Field coverage for weed mapping: toward experiments with a UAV swarm

Dario Albani[1 2], Tiziano Manoni[2], Arikhan Arik[2], Daniele Nardi[2], and Vito Trianni[1]

[1] ISTC, National Research Council, Italy
{dario.albani,vito.trianni}@istc.cnr.it
[2] DIAG, Sapienza University of Rome, Italy
{surname}@diag.uniroma1.it

Abstract. Precision agriculture represents a very promising domain for swarm robotics, as it deals with expansive fields and tasks that can be parallelised and executed with a collaborative approach. Weed monitoring and mapping is one such problem, and solutions have been proposed that exploit swarms of unmanned aerial vehicles (UAVs). With this paper, we move one step forward towards the deployment of UAV swarms in the field. We present the implementation of a collective behaviour for weed monitoring and mapping, which takes into account all the processes to be run onboard, including machine vision and collision avoidance. We present simulation results to evaluate the efficiency of the proposed system once that such processes are considered, and we also run hardware-in-the-loop simulations which provide a precise profiling of all the system components, a necessary step before final deployment in the field.

1 Introduction

Swarm robotics can have a large impact in many application domains, especially when the tasks to be accomplished are distributed widely in space, and when there is room for collaboration among robots [5]. In such conditions, not only it is possible to profit of the parallel execution of tasks by multiple robots, but the execution of a single task is made more efficient thanks to collaboration, opening to the exploitation of solutions designed for multi-robot task allocation problems [7,11]. An application domain presenting expansive fields and needs for collaboration is certainly precision agriculture, whereby robotics technologies promise a remarkable impact [9]. Current practice is however far from exploiting the full potential of autonomous robots and multi-robot collaboration. Off-the-shelf remote sensing technologies with unmanned aerial vehicles (UAVs), for instance, rarely go beyond passive data collection with predefined mission plans [13,8]. Therefore, even in field mapping applications, much improvement is expected by the introduction of adaptive mission planning and collaboration among multiple UAVs [14,2].

Previous work proposed swarm robotics approaches for UAVs engaging in a weed monitoring and mapping problem [2,1]. In this problem, weeds must be recognised from crops in order to create a precise weed-density map to be exploited for weed control operations (e.g., for variable-rate herbicide applications). Swarm robotics solutions are useful to parallelise the monitoring and mapping operations, and to have robust controllers scalable to different field sizes [2]. Additionally, the weed distribution across an expansive field is often heterogeneous, with weed patches infesting certain areas while other areas remain devoid of weed. In such conditions, it is useful to adaptively monitor only those areas where weeds are present—and to collaborate for that purpose—resorting to non-uniform coverage and mapping strategies [15,1]. These studies just provide a proof of concept in abstract simulations, and several relevant features that pertain to the real-world implementation have been overlooked. For instance, collision avoidance among UAVs was not taken into account, and the onboard weed detection through machine vision was streamlined by a simple mathematical model of the weed detection error [2].

In this paper, we move toward real-world testing with the following contributions. (i) We provide the full implementation of a weed coverage and mapping system for a UAV swarm, with a design tailored to the Avular Curiosity platform [3]. For experimental purposes, we set up an experimental indoor arena whit a green carpet and pink golf balls to represent weeds, following the rules of the 2017 Field Robot Event [6], a renowned competition for autonomous farming robots. This choice drastically simplifies the vision routines, but ensures that all the components of the control loop are taken into account. (ii) We provide an improved implementation of the stochastic coverage and mapping presented in [2], including collision avoidance among UAVs and onboard vision. We test the performance of the deployed algorithm in simulation for a wide range of parameterisations, identifying the most suitable parameters sets for efficient coverage and mapping. (iii) We integrate the hardware platform into the simulation environment and we perform hardware-in-the-loop (HIL) simulations to profile the different components of the onboard system. We propose HIL simulations as a convenient way of testing swarm robotics controllers before actual deployment, especially when physical interactions are not needed. (iv) We provide proof-of concept videos of the real flying system.

The paper is structured as follows. Section 2 describes the design of the different components implementing the stochastic monitoring and mapping strategy. Section 3 discusses the results of simulations for field coverage, weed mapping and HIL simulations. Section 4 concludes the paper discussing the results obtained and the future steps necessary for field deployment of UAV swarms.

2 Experimental setup

Weed monitoring and mapping requires UAVs to fully cover a field, detect the presence of weeds and map their exact position. We considered a mockup version of the real problem by simplifying the vision requirements, so that we can focus on the development and testing of the swarm-level strategy. In this mockup version, weeds are represented by pink golf balls placed on a green carpet, making their detection a relatively simple task. Here, our main objective is to develop and test a simple and reliable coverage and mapping algorithm in realistic conditions, and to flexibly switch between simulation and execution on the UAV platform, also mixing the two with HIL simulations.

2.1 Hardware

The Avular Curiosity platform is a small square quad-rotor of about 40 cm side and 1 kg weight (see Figure 1) [3]. The platform provides dedicated hardware for low-level flight control and high-level mission execution. The low-level control unit consists of a PX4 micro-controller that acts as a bridge between the high level abstraction and another PX4 that implements the autopilot. The low-level unit integrates a variety of sensors for motion control, such as a real-time kinematic global navigation satellite system unit (RTK-GNSS), a dual 9-DOF IMU, a barometer and a laser range-finder for altitude control, and the Ultra-Wide Band (UWB) system for indoor self-localisation. These sensors support precise absolute localisation both indoor and outdoor, hence allowing waypoint navigation and simple trajectory generation.

The high-level control unit consists of a Raspberry Pi3 (RPi), a small and affordable programmable device characterised by low energy consumption. The role of the RPi is to support the overall mission execution strategy (see Section 2.2), which includes both onboard vision and communication with other UAVs within the swarm. The latter is achieved thanks to the Digi ZigBee, a specific module coming from the Xbee family of communication devices.

2.2 Software

The proposed implementation of the high-level mission execution strategy exploits the Robot Operating System (ROS) structure, in particular ROS Kinetic running on the RPi platform, and is illustrated in Figure 1. The overall framework improves over the stochastic coverage and mapping strategies presented in [2] by refining the reinforced random walk model including collision avoidance (see also Section 2.3), by removing inefficient computations and by adding onboard vision (Section 2.4).

More in detail, the proposed implementation offers three main ROS packages: (i) `Core`, (ii) `Perception` and (iii) `Communication`, each one responsible of a specific task. The `Core` package is responsible for defining the navigation of the UAV between different areas of the field, assuming knowledge of the absolute position of the drone, either from GNSS outdoor or from the UWB indoor. The `Perception` package is responsible for image acquisition and processing. It presents two different nodes, one for communication with an external camera (i.e., the Raspicam connected to the RPi) to be used for real field tests, and another one used in the HIL simulations that loads real images from the local dataset and processes them on-line. The last package is `Communication`, which enables swarm operations and allows drones to perform cooperative field mapping as well as communication between high-level and low-level control units.

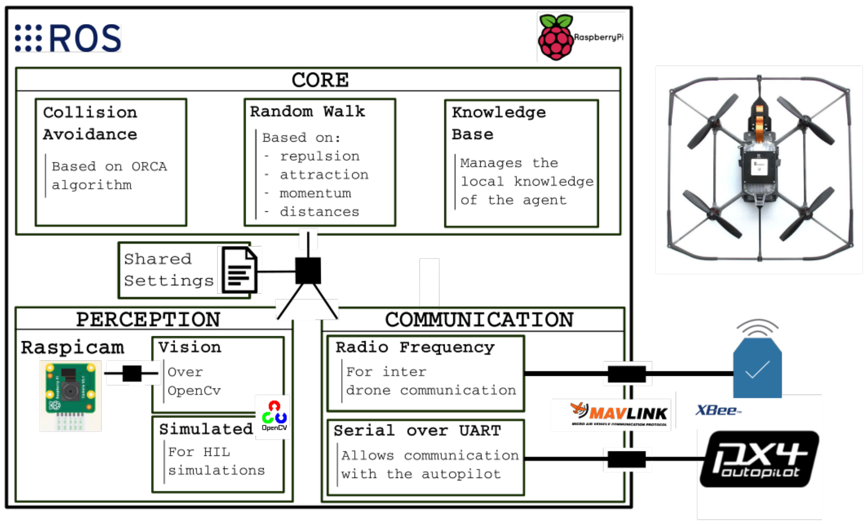

Fig. 1. Proposed implementation of the mission execution strategy for the Avular Curiosity platform [3], which is shown in the top right side. The development is made on the Raspberry Pi3 platform using ROS running over a custom version of Ubuntu Mate 16.04. There are three main ROS packages, namely (i) `Core` (ii) `Perception`, and (iii) `Communication`. Each package allows to run any of different nodes it embeds.

The node responsible for the latter is the `Serial` node, which enables direct communication with the low level autopilot through the RPi serial port. Communication with other drones is made possible by the `Radio Frequency` node that forwards and processes information to and from the ZigBee external module, implementing also the selected information-aware broadcasting protocol [2]. Information sharing is made possible by the Micro Air Vehicle Link (MAVLink) communication protocol, responsible of encoding and decoding both default and custom messages.

2.3 Stochastic coverage and mapping

The basic navigation strategy implemented by the `Core` package derives from previous work [2], and improves it in several ways. As in previous work, we consider a field divided in square cells of 1 m side. The UAVs have access to their global position (either through GNSS or UWB) and can communicate reliably with each other.[3] We assume a maximum density for the point of interests in each cell. For instance, in the indoor tests here presented we assume that each cell contains up to 12 pink golf balls. To accomplish the coverage task, each cell

[3]We ignore here communication limitations, which have been suitably accounted for through information re-broadcasting protocols [2,1].

needs to be visited at least once and by at least one UAV while, to accomplish the mapping task the number of balls within the cell should be reliably detected. Similarly to [2], the `Knowledge Base` node contains the local knowledge of the current coverage and mapping activities, which also integrates information received from other UAVs through the `Communication` nodes. This knowledge is completely distributed, meaning that each UAV in the swarm posses its own representation of the world. The local knowledge is constantly updated by means of information observed locally or received from other UAVs, and determines the navigation strategy implemented in the `Random Walk` node.

More in details, at every decision step, a UAV selects the next cell to visit randomly choosing one cell from a valid set $\mathcal{V}$. A cell is considered valid if it has not been covered or mapped before by any other UAV, and if it is not occupied/targeted by other UAVs within the swarm. Cells are added to $\mathcal{V}$ in order of increasing distance from the UAV (see Figure 2a), until a maximum number V is reached (in this study, $V = 1$). At this point, the set $\mathcal{V}$ is completed including all valid cells within the maximum distance reached. In this way, the distance to be covered by a UAV to reach the next cell is always minimised. Here, we simplify the construction of $\mathcal{V}$ by looking at the whole plane, while in [2] priority was given to the semi-plane in the forward direction of motion. In this way, we save precious computational time for implementing the strategy on the RPi.

We consider here a reinforced random walk [17], in which a directional bias is given by three components, as shown in Figure 2a: (i) the individual momentum $\boldsymbol{m}_h$, a unit vector in the direction of motion of the UAV h resulting in a correlated random walk; (ii) the repulsion vector $\boldsymbol{r}_h$ from all other UAVs in the swarm; (iii) the attraction vector $\boldsymbol{a}_h$ towards cells previously marked as attractive by other UAVs with virtual beacons $b \in \mathcal{B}$. Attraction and repulsion vectors are computed as follows:

$$\boldsymbol{r}_h = \sum_{u \neq h} S(\boldsymbol{x}_h - \boldsymbol{x}_u, \sigma_a), \ \boldsymbol{a}_h = \sum_{b \in \mathcal{B}} S(\boldsymbol{x}_b - \boldsymbol{x}_h, \sigma_b), \ S(\boldsymbol{v}, \sigma) = 2e^{\mathrm{i}\angle \boldsymbol{v}} e^{-\frac{|\boldsymbol{v}|}{2\sigma^2}}, \quad (1)$$

where $\boldsymbol{x}_i$ represents the position of agent/beacon i, and $S(\boldsymbol{v}, \sigma)$ returns a vector in the direction of $\boldsymbol{v}$ with a Gaussian length with spread σ. With respect to [2], we simplify the way in which beacons are placed and removed. Here, an agent places a beacon on a cell if something has been detected, and only if no other beacon is present. Beacons are removed from cells that are reliably mapped.

The selection of the next cell to visit is performed randomly using the vector $\boldsymbol{v}_h = \boldsymbol{m}_h + \boldsymbol{r}_h + \boldsymbol{a}_h$ as a bias. Each cell $c \in \mathcal{V}$ is assigned a probability $P_c = u_c / \sum_{i \in \mathcal{V}} u_i$ of being selected, where the utility u_c is computed according to the angular difference $\theta_c = \angle(\boldsymbol{x}_c - \boldsymbol{v}_h)$ as follows:

$$u_c = C(\theta_c, 1 - e^{\frac{|\boldsymbol{v}_h|}{2}}), \quad C(\theta, p) = \frac{1}{2\pi} \frac{1 - p^2}{1 + p^2 - 2p \cos\theta} \quad (2)$$

where $C(\cdot)$ is the wrapped Cauchy density function with persistence p. Differently from [2], we use the length of the bias vector $\boldsymbol{v}_h$—filtered by a smooth

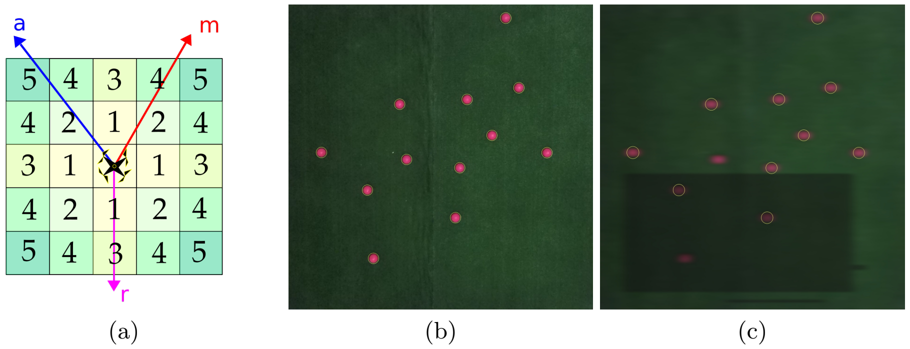

Fig. 2. (a) The RRW gives priority to neighbouring cells—numbered for increasing distance from the UAV placed in the center—and to cells in the direction of the resultant vector from the individual momentum $\boldsymbol{m}$, the repulsion from other drones $\boldsymbol{r}$ and the attraction to beacons $\boldsymbol{a}$. (b-c) Weeds are represented by pink golf balls over a green carpet. Circles around the balls represent the output of the vision module. Artificial disturbances are included in the left image to obtain realistic output from the visual processing. Disturbances represent illumination variance, shadows and motion blur.

exponential ceiling function—to modulate the persistence of the random motion, so that a small module results in lower directional bias in the cell choice.

One of the biggest limitations of [2] is the absence of collision avoidance, which is required for real world deployment. Here, the navigation strategy includes the `Collision Avoidance` ROS node, which is responsible to generate sage and accurate trajectories. To this end, information about the position of other UAVs (as broadcasted through the `Communication` node) is made available by the `Knowledge Base` node. Based on such information, we implement the Optimal Reciprocal Collision Avoidance (ORCA) method [18,4]. ORCA is based on the knowledge of the position and velocities of the agents that may interfere with the planned trajectory. To avoid collisions, ORCA assigns part of the responsibility of implementing a correct manoeuvre to all agents involved. The method calculates all possible collisions within a time interval τ (in this work, $\tau = 1\,\mathrm{s}$). To this end, an agent is approximated by a disk with a safety radius r_o, which indicates the area around the agent that should not be violated. Then, all possible collision-free velocities are computed and the one that remains closer to the original velocity is selected. In this work, we integrate ORCA with the reinforced random walk strategy by letting the former change the trajectory to reach the desired destination. In cluttered conditions, collision avoidance may be very time-consuming. To avoid deadlocks, we let UAVs abandon their current target in favor of a new one if the flying time exceeds twice the expected value.

2.4 Onboard vision module

The goal of the `Vision` package is to implement the routines necessary to detect and count pink golf balls over a green carpet. Field experiments will add

ROS nodes for the real weed recognition systems based on previous and ongoing studies [12,10]. To detect pink golf balls, a simple blob-detection algorithm has been implemented using OpenCV. We used a trivial procedure consisting of circularity filtering and converted the image color space from BGR to HSV in order to make the extraction of the pink golf balls more accurate. To tune and validate the blob detection algorithm, as well as for HIL simulation purposes, we generated a large dataset by placing pink golf balls randomly over a synthetic grass carpet within a $1\,\mathrm{m}^2$ area (see Figure 2b). We varied the number of golf balls from 0 to 12, and for each size we collected 20 images. Finally, the dataset has been extended by rotating and flipping each image generating further unique configurations of the ball positions.

After tuning, the developed blob detection algorithm perfectly detects and counts the balls within the dataset. However, images taken from the UAVs while flying will never be as sharp and bright as the ones in the collected dataset. To evaluate the effectiveness of the devised mapping strategy under realistic working conditions, we artificially add disturbances to the images, to represent (i) changes in global illumination, (ii) shadows locally affecting portions of the image, and (iii) motion blur due to sudden rapid movements of the UAV. We simulate these disturbances by reducing the contrast of the image globally as well as within randomly generated blobs in the image, and then we apply a convolution with an averaging Gaussian kernel with a random direction (see Figure 2c). To evaluate the performance of the `Vision`, we traversed all of the dataset 100 times and for each image we added artificial disturbances. We tuned the artificial disturbances to obtain an overall 22% error (mean: 0.2175 standard deviation: 0.018) so to ensure realistic and competitive tests. Indeed, given such error, it is not possible for the UAVs to know whether a cell is reliably mapped or not. In this paper we approximated this decision by marking a cell as reliably mapped within the `Knowledge Base` when the number of detected balls does not differ from a previous passage. In this way, at least two passages over the same cell are necessary to mark it as reliably mapped. The mapped state is then broadcasted to all other UAVs in the swarm via the `Communication` package.

3 Results

As mentioned above, the software we developed can be seamlessly integrated with simulations or executed by the UAV platform [3]. We present here simulation results to study both simple coverage by UAV swarms (Section 3.1) and weed mapping (Section 3.2). We also present results of HIL simulations, whereby the UAV platform is integrated within the simulator and used to execute—without flying—the coverage and mapping algorithm (see Section 3.3).

3.1 Field coverage

The simple coverage problem requires that UAVs inspect every portion of the field by visiting (and inspecting) each cell at least once. Ideally, every cell

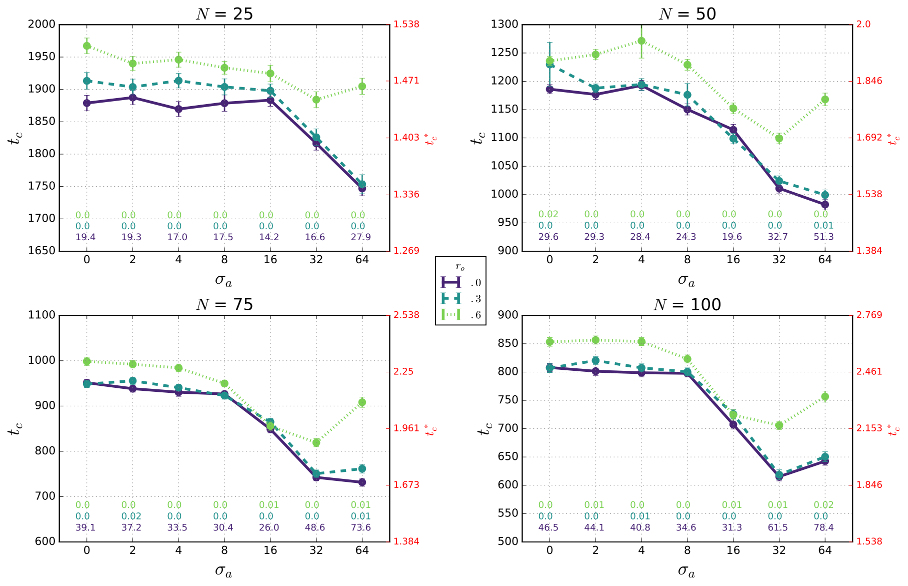

Fig. 3. Coverage time (absolute t_c and relative t_c^*) for varying swarm size N. Each panel shows the average results over 150 runs varying the Gaussian spread σ_a used to compute the repulsion vector $\boldsymbol{r}$ and the safety radius r_o that parametrises the ORCA algorithm. Errorbars represent the standard error. The number of collisions detected in each conditions is printed in the bottom of each panel, and color-coded according to the corresponding value of r_o.

is visited only once by one UAV and the coverage time t_c is minimised. In this study, we consider a field of $M = 50 \times 50 = 2500$ cells, and swarm size $N \in \{25, 50, 75, 100\}$. Following the approach taken in [2], we consider the absolute coverage time t_c as well as the time t_c^* relative to a lower bound computed as $t_1 M/N$, where t_1 is the time taken by a UAV to move between two adjacent cells. Focusing on coverage only, we disable the `Vision` node, and cells are marked as covered as soon as visited by a UAV, so that they are not visited a second time. Also, virtual beacons are not used for coverage. We instead focus on the interaction between the random walk, repulsion from other UAVs (as parametrised by the Gaussian spread $\sigma_a \in [0, 64]$), and collision avoidance (as parametrised by the ORCA radius $r_o = \{0, 0.3, 0.6\}$, considering collision avoidance disabled when $r_o = 0$). We also count the number of potential collision events detected anytime two UAVs get closer than 0.3 m. Note that these events do not affect the UAV motion in simulation (i.e., UAVs continue their mission even after a potential collision has been detected). In this way, we can evaluate the effects of collision avoidance on the overall performance.

The results of simulations are presented in Figure 3. Coverage time tends to decrease with larger repulsion among UAVs, especially for $\sigma_a > 8$ until it hits a lower bound and starts increasing again (e.g. $\sigma_a > 64$). This means that a sufficiently high repulsion is necessary to ensure that UAVs remain separated from each other to cover different areas. As the size of the swarm increases, the coverage time decreases in absolute terms, but not when compared with the lower bound, corresponding to a sub-linear increase of efficiency (i.e. a linear increment of agents does not correspond to a linear increase of the efficiency). Moreover, as the size of the swarm increases we observe that high repulsion is detrimental. Indeed, high repulsion values associated to swarms of a considerable size (i.e. 75 and 100) do not aid the overall performance, pushing UAVs to the boundaries of the environment and causing inner cells to have low probability of being visited. Still, the performance obtained by the reinforced random walk strategy is remarkable, if we consider that the lower bound represents an ideal performance that is hardly achievable in practice.

In general collision avoidance results in slower coverage as UAVs take time to avoid each other. However, when $r_o = 0.3$, the difference with the ideal no-collision case is negligible. A larger safety radius ($r_o = 0.6$) has a larger impact because resolving collisions is generally more complex, especially when $\sigma_a > 16$. In these conditions, the strong repulsion makes UAVs move in a more directed way, hence interfering more with each other. Despite the slightly longer coverage time, ORCA successfully manages to avoid collisions among UAVs. The number of detected collision events is very high when ORCA is disabled, but practically all collisions are avoided when avoidance is enforced.

3.2 Weed mapping

Once studied the effect of collision avoidance on the field coverage efficiency, we move to study the ability to precisely map the field for the presence of weeds, here represented by pink golf balls. We consider again fields of $M = 50 \times 50$ cells containing 5 patches of weeds, each represented by a square of 6×6 cells where the distribution of balls follows a 2D Gaussian, having (higher density in the center and lower density in the periphery. Each weed-infested cell has an associated image from the dataset, over which artificial disturbances can be added at runtime, as described in Section 2.4. We study both the conditions with and without such disturbances, to evaluate their effect on the mapping strategy. As mentioned in Section 2.4, a cell is marked as reliably mapped when the number of balls detected is equal to the number previously stored in the `Knowledge Base`. Hence, when no disturbance is added, two visits per cell are required even when no weed is present. More visits may be required in case of perception errors. UAVs perform a reinforced random walk under the influence of both repulsion from other agents ($\sigma_a \in \{0, 2, 4, 8, 16, 32\}$) and attraction to virtual beacons ($\sigma_b \in \{0, 2, 4, 8, 16, 32\}$). We fix in this case the value of the ORCA radius $r_o = 0.3$, which provides safe conditions for collision avoidance with negligible performance cost, as discussed in Section 3.1. To evaluate the performance of the system, we consider here the coverage time t_c and the mapping time t_m. The latter corresponds to the time in which all weed-infested cells

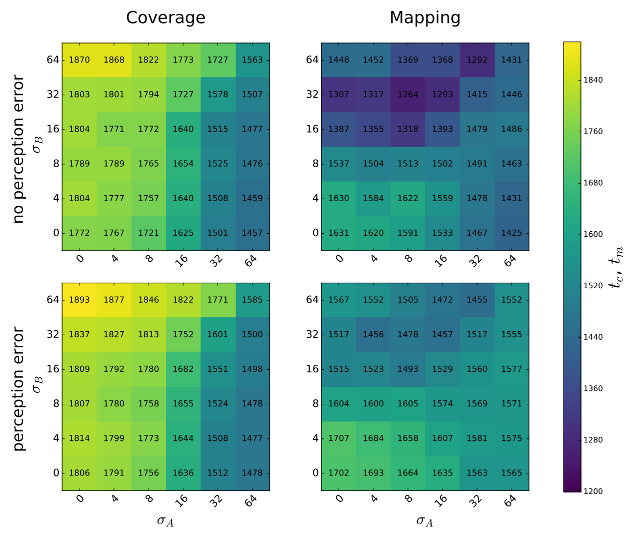

Fig. 4. Comparison of the coverage and mapping time for $N = 50$ robots. Each cell in the heatmap represents the average of 150 runs.

are marked as reliably mapped. Additionally, we look at the mapping accuracy by recording the average detection error with respect to ground truth.

Figure 4 shows the average coverage and mapping time for $N = 50$ UAVs.[4] When no error in perception is considered (top-left and top-right panels), coverage time decreases with increasing repulsion among the UAVs—similarly to what observed before—but increases with higher attraction towards beacons. Indeed, a high value of σ_b makes all UAVs move towards the weed patches first, leaving other areas of the field unattended and resulting in a overall higher coverage time, as the field gets fully covered only when all cells receive at least one visit. Note that coverage times are slightly higher in this case than what is shown in Figure 3, because the mapping task requires multiple passages over the same cell, hence slowing down coverage. On the other hand, the mapping time t_m decreases with high attraction towards beacons, as more UAVs are dedicated to mapping only those areas that require attention. Conversely, mapping is less efficient when repulsion among UAVs is too strong, to the point that no substantial difference with coverage is visible when $\sigma_a = 64$. The smallest values of t_m occur for medium values of attraction and large values of repulsion ($\sigma_a = 8$ and $\sigma_b = 32$ in Figure 4). These values slightly vary with the group size N, as shown

[4]Data for different group sizes are available in the appendix at the end of the manuscript.

in the appendix, but generally indicate that there can be positive interactions between repulsion among UAVs and attraction towards beacons.

When some error in perception is introduced with artificial disturbances on the images (bottom panels in Figure 4), the coverage and mapping time in general increases due to the need to frequently revisit those cells where some perception error occurred. Such negative effects are negligible for what concerns the coverage time, because the entire field must be covered in any case. The mapping time gets instead much worse, due to the need to visit multiple times just those cells where weed is present. For $\sigma_a = 64$, mapping terminates even after coverage, indicating that repulsion among agents is too strong to make UAVs focus on the weed infested areas en masse.

The accuracy of mapping in presence of perception error increases thanks to the multiple visits performed to weed-infested areas, which allow to increase the probability of detecting the correct number of balls, as UAVs visiting a cell at different times get a different perception error. The detection error decreases below 5% (mean: 0.046, standard deviation: 0.009), indicating that collaboration among UAVs is effective to increase mapping accuracy.

3.3 Hardware-in-the-loop (HIL) simulations

Simulations have been performed also to profile the developed algorithm when run on the RPi of the UAV. One drone has been connected through the serial port to a desktop, and interacted with the simulator through UDP messages. The UAV process onboard images from the dataset corresponding to the simulated cells, and decides the next cell to visit. The new waypoint is not sent to the autopilot but rather it is communicated to the simulator that implements the UAV motion. Similarly, ORCA is executed onboard and new waypoints are generated and sent to the simulator. We have performed several profiling tests to understand how much time is required for each operation. Overall, the module that takes longer time is `Perception`, which takes approximately 0.244 s in average, while `Core` takes about 0.107 s. Considering that `Perception` is executed only once per cell while the UAV is hovering, these values are compatible with field deployment, confirming that the proposed strategy can be reliably tested with real UAVs.

4 Conclusions

With this study, we have moved a fundamental step in the direction of field deployment for UAV swarms. We have described an efficient implementation on a real platform of a scalable coverage and mapping strategy based on reinforced random walks. The navigation strategy simplifies and improves over previous work [2], including collision avoidance among UAVs and also enhancing the random walk. The latter has been modified to take into account both the direction and the intensity of the bias vector resulting from attraction to beacons and repulsion from other agents. In this way, all available information are exploited for the benefit of both field coverage and weed mapping. In computing the results, we put particular care in the realism of the simulation, including artificial vision. Additionally, we implemented the proposed strategy on the real UAV hardware and tested it into hardware-in-the-loop simulations, verifying the suitability of the implementation prior to deployment with flying UAVs. Proof-of-concept videos of the real flying system is available online [16]. Our current effort is in collecting evidence of the suitability of the proposed strategy with more field tests, to be performed both indoor, using the mockup experimental scenario described in this paper, and outdoor, with tests performed on agricultural fields and real onboard classification of corps and weeds. The latter proves particularly challenging, due to the complexity of the vision routines that are prone to non-negligible errors. However, this paper suggests that swarms of UAVs can improve the detection accuracy beyond the individual limitation, and field tests will be dedicated to support and strengthen this concept. We are already moving the next steps toward future work, specifically by introducing a Bayesian estimator to reliably determine if a cell can be considered mapped or not. This is then used by the RRW to improve the exploration strategy based on the information that a specific region is expected to provide.

Acknowledgments This work has been supported by SAGA (Swarm Robotics for Agricultural Applications), an experiment founded by the European project ECHORD++ (GA: 601116). Dario Albani and Daniele Nardi acknowledge partial support from the European project FLOURISH (GA: 644227).

References

1. Albani, D., Manoni, T., Nardi, D., Trianni, V.: Dynamic UAV Swarm Deployment for Non-Uniform Coverage. In: AAMAS '18: Proceedings of the 2018 International Conference on Autonomous Agents and Multiagent Systems. pp. 1–9 (2018)
2. Albani, D., Nardi, D., Trianni, V.: Field coverage and weed mapping by UAV swarms. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS. pp. 4319–4325. IEEE (2017)
3. Avular. https://www.avular.com, accessed: 2018-04-22
4. Bareiss, D., van den Berg, J.: Generalized reciprocal collision avoidance. The International Journal of Robotics Research **34**(12), 1501–1514 (2015)
5. Brambilla, M., Ferrante, E., Birattari, M., Dorigo, M.: Swarm robotics: a review from the swarm engineering perspective. Swarm Intelligence **7**(1), 1–41 (2013)
6. Field robot event. http://www.fieldrobot.com/event/, accessed: 2018-04-22
7. Gerkey, B.P., Matarić, M.J.: A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. The International Journal of Robotics Research **23**(9), 939–954 (2004)
8. Hoffmann, H., Jensen, R., Thomsen, A., Nieto, H., Rasmussen, J., Friborg, T.: Crop water stress maps for an entire growing season from visible and thermal UAV imagery. Biogeosciences **13**(24), 6545–6563 (2016)
9. King, A.: Technology: The Future of Agriculture. Nature **544**(7651), 21–23 (2017)
10. Koeveringe, M., van Evert, F., Li, Y., Kootstra, G.: Detection of broad-leaved weed plants in grasslands, in preparation

11. Korsah, G.A., Stentz, A., Dias, M.B.: A comprehensive taxonomy for multi-robot task allocation. The International Journal of Robotics Research **32**(12), 1495–1512 (2013)
12. Nieuwenhuizen, A.T., Hofstee, J.W., van Henten, E.J.: Adaptive detection of volunteer potato plants in sugar beet fields. Precision Agriculture **11**(5), 433–447 (2009)
13. Peña, J.M., Torres-Sánchez, J., de Castro, A.I., Kelly, M., López-Granados, F.: Weed Mapping in Early-Season Maize Fields Using Object-Based Analysis of Unmanned Aerial Vehicle (UAV) Images. PLoS ONE **8**(10), e77151 (2013)
14. Popović, M., Vidal-Calleja, T., Hitz, G., Sa, I., Siegwart, R., Nieto, J.: Multiresolution mapping and informative path planning for UAV-based terrain monitoring. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1382–1388. IEEE (2017)
15. Sadat, S.A., Wawerla, J., Vaughan, R.: Fractal trajectories for online non-uniform aerial coverage. In: Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA 2011). pp. 2971–2976. IEEE (2015)
16. Saga experiment media center. http://laral.istc.cnr.it/saga/index.php/media-center, accessed: 2018-04-22
17. Stevens, A., Othmer, H.: Aggregation, Blowup, and Collapse: The ABC's of Taxis in Reinforced Random Walks. Siam Journal on Applied Mathematics **57**(4), 1044–1081 (1997)
18. Van Den Berg, J., Guy, S.J., Lin, M., Manocha, D.: Reciprocal n-Body Collision Avoidance. In: Pradalier, C., Siegwart, R., Hirzinger, G. (eds.) Robotics Research. pp. 3–19. Springer Berlin Heidelberg (2011)

5 Appendix - Additional Experiments

In this section we present additional data coming from some more experiments performed within this study. In particular, we present results for coverage and mapping time obtained by varying the number of robots involved in the simulation. As expected, both the mapping and the coverage problem benefit from the increased density of agents. We also observe that such increase in performance does not scale linearly due to non-beneficial interactions between the agents (i.e. issues related to overcrowding). Last, we observe a “right shift” in the minima of the mapping time t_m when the number of agents in the swarm increases. This is expected, as a larger number of agents increases the repulsion force acting on the single UAV, thus requiring higher attraction forces from the beacons to be effective.

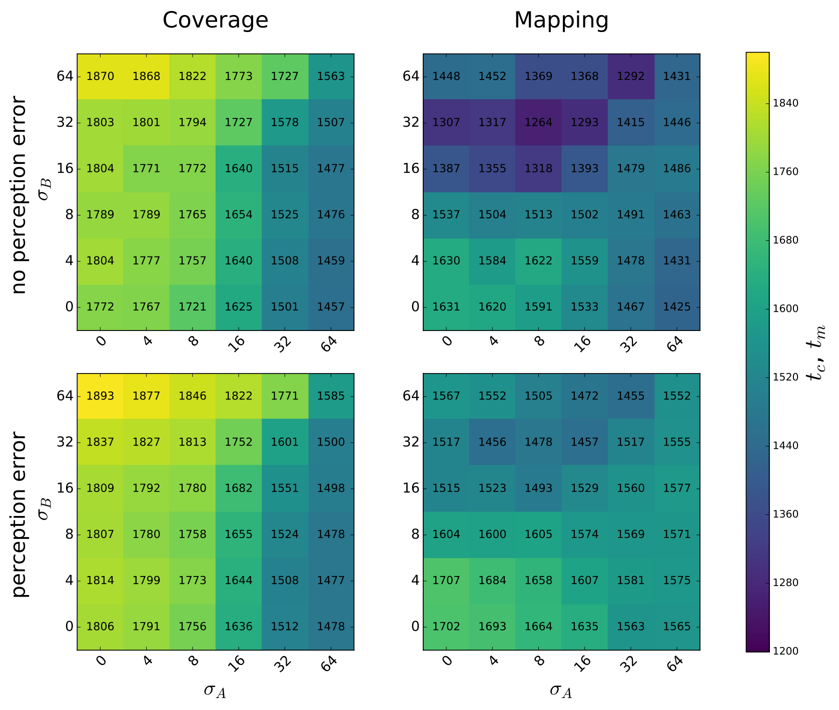

Fig. 5. Comparison of the coverage and mapping time for N=50 robots. Each cell in the heatmap represents the average of 150 runs.

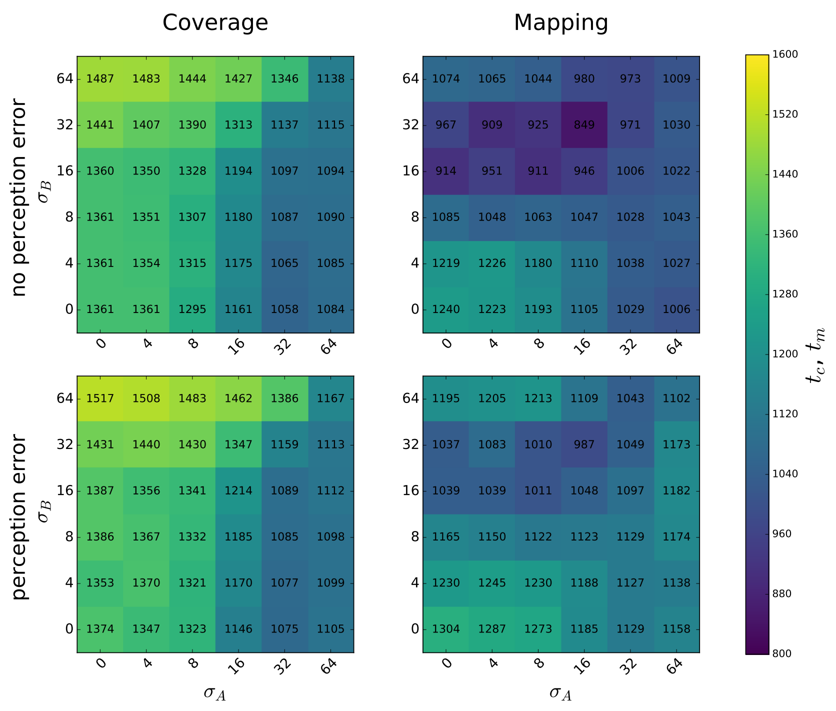

Fig. 6. Comparison of the coverage and mapping time for N=75 robots. Each cell in the heatmap represents the average of 150 runs.

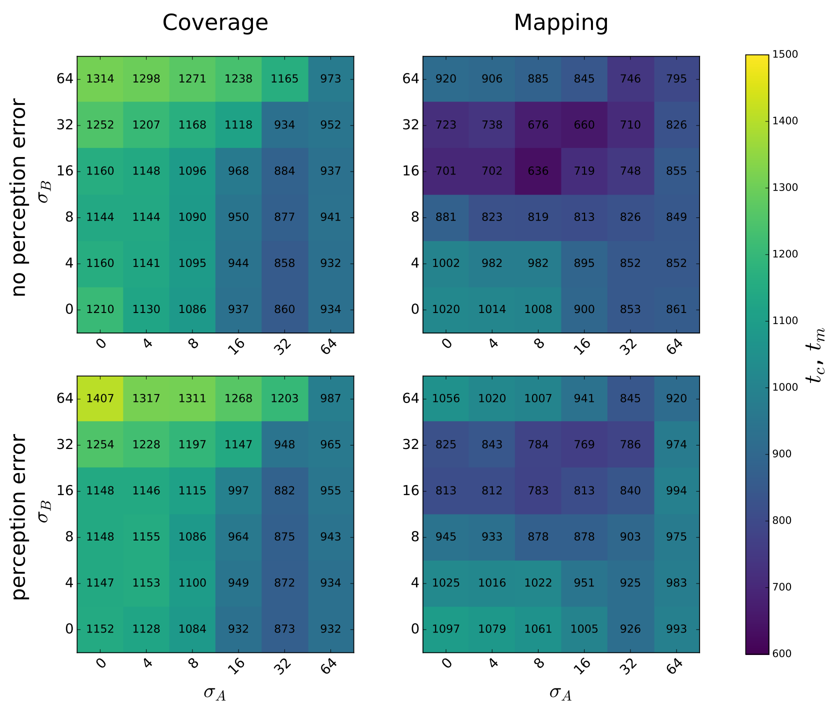

Fig. 7. Comparison of the coverage and mapping time for N=100 robots. Each cell in the heatmap represents the average of 150 runs.