2016 IEEE International Conference on Robotics and Automation (ICRA)
Stockholm, Sweden, May 16-21, 2016

Optimal Event Handling by Multiple Unmanned Aerial Vehicles

Martijn de Roo, Paolo Frasca, and Raffaella Carloni

Abstract— **This paper proposes a control architecture for a fleet of unmanned aerial vehicles that is responsible for handling the events that take place in a given area. The architecture guarantees that each event is handled by the required number of vehicles in the shortest time, while the rest of the fleet is optimally distributed in order to achieve an optimal coverage of the area. Each vehicle is steered to the specified location by a path planner that follows the shortest path while avoiding collisions. The control architecture has been validated both in simulations and in experiments on a fleet of three vehicles.**

I. Introduction

Thanks to their flexibility, fleets of autonomous Unmanned Aerial Vehicles (UAVs) are starting to be used for surveillance, inspection [1], search-and-rescue missions [2], and in agriculture [3]. In these application scenarios, the fleet of UAVs is responsible for a certain environment where events can randomly occur and the UAVs should be positioned so to guarantee an optimal coverage of the environment according to the probability of the events. Upon occurrence, every event should be quickly serviced by one or more UAVs, while the rest of the fleet should redistribute itself to maintain the optimal coverage.

This paper proposes a control architecture to achieve this optimal event handling by a fleet of UAVs. As shown in Figure 1, the control architecture consists of two parts, i.e., a *Group System* that acts based on *Events*, and an *Individual System* that controls each UAV. The *Group System* is in charge of ensuring the optimal event handling and the optimal coverage, in the following way. When an *Event* occurs, the *Event Handling* module determines which UAVs are the closest and then forwards the event location to the *Individual System* control of the required UAVs. In this work, the timing of the events is assumed to be unknown, therefore no sequential events are considered and the system works on a first-comes first-served basis. The *Coverage* module determines the placement of the UAVs before the mission starts and, once an event occurs, of all the UAVs that are not involved in the event handling. More specifically, the *Coverage* module takes into account a certain probability distribution of events occurring in an area and the possibility of requiring more than one UAV for an event. Optimality is defined in terms of minimizing the average time to reach the location of the event with the required number of UAVs.

This work has been funded by the European Commission's Seventh Framework Programme as part of the project SHERPA under grant no. 600958.

The authors are with the Faculty of Electrical Engineering, Mathematics and Computer Science, CTIT Institute, University of Twente, The Netherlands. Emails: m.deroo-1@alumnus.utwente.nl, {p.frasca,r.carloni}@utwente.nl.

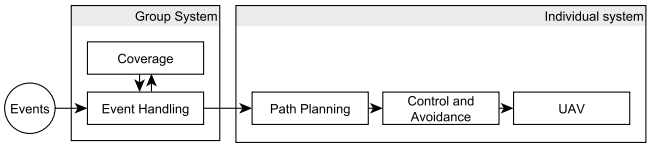

Fig. 1. The overall control architecture consists of the group system control and of the individual system control.

Allowing for more than one UAV to service a single event adds to the flexibility of the system: for instance, multiple UAV can cooperatively carry large loads that are unfeasible for single UAVs.

The classical solutions to similar coverage problems are based on Voronoi tessellations [4]–[6]. Voronoi tessellations divide the environment in a number of regions based on a set of generators (that is, the positions of the UAVs), on which the resulting regions give the area for which that generator is closer than any other generator. This work fits in this framework but goes beyond the state of the art by considering events that require *multiple* UAVs, i.e., the optimal coverage is defined by minimizing a novel cost function, which depends on using *multiple density functions* (as many as the maximum number of required UAVs). The optimization problem is solved by a gradient algorithm implemented in a central processing unit. However, in principle, the gradient dynamics can be implemented by each UAV in a distributed way, based on the positions of a limited number of neighbors [7], [8]. The analysis presented here assumes that the environment is a convex set, but is amenable to be extended to non-convex environments along the lines of [5].

The *Individual system* is responsible for moving each UAV from its original location to the desired position, i.e., either the location of the event or the position ensuring optimal coverage. The *Path Planning* module computes the path for each UAV, the *Control and Avoidance* module guarantees that the path is accurately tracked while avoiding collisions, and the *UAV* module is the physical UAV. In this work, a centralized approach is used that takes into account all the UAVs at the same time, together with the paths they follow and so to avoid possible collisions by changing the paths they follow [9]–[12].

