicles can be avoided. The repulsion behavior will tend to enlarge the cell, through the parameter d_{col}. Note that since the approach is distributed, each robot will compute its own Voronoi partition and cell. Non-overlapping of the cells can only be guaranteed in case of fully connected communication graphs (i.e. $N_i = N - 1, \forall i$) and if this computation is done in a synchronized way, with all the robots disposing of information corresponding to the same situation of the fleet. In practice, as we do not want to enforce a synchronization mechanism, non-overlapping of the cells can be obtained if the vehicle dynamics are slower than the computation period of the Voronoi partition, as mentioned also in [1], [2]. In addition, parameters λ^a and λ^r can be chosen to define margins with respect to the edges of the Voronoi cell in the placement of p_i^a, p_i^r and hence p_i^* and to keep each robot and its reference to be tracked in a segregated partition of the space. In case of non fully connected communication graphs, other mechanisms must be looked at to provide non-overlapping guarantees for the Voronoi cells.

B. Flexibility of the fleet

Flexibility of the fleet can be adjusted by the parameters d_{mir} and d_{col} which set a compromise between attraction and repulsion between the robots. Choosing $d_{mir} \gg d_{col}$ adds more flexibility to the fleet. A fleet behavior close to a more rigid-formation can be obtained on the contrary for $d_{mir} \approx d_{col}$.

C. Fleet pattern

The pattern obtained for the fleet is not pre-specified but can be influenced by the initial positioning of the robots, making this feature an interesting one for practical applications. For instance a pattern close to a "platooning-like" formation can be obtained for an initial positioning of the robots close to a single line. This can be of interest for motion in narrow corridors. More regular patterns (triangle, square, etc.) can also be easily obtained and maintained during the motion by the same consideration.

D. Robustness to robot failure and communication loss

In practice, the number of robots in the fleet and/or in the neighborhood of each robot may vary during the mission: loss or addition of robots, communication links temporarily/definitively unavailable, etc. Robustness with respect to these issues is ensured in practice by the proposed algorithm, being fully distributed and handling the limit cases with one or two robots.

V. Field Experiments

A. Experimental Setup

Several experiments were carried out to evaluate the performance of the proposed algorithm. Four identical Wifibot Lab V4 ground mobile robots from Nexter Robotics (Figure 4a) were used, each of them being equipped with a stereo-rig sensor with two identical monochrome cameras and an embedded computer (Intel NUC with i7-7567U CPU). Robots were connected together using the 5Ghz band of a WiFi router. The ROS middleware was used and clock synchronization performed by a common NTP reference. The Multimaster node manager system [18] was deployed to facilitate information exchange (positions) by synchronising ROS topics between the robots.
The multi-robot system has been designed to be distributed and autonomous: the same suite of algorithms is executed on each robot, and everything is computed on-board. The global architecture is summarized in Figure 3. Voronoi partitions are computed using the *Boost* polygon library and 2D reference points are generated from the positions of the robots. Low-level guidance is based on a proportional controller for a unicycle model derived from [19] to compute steering inputs from 2D position and reference. The controller has been implemented so as to prefer trajectories close to straight lines in the direction of waypoints by correcting first large orientation errors at a lower speed. This is of a particular importance especially when dealing with nonholonomic vehicles, to obtain trajectories remaining inside Voronoi cells. Localization is computed from the stereo-vision data using the eVO algorithm [20] for visual odometry.

The global position (with respect to a common reference frame) is used to define the list of waypoints needed by the distributed algorithm. The local position is estimated by the visual odometry algorithm embedded in each robot. An additional initialization procedure is used to align the local frame of each individual visual odometry with the same global frame. In our experiments, the global frame was

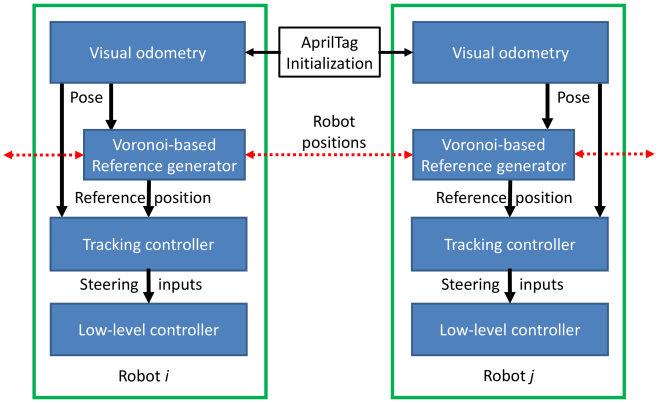
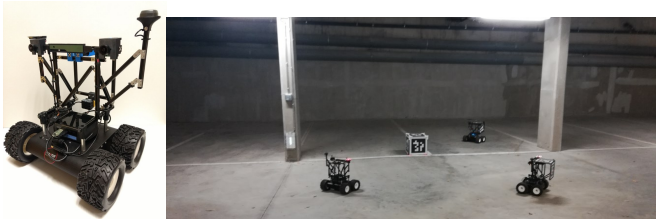


Fig. 3: Architecture of the multi-robot system



(a) Wifibot Lab V4 (b) Example of initialization configuration

Fig. 4: Mobile robots and indoor experimental setup

defined by a cube of AprilTags [21] and each robot computed its initial global position automatically by estimating its relative position with respect to this cube (Figure 4b). This procedure allows to estimate a global position as long as the visual odometry presents a limited drift. In the presented experiments, the final drift (measured by loop closure on the AprilTags cube) was less than 0.5m. This can be considered accurate enough despite possible perturbations on visual odometry due to the presence of moving robots in the field of views, and this did not disturb the demonstration of control performance. For a larger scenario, other visual localization methods should be considered like collaborative localization and distributed SLAM.

B. Scenarios

Field experiments have been realized both in indoor and outdoor environments. During the experiments, all the robots are within communication range from each other, therefore $N_i = N - 1$. Tuning parameters have been set to the values presented in Table I.

Navigation to successive waypoints has been implemented in the following way. All robots dispose of the list of all the waypoints assigned to the fleet. Each robot checks whether the waypoint has been validated (distance criterion, 1 meter in the experiments) by one of the robots of the fleet, then it can switch to the next waypoint of the list. Due to full communication within the fleet, all robots will be heading to the same destination. This is an arbitrary choice as other strategies can be easily implemented depending on the desired behavior of the fleet, e.g. validation of a distance criterion for all the vehicles or addition of a condition on inter-distances between the vehicles of the fleet.

C. Indoor experiments

Indoor experiments have been realized in a parking lot of dimensions $22\ \mathrm{m} \times 15\ \mathrm{m} \times 2.5\ \mathrm{m}$ (see Figure 4b).

In a first scenario, a fleet of four robots was used and two successive waypoints defined with (x, y)-coordinates (11.5, 0.0), (0.75, 0.0) meters. The trajectories of the vehicles are presented in Figure 5. The circle and cross markers respectively correspond to initial and final positions of the robots. Black segments are used to represent the formation obtained between the four robots at different instants of the trajectory chosen for illustration purpose. Flexibility in the obtained pattern can be easily observed through the deformation of the black parallelogram between the four robots. Some "pure" rotation movements can be observed in the trajectories at their beginning and end as well as when changing direction after validation of the first waypoint. They are due to the low-level controller. During "pure" rotation movements, the position of the robot estimated by visual odometry may vary since no compensation is performed between the camera frame and the frame attached to the instantaneous rotation center of the vehicle. Time evolution of the distance to the closest robot is presented in Figure 6 for each robot (plain lines) as well as the repulsion threshold corresponding to d_{col} (dashed line). Note that d_{col} only defines an activation threshold of the repulsion behavior and does not impose a constraint on the inter distances of the robots. This is the reason why some inter distances can be lower than this threshold. Some peaks at low values in the distances to the nearest neighbor correspond to transitions with "pure rotation" motions, as the repulsion behavior imposed by the Voronoi algorithm may not be considered during these short and transient motions. Some chattering can also be observed which is mainly due to the deadzone of the low-level controller and to the fact that positioning mirror robots at a constant distance results in some "waiting behavior" for robots which are "far in advance" from the rest of the group. The constant values at the end of the plot correspond to instants after the final stop of the robots. Figures 5 and 6 illustrate the good performance obtained by the algorithm which enables the robots to perform the mission while navigating as a fleet and without collisions between them.

In the second indoor scenario, three robots were used to follow a square path composed of four waypoints with (x, y)-coordinates (9,0), (9,7.5), (1,7.5), (1,0) meters. Figures 7 and 8 present the obtained trajectories and the time evolution of the distance to the nearest neighbor for each robot. The same analysis as for indoor scenario 1 can be performed.

TABLE I: Values used for the tuning parameters

Indoor scenarios	d_{mir} = 1.5m	d_{col} = 1.0m
Outdoor scenario	d_{mir} = 1.7m	d_{col} = 1.5m
All scenarios	d^a_{max} = 2m	λ^a = 0.75
	d^r_{max} = 2m	λ^r = 0.9
	β_{min} = 0.1	β_{max} = 0.5

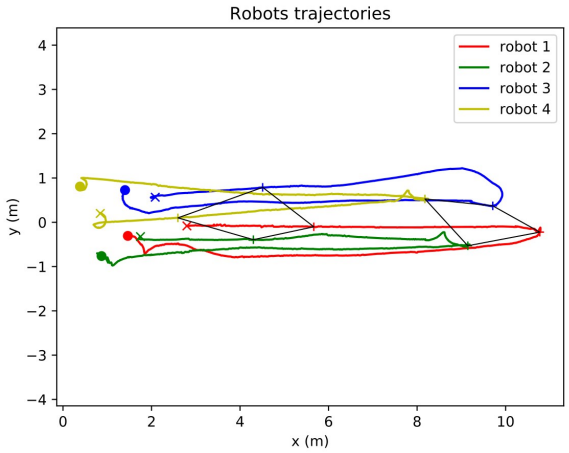


Fig. 5: Trajectories of the robots for indoor scenario 1

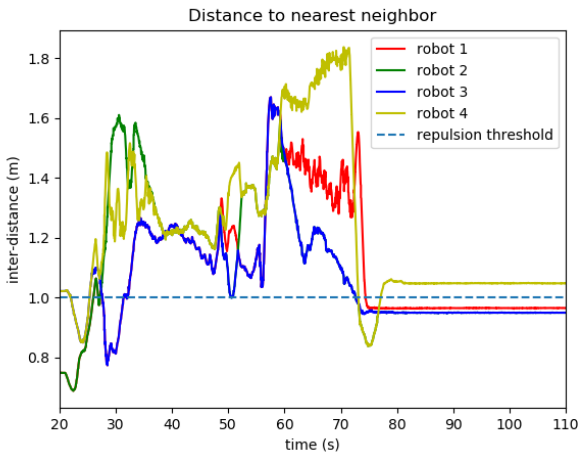


Fig. 6: Distances to nearest neighbor for indoor scenario 1

Good performances of the algorithm are also demonstrated, the three robots being able to navigate as a fleet without collisions and with a flexible pattern close to a triangle.

D. Outdoor experiments

Outdoor experiments have been realized in a semi-urban environment as illustrated in Figure 9. Four robots were used to follow a path on a road between buildings composed of eight waypoints with the following (x, y)-coordinates given in meters: (-2, 8), (7, 3), (38, 2.5), (39, 7), (38, 9), (38, 2.5), (34.5, 1), (0, 2.5). Two robots (Robots 3 and 4) are excluded from the fleet during the mission to illustrate robustness of the proposed approach to the loss of robots. They are then re-integrated into the fleet just before the end of the mission, while heading to the last waypoint. Figure 10 presents the trajectories of the robots. Two bold markers are used for Robot 3 and 4 to indicate their stopping positions (with x-coordinate close to 10 m) during the mission, corresponding to their exclusion period from the fleet. As can been observed, the mission is a success despite the temporary loss of two robots. Navigation for the two remaining robots is successfully ensured by the proposed algorithm, illustrating the special case $N_i = 1$. Online re-

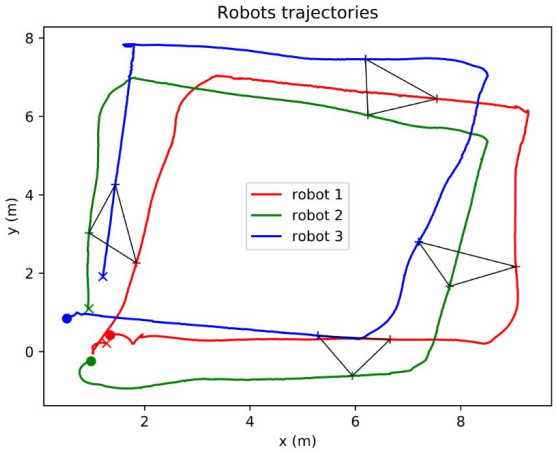


Fig. 7: Trajectories of the robots for indoor scenario 2

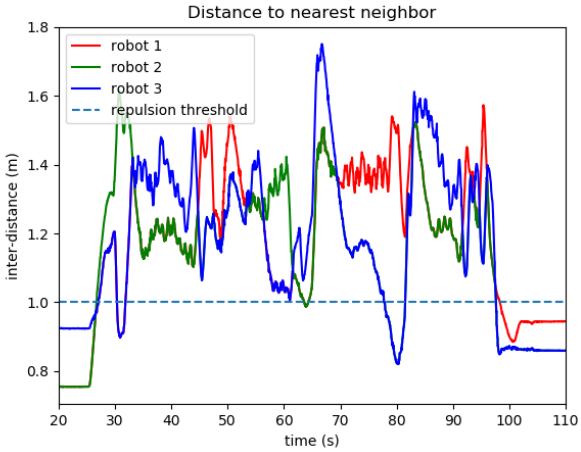


Fig. 8: Distances to nearest neighbor for indoor scenario 2

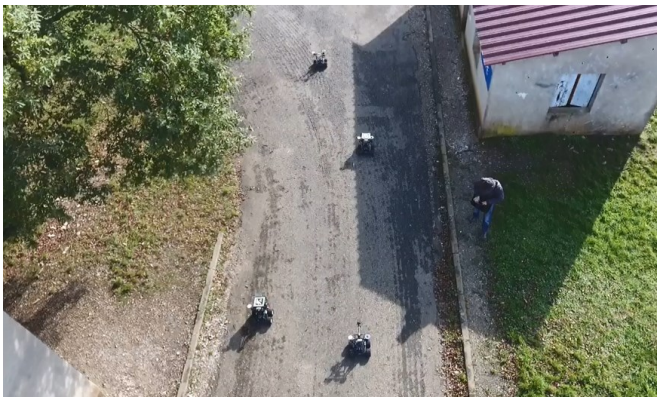


Fig. 9: Experiment in outdoor testing environment

integration of the two robots temporarily excluded is also demonstrated successfully. The exclusion period of the two robots from the fleet can also be observed on Figure 11. Indeed, distances to nearest neighbors remain constant and equal for Robots 3 and 4 (from ∼110s to ∼240s), indicating that they have been stopped. During this exclusion period, distances to the nearest neighbors for Robot 1 and 2 are equal to each other, indicating that they are the only two robots remaining within the fleet.

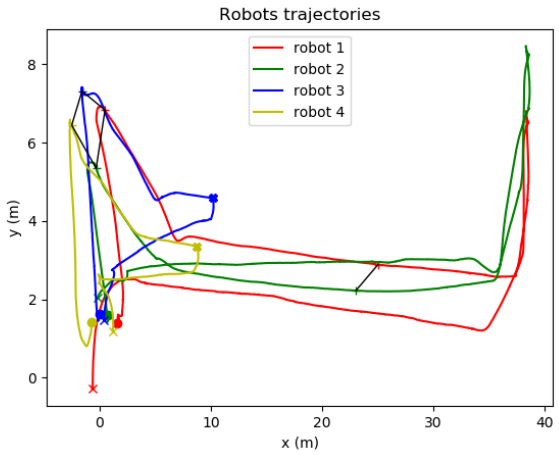


Fig. 10: Trajectories of the robots for outdoor scenario

Fig. 11: Distances to nearest neighbor for outdoor scenario

VI. Conclusions

A new algorithm has been presented for distributed fleet control of a multi-robot system, based on Voronoi partitions. Contrary to similar existing works which rely on potentials, distributed generation of a reference point to be tracked by each robot is done here by a pure geometric approach making the algorithm more easy to tune. Convergence of the fleet to a waypoint and collision avoidance between the vehicles are obtained and flexibility of the resulting pattern for the fleet can also be adjusted. Results from field experiments with mobile ground robots have been presented for indoor and outdoor scenarios, illustrating non-rigid fleet motion capability for waypoint navigation and robustness to the loss of one or several robots.

Future work will consider field experiments with heterogeneous (ground and aerial) robots and extension of the method to static obstacle avoidance.

References

[1] M. Lindhé, P. Ogren, and K. H. Johansson, "Flocking with obstacle avoidance: A new distributed coordination algorithm based on Voronoi partitions", IEEE International Conference on Robotics and Automation, Barcelona, Spain, 2005.

[2] M. Lindhé and K. H. Johansson, "A formation control algorithm using Voronoi regions", Taming Heterogeneity and Complexity of Embedded Systems, Wiley, 2006.

[3] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks,* Princeton University Press, 2010.

[4] W. Ren and R. Beard, *Distributed Consensus in Multi-vehicle Cooperative Control: Theory and Applications,* Springer, 2010.

[5] F. Bullo, J. Cortes and Sonia Martinez, *Distributed Control of Robotic Networks,* Applied Mathematics Series, Princeton University Press, 2009.

[6] R. M. Murray, "Recent research in cooperative control of multivehicle systems," Journal of Dynamic Systems, Measurement, and Control, vol.129, no.5, pp.571-583, 2007.

[7] W. Ren, R. W. Beard, and E. M. Atkins, "Information consensus in multivehicle cooperative control," IEEE Control Systems Magazine, vol.2, pp.71-82, 2007.

[8] C. C. Cheah, S. P. Hou, and J. J. E. Slotine, "Region-based shape control for a swarm of robots," Automatica, vol.45, pp.2406-2411, 2009.

[9] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks", IEEE Transactions on Robotics and Automation, vol.20, no.2, pp.243–255, 2004.

[10] G. Lafferriere, A. Williams, J. Caughman, and J. J. P. Veerman, "Decentralized control of vehicle formations", Systems & Control Letters, vol.54, pp.899-910, 2005.

[11] K-K. Oh, M-C. Park and H-S. Ahn, "A survey of multi-agent formation control", Automatica, vol.53, pp.424-440, 2015.

[12] K. Fathian, D. I. Rachinskii, M. W. Spong, T. H. Summers and N. R. Gans, "Distributed Formation Control via Mixed Barycentric Coordinate and Distance-Based Approach", American Control Conference, Philadelphia, PA, USA, 2019.

[13] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, " Coverage control for mobile sensing networks", IEEE International Conference on Robotics and Automation, Washington DC, USA, 2002.

[14] K. R. Guruprasad, and P. Dasgupta, "Voronoi Partitioning for Multi-Robot Systems with Limited Range Sensors", IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, Portugal, 2012.

[15] J. Hatleskog, S. Olaru, and M. Hovd, "Voronoi-based deployment of multi-agent systems", IEEE Conference on Decision and Control, Miami, USA, pp. 5403-5408, 2018.

[16] T. Chevet, C. Stoica Maniu, C. Vlad, and Y. Zhang, "Guaranteed Voronoi-based Deployment for Multi-Agent Systems under Uncertain Measurements", 18th European Control Conference, Naples, Italy, 2019.

[17] Q. Jiang, "An Improved Algorithm for Coordination Control of Multi-Agent System Based on r-limited Voronoi Partitions", IEEE International Conference on Automation Science and Engineering, Shanghai, China, 2006.

[18] A. Tiderko, F. Hoeller, and T. Röhling, "The ROS multimaster extension for simplified deployment of multi-robot systems", Robot Operating System (ROS), Springer, pp.629–650, Cham, 2016.

[19] M. Bak, N. K. Poulsen, and O. Ravn, "Path following mobile robot in the presence of velocity constraints", Technical Report, Informatics and Mathematical Modeling, Technical University of Denmark, 2001.

[20] M. Sanfourche, V. Vittori, and G. Le Besnerais, "eVO: A realtime embedded stereo odometry for MAV applications", IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 2013.

[21] E. Olson, "AprilTag: A robust and flexible visual fiducial system", IEEE International Conference on Robotics and Automation, Shanghai, China, 2011.