The remainder of the paper is organized as follows. Section II describes the group system, the mathematical formulation of the coverage problem, and its gradient-descent solution. Section III describes the individual system, including path planning, path following, and collision avoidance. Sections IV and V present the results from simulations and experiments, respectively. Section VI concludes the paper.

II. Group System

The goal of the group system is optimally assigning UAVs to the events. In this work, optimality is defined as minimizing the average waiting time incurred by the events. This goal is achieved by deploying the group of UAVs in an optimal configuration (via the *Coverage* module) and by assigning, upon the appearance of an event, the closest UAVs to service it (via the *Event Handling* module). The key part here is optimizing the configuration (i.e., the positions) of the UAVs while they wait for the appearance of the events. This problem is formally defined in Section II-A and solved by on a gradient descent algorithm in Section II-B.

A. Optimal coverage problem

Let Ω be a bounded closed and convex domain of $\mathbb{R}^N$, where in our applications $N \in \{1, 2, 3\}$. In this domain, events take place according to a random spatial process. The events are heterogeneous in nature and thus require different numbers of UAVs to be serviced: consequently, we define for each $m \in \{1, \dots, M\}$ a smooth density function $\phi_m : \Omega \to \mathbb{R}_{>0}$, representing at each point in Ω the density of events requiring m UAVs. For each $i \in \{1, \dots, n\}$, we let p_i denote the position of UAV i, so that $p \in \Omega^n$ is the vector of the UAV positions. We let $||q - p_i||$ denote the Euclidean distance between p_i and a generic point q in Ω. For the sake of generality and in order to account for heterogeneities among the UAVs in terms of speed, we define for each $i \in \{1, \dots, n\}$ a smooth strictly increasing convex function $f_i : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, so that $f_i(||p_i - q||)$ represents the traveling time of UAV i from its current position to q. Whenever m UAVs are required by an event, we assume that the event cannot be serviced until it has been reached by all m UAVs: hence, the waiting time of the event is the traveling time of its m-th closest UAV. To formalize this fact, we shall write $m\text{-}\min S$ to denote the m-th smallest element in a finite set of real numbers S. With this notation in place, we are ready to define the following integral cost function, which represents the *expected service time of an event*,

$$\mathscr{J}_{exp}(p) = \sum_{m=1}^{M} \int_{\Omega} \underset{i \in \{1,\dots,n\}}{m\text{-}\min} f_i(||q - p_i||)\phi_m(q)\,\mathrm{d}q \quad (1)$$

and the corresponding optimization problem

$$\min_{p \in \Omega^n} \mathscr{J}_{exp}(p).$$

In the special case when $M = 1$, this cost reduces to

$$\int_{\Omega} \min_{i \in \{1,\dots,n\}} f_i(||q - p_i||)\phi(q)\,\mathrm{d}q.$$

The optimization of this cost has been extensively studied, see for instance [13], by using Voronoi tessellations of the domain Ω. In fact, also the cost (1) can be conveniently rewritten by using a suitable decomposition of Ω into sub-regions, in the following way.

Let W be a collection of measurable subsets of Ω such that each region W_i^m is indexed by $i \in \{1, \dots, n\}$ and $m \in \{1, \dots, M\}$ and, for each m, the sub-collection $\{W_i^m\}_{i \in \{1,\dots,n\}}$ is a *tessellation* of Ω: that is, $\cup_{i=1}^n W_i^m = \Omega$ for every m and $W_i^m \cap W_j^m$ is a set of measure zero if $i \neq j$. Note that the regions W_i^m are not assumed to be connected, i.e., a region can consist out of several disjoint sub-regions. Given such a collection W, we can define the auxiliary cost

$$\mathscr{J}(p, W) = \sum_{i=1}^{n} \sum_{m=1}^{M} \int_{W_i^m} f_i(||q - p_i||)\phi_m(q)\,\mathrm{d}q. \quad (2)$$

Note that this cost depends both on the positions p and on the regions W. By writing this cost, we assume that each UAV i as responsible for servicing the events happening in the region $\cup_{m=1}^M W_i^m$: in region W_i^1 it shall service all events, in region W_i^2 it shall service all events that require at least two UAVs, and so forth.

In order to link the generic cost (2) with the expected cost (1), we need to define one specific collection Z. Given a configuration p with no coincident positions, let us define the region $Z_i^m(p)$ as the subset of Ω such that i is the m^{th} closest UAV to the points in $Z_i^m(p)$. More formally, let $Z_i^m(p) = \{q \in \Omega : \exists\, H \subset \{1 \dots n\}$ such that $|H| = m - 1,\ i \notin H$ and $f_h(||q - p_h||) \leq f_i(||q - p_i||) \leq f_\ell(||q - p_\ell||)$ for all $h \in H,\ \ell \notin H,\ \ell \neq i\}$. Note that the sub-collection $\{Z_i^1(p)\}_{i \in \{1,\dots,n\}}$ is just the classical Voronoi tessellation generated by p. The dependence on p will be dropped in what follows, provided this causes no ambiguity.

By virtue of its definition, it is clear that $Z(p)$ minimizes $\mathscr{J}(p, W)$ as a function of W for fixed p. That is, we have

$$\mathscr{J}_{exp}(p) = \mathscr{J}(p, Z(p)) = \min_{W} \mathscr{J}(p, W)$$

for every $p \notin C$, where $C = \{p \in \Omega^n : \exists i, j \text{ s.t. } p_i = p_j\}$. This fact also permits to show –along the lines of [13, Theorem 2.16]– that the function $\mathscr{J}_{exp}$ is Lipschitz continuous on its domain and continuously differentiable outside the set of coincident configurations C.

Constructing the regions Z_i^m: For the sake of clarity and concreteness, we now illustrate with the help of Figure 2 how to construct such regions Z_i^m for an example with $n = 3$ and $m = 2$. First, we construct the Voronoi regions Z_i^1 for each i. Next, in order to get the second-closest tessellation of the Voronoi region Z_i^1, we leave out UAV i from the generators and construct the Voronoi tessellation generated by $\{1, \dots, n\} \setminus \{i\}$: its $n - 1$ regions are denoted by X_k^i. Note that $\cup_{k \neq i} X_k^i = \Omega$. By taking the intersection of the new regions X_k^i and the regions Z_i^1 for all $i \in \{1, \dots, n\}$ and $k \neq i$ we get $T_k^{2,i} = Z_i^1 \cap X_k^i$. Finally, for all $i \in \{1, \dots, n\}$ we define $Z_i^2 = \cup_{\ell \neq i} T_i^{2,\ell}$.

B. Gradient descent optimization

The minimization of the coverage function can be sought by following its gradient by the dynamics

$$\dot{p}_k = -\kappa \frac{\partial \mathscr{J}_{exp}(p)}{\partial p_k} \quad \forall\, k \in \{1, \dots, n\}, \quad (3)$$

where $\kappa \geq 0$. However, the coverage function is not convex, so a gradient descent will not in general reach a global minimum. We shall now compute the gradient under the

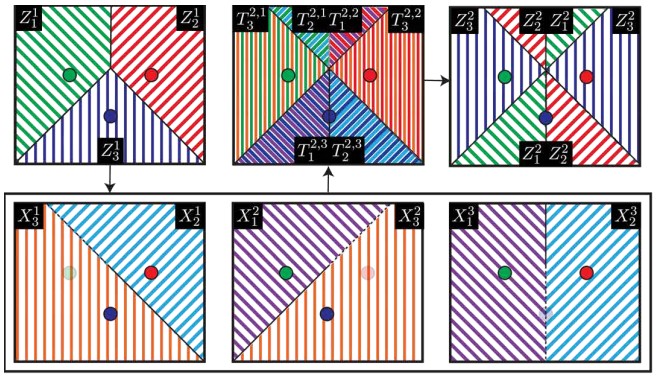

Fig. 2. Example of the construction of the regions Z_i^m with $n = 3$ and $m = 2$.

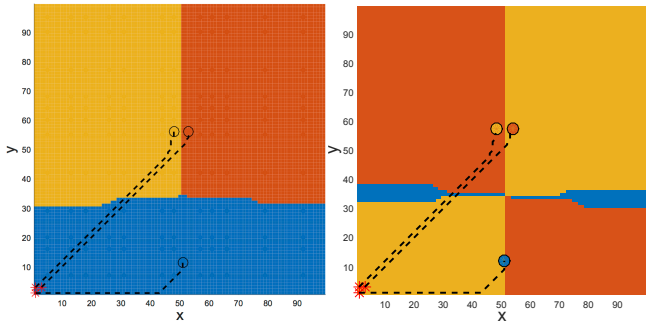

Fig. 3. Simulation of the group system for $n = 3$, $M = 2$ on a 100x100 grid. Virtual trajectories of the UAVs from the initial locations (star) to the final locations (circles) are drawn as dashed black lines; colors illustrate responsibility regions at convergence (the blue region corresponds to the blue UAV, and so on): Z^1 (left plot) and Z^2 (right plot).

simplifying assumption that $f_i(x) = x^2$ for all i: this assumption, according to [5], is often adequate in practice and can be relaxed at the cost of more involved notation. Since $J_{exp}(p) = J(p, Z(p))$, we have

$$\frac{\partial \mathscr{J}_{exp}(p)}{\partial p_k} = \frac{\partial}{\partial p_k} \sum_{m=1}^{M} \sum_{i=1}^{n} \int_{Z_i^m} ||p_i - q||^2 \phi_m(q)\, \mathrm{d}q \tag{4a}$$

$$= \sum_{m=1}^{M} \sum_{i=1}^{n} \int_{Z_i^m} \frac{\partial}{\partial p_k} ||p_i - q||^2 \phi_m(q)\, \mathrm{d}q \tag{4b}$$

$$= 2 \sum_{m=1}^{M} \sum_{i=1}^{n} \int_{Z_i^m} ||p_i - q|| \frac{\partial}{\partial p_k} ||p_i - q|| \phi_m(q)\, \mathrm{d}q \tag{4c}$$

$$= 2 \sum_{m=1}^{M} \int_{Z_k^m} ||p_k - q|| \frac{\partial}{\partial p_k} ||p_k - q|| \phi_m(q)\, \mathrm{d}q, \tag{4d}$$

where for step (4a) to (4b) it is allowed to take the derivative under the integral according to [6, Proposition 2] and for step (4c) to (4d) the summation is removed because $\frac{\partial}{\partial p_k} ||p_i - q|| = 0$ if $k \neq i$. Since $\frac{\partial}{\partial p_i} ||p_i - q|| = \frac{p_i - q}{||p_i - q||}$, Equation (3) becomes

$$\dot{p}_i = -2\kappa \sum_{m=1}^{M} \int_{Z_i^m} (p_i - q) \phi_m(q)\, \mathrm{d}q, \tag{5}$$

which corresponds to steering each UAV towards a weighted "centre of mass" of its own responsibility region $\cup_m Z_i^m$. Some convergence properties of the gradient dynamics are stated in the next result.

Theorem 1: The trajectories $p(t)$ of the dynamics (3) converge to a set contained in

$$\{q \in \Omega^n \,:\, \nabla \mathscr{J}_{exp}(q) = 0\} \cup C.$$

Proof: First of all, note that the Lie derivative, that is the change in the objective function,

$$\frac{\mathrm{d}\, \mathscr{J}_{exp}(p(t))}{\mathrm{d}t} = \sum_{i=1}^{n} \frac{\partial \mathscr{J}_{exp}(p(t))}{\partial p_i} \dot{p}_i$$

$$= -\kappa \sum_{i=1}^{n} \left(\frac{\partial \mathscr{J}_{exp}(p(t))}{\partial p_i} \right)^2$$

is negative semidefinite. However, since $\mathscr{J}_{exp}(p)$ is not continuously differentiable in C, the classical LaSalle principle can not be used. Instead, we resort to the generalized invariance principle in [13, E1.8]. Note that the trajectories $p(t)$ stay in the compact set Ω^n, but the set $\Omega^n \setminus C$ is not closed and is dense in Ω^n. Then, the generalized invariance principle implies convergence of the solutions to a subset of the union between C and the largest invariant subset contained in the zeros of the Lie derivative, that is, $\{q \in \Omega^n \,:\, \nabla \mathscr{J}_{exp}(q) = 0\}$. ■

In words, we have proved that the gradient dynamics converges to either the critical points of the cost or to configurations with coincident positions. Note that possible convergence to coincident positions is not an artefact of our analysis but instead an inherent feature of this optimization problem, as examples show that local minima of $\mathscr{J}_{exp}$ can in fact involve coincident positions. Clearly, the implementation of such minima would be approximate, so to avoid collisions between UAVs. The convergence of (3) is illustrated in Figure 3 in a simple square environment. For the implementation of the control law in equation (5), we perform a discretization of the environment, resulting in an approximation of the algorithm. Similar discretizations are done in [5], [14].

III. Individual System

The goal of the individual system is to move the UAV(s) to the location where the event occurs. To this goal, path planning, path following, obstacle avoidance, and attitude control should be implemented on each UAV. A detailed overview of the Individual System controller of a UAV is shown in Figure 4.

A. Quadcopter UAV dynamics

In this work, quadcopter UAVs are considered, as schematically depicted in Figure 5. The dynamics of a quadcopter is based on the model presented in [15], on which a drag force for each angular velocity has been added in the inertial frame.

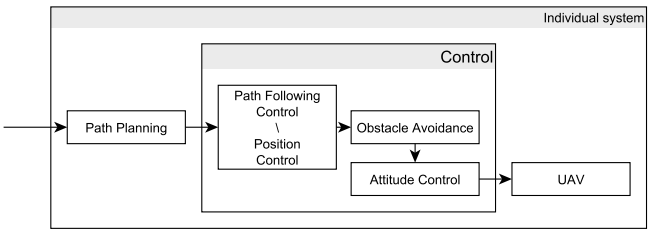

Fig. 4. Control scheme of a UAV.

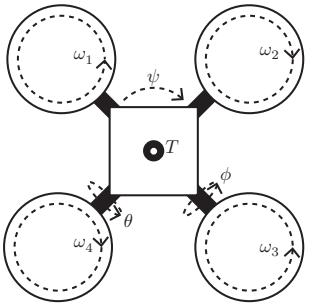

Fig. 5. Quadcopter with the angular velocities of the rotors, ω_1, ω_2, ω_3 and ω_4; the roll, pitch and yaw are the angles ϕ, θ and ψ; the thrust is indicated by T.

Thus, the dynamics is given by:

$$m\ddot{\xi} = -mge_z + Re_zT - C\dot{\xi}$$
$$I\dot{\Omega} = -\Omega \times I\Omega - G + \tau$$
$$T = b\sum_{i=1}^{4}\omega_i^2$$
$$\tau = \begin{bmatrix} d\ b\ (\omega_2^2 - \omega_4^2) \\ d\ b\ (\omega_1^2 - \omega_3^2) \\ \kappa\ (\omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2) \end{bmatrix}$$
$$G = \sum_{i=1}^{4} I_r(\Omega \times e_z)(-1)^{i+1}\omega_i$$

where $m \in \mathbb{R}$ and $I \in \mathbb{R}^3$ are the mass and the diagonal inertial matrix of the UAV with respect to the body-fixed frame, $\xi \in \mathbb{R}^3$ and $R \in SO(3)$ the position and the rotation matrix of the body-fixed frame in the inertial frame, $g \in \mathbb{R}$ the gravity acceleration, $T \in \mathbb{R}$ the thrust applied to the airframe, $C \in \mathbb{R}^{3\times3}$ the diagonal drag force matrix, ω_i, with $i = (1, \cdots, 4)$ the rotors' angular velocities, $\Omega \in \mathbb{R}^3$ the angular velocity of the airframe in the body-fixed frame, G the gyroscopic forces, and τ the external torque. The constants are $b \in \mathbb{R}$ (the proportionality constant of the rotor), $d \in \mathbb{R}$ (the distance of the rotor to the airframe's center of mass) and $\kappa \in \mathbb{R}$ (the rotor thrust factor).

The rotation matrix R is expressed as the Z_ψ - Y_θ - X_ϕ (yaw - pitch - roll) Euler angles to map the inertial and the body-fixed frame, resulting in

$$R = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta c\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}$$

where for brevity $s = \sin$ and $c = \cos$.

B. Path planning

Once the event location is known, a path has to be generated for the UAV to reach the desired location. An overview on different path planning methods is given in [16].

In this work, the path is planned though a graph-based approach, in which the environment is discretized into a grid. To find the path on a grid, several methods can be used. The A* algorithm [17] or the Dijkstra's algorithm [18] are well known algorithms and they both ensure that the planned path is the shortest path between the initial location to the desired one. The A* algorithm has been chosen because, in general, it is faster thanks to the possibility of selecting heuristics on a preferred direction. The A* algorithm generates a set of points on the planned path.

Figure 6 shows the result of the A* algorithm, as simulated on a 100×100 grid for three UAVs in Matlab/Simulink (MathWorks, Natick, MA, USA). The three UAVs starts from a position close to $(0, 0)$ and the algorithm gives the shortest path for each UAV to reach the target. In case several paths have the same length, the shortest path that has no overlap with other paths is selected to reduce the risk of collisions.

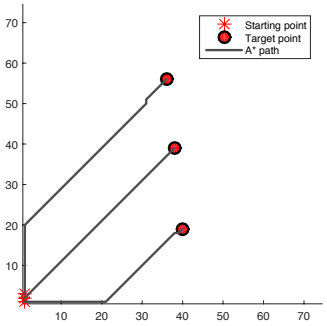

Fig. 6. A* shortest path in a 100x100 grid.

C. Control of the quadcopter UAV

Once a path has been planned, each UAV has to follow the path and stabilize in the location of the event. Commercially available quadcopters have built-in PID attitude control and, therefore, this section describes only the position control, path following and collision avoidance.

1) Position control: The position controller is implemented as a PID controller [19], [20]. If a high gain attitude control is considered and small angles are assumed for the roll and pitch, the set-points for the roll and pitch can be linearized and be used as the velocity set-points.

2) Path following: The path following method is similar to the ones described in [21], [22]. The position error that needs to be minimized is defined in the normal direction to the line that goes from the previous waypoint to the next waypoint, while the desired velocity is defined in the tangent direction of the path. This allows to move the UAV through a set of waypoints without stopping or converging to each of them, which happens if position control is used for the

waypoints. The path is defined as a sequence of waypoints $x_i \in \mathbb{R}^3$. Let v_i be the desired velocity along the path segment connecting waypoint i to $i+1$, and $x(t)$ the current position. The cross-track e_{ct} error is given by

$$e_{ct} = \|x_i - x(t)\|\tilde{n} \tag{6}$$

where $\tilde{n}$ is the unitary vector, normal to the path segment between x_{i-1} and x_i. The position control is designed to minimize the cross-track error. The transition to the next path segment occurs when the UAV is $c_{error} = \|x_i - x(t)\|\tilde{b}$ far from the next path segment, where $\tilde{b}$ is the unitary vector, parallel to the path segment between x_{i-1} and x_i. The path following is schematically depicted in Figure 7.

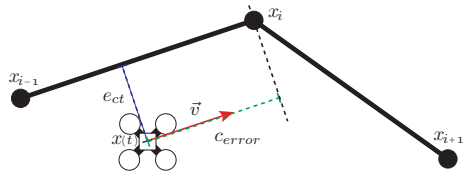

Fig. 7. Path following method.

The position controller requires a desired velocity. To get this velocity we propose to make the velocity dependent on the curvature of the path. The more curved the path is, the slower the desired velocity is. That means that v_{max} would be a straight line and v_{min} a 90° turn. To get the curvature for the waypoints we approximate it by using a circle, computed with three consecutive waypoints to find a unique radius that can represent the curvature. A large radius would indicate an almost straight line, while a small radius would indicate a sharp turn. The circle of three points can be determined by solving the following determinant

$$\det \begin{vmatrix} x^2+y^2 & x & y & 1 \\ x_1^2+y_1^2 & x_1 & y_1 & 1 \\ x_2^2+y_2^2 & x_2 & y_2 & 1 \\ x_3^2+y_3^2 & x_3 & y_3 & 1 \end{vmatrix} = 0 \tag{7}$$

where (x, y) is the set of points of the circumcircle and (x_i, y_i) the i-th point. The determinant is $(x - x_0)^2 + (y - y_0)^2 = r^2$, resulting in a radius r from the three points.

The radius r is ∞ when all the points are on a straight line. The desired velocity is based on the inverse of the radius and is given by $v_{des}(r) = f(r) + v_{min}$, where

$$f(r) = \begin{cases} (v_{max} - v_{min}) - c\frac{1}{r} & \text{if } (v_{max} - v_{min}) - \frac{1}{r} > 0 \\ 0 & \text{if } (v_{max} - v_{min}) - \frac{1}{r} \leq 0 \end{cases}$$

and $c \in \mathbb{R}$ is a constant.

3) Collision avoidance: To ensure that no collision occur among the UAVs, a collision avoidance algorithm is included in the overall controller,where a circle of a radius that sets a safety margin is considered around each UAV.

IV. Simulations

Simulations are used to validate the proposed control architecture for optimal event handling by multiple UAVs.

The type of UAV that has been used is the quadcopter Crazyflie 2.0 (Bitcraze AB, Sweden), which has a built-in attitude control and whose parameters have been reported in [23]. The dynamic simulations of the system and of the overall control architecture have been made using the Robot Operating System (ROS, `www.ros.org`), the robot simulator Gazebo (`http://gazebosim.org`) and Matlab. ROS and Gazebo allow for an easy transfer from simulation code to experimental code: the model for the quadcopter is derived from [24]. Matlab has been used for the group system and the Matlab ROS I/O add-on has been used for the communication between Matlab and ROS.

A. Simulation results of the group system

The implementation of the group system of three UAVs has been realized in Matlab and is shown in Figure 8.

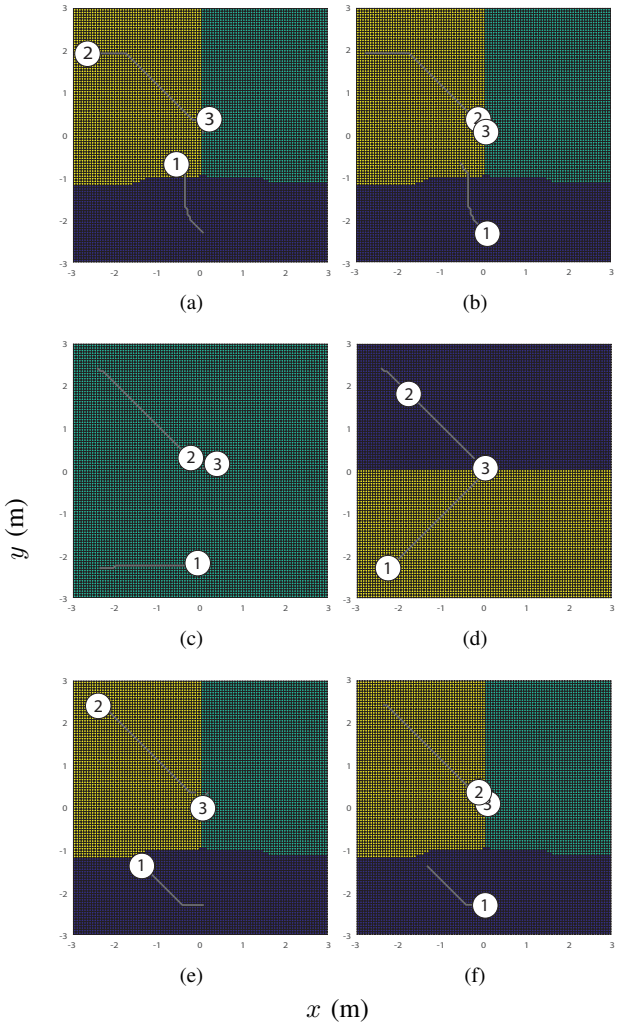

Fig. 8. Example of the group system showing the UAVs and their paths.

The desired position in which an event appears is decided by the user and the optimal distributions of the three UAVs are calculated using the results of the optimal coverage problem. In the figure, the background shows the regions of each UAV. In Figure 8(a) the system is enabled, from the

current positions of the UAVs a path is generated towards the optimal location of the UAVs according to the optimal coverage for three UAVs and the desired positions are achieved, as shown in Figure 8(b). In Figure 8(c), two points in the area are selected, towards which paths are generated for the two closest UAVs and the available UAV 3 is sent to the optimal coverage position, i.e., the centre. In Figure 8(d) UAV 1 reaches the desired position, and the system sends it back to the optimal coverage for two UAVs, which requires both available UAVs in the middle. When UAV 2 reaches its desired position in Figure 8(e), three UAVs are available, for which the systems generates paths to move the UAVs, the final positioning is then achieved Figure 8(f), which is similar to Figure 8(b). Figure 9 shows the simulated environment in Gazebo, in which the quadcopter has been implemented as in [24].

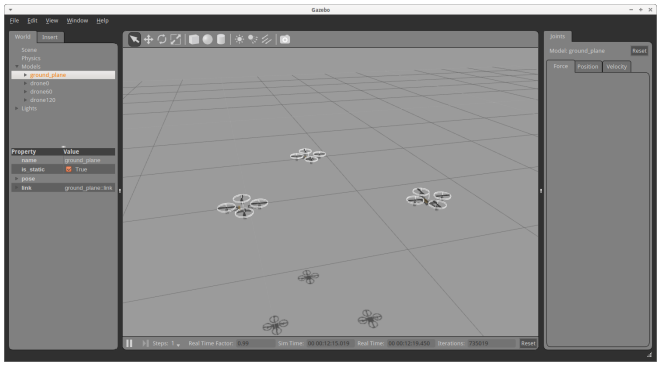

Fig. 9. Example of the Gazebo simulation environment with three UAVs.

B. Simulation results of the individual system

The velocity profile has been generated with $v_{max} = 0.065$ rad, $v_{min} = 0.015$ rad and $c = 0.15$. For the collision avoidance a radius of 50 cm has been used around each UAV.

Two simulations have been made. In the first simulation, the quadcopter UAV follows a Lissajous figure in order to test the path following controller and the velocity profile generation. Figure 10 shows the results of the simulated model of a Crazyflie 2.0 while tracking the Lissajous figure. The trajectory is closely followed with a cross-track error of about 10 cm in the corners.

In the second simulation, shown in Figure 11, two identical quadcopters UAVs follow the same Lissajous figure by starting with opposite positions and by moving in opposite directions in order to test the collision avoidance. The two UAVs, on opposite ends, are started at the same time and meet each other in the middle: at this point, the obstacle avoidance takes control due to the quadcopters heading into a collision and ensures that the quadcopters do not collide. The trajectory is closely followed with a cross-track error of about 12 cm when the two UAVs are in close vicinity.

V. Experiments

The proposed control architecture has been validated thought experiments.

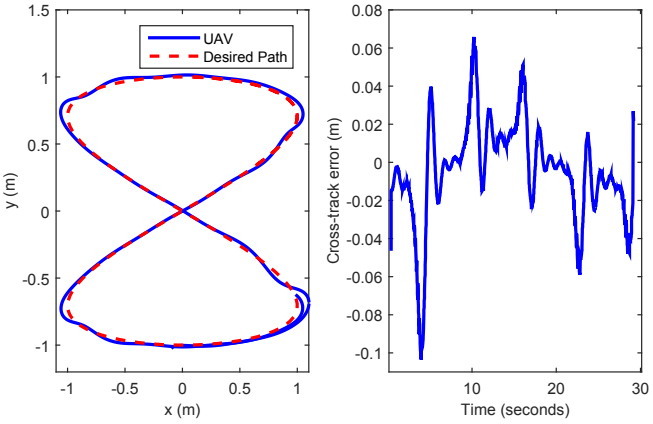

Fig. 10. Simulation in ROS/Gazebo: path following of a Lissajous figure, showing both the path and the cross-track error (10 cm at its largest).

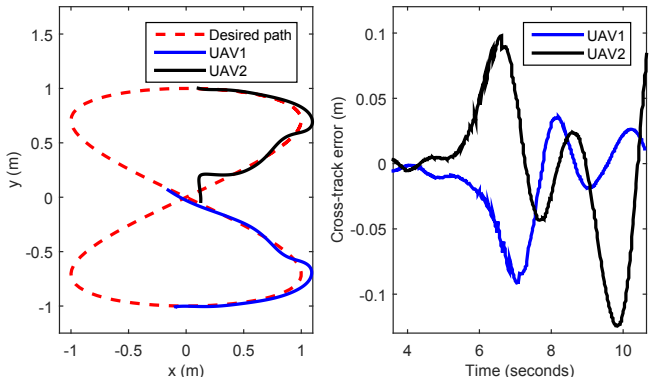

Fig. 11. Simulation in ROS/Gazebo: path following of a Lissajous figure by two quadcopter avoiding each another, showing both the path and the cross-track error (12 cm at its largest).

The overall hardware architecture is shown in Figure 12. The experiments have been conducted in an indoor test area, within which the OptiTrack optical tracking system (Naturalpoint, Inc., Corvalis, USA) has been used as external positioning system that allows a 100 Hz position and attitude tracking. The Crazyflies 2.0 have been endowed with retroreflective markers held by carbon tubes, as is shown in Figure 13. During the experiment Matlab has been used for the group system and ROS/Gazebo for the individual system.

Experiments have been performed to test both the individual systems and the overall group system. For the group system, the simulation results could be replicated as reported in Figure 8. As for the individual system, the first simulated test has been reproduced on the experimental set-up. The results of the experiment are shown in Figure 14. The Lissajous figure is tracked by the UAV but it shows a larger error when compared to the simulations. The trajectory is closely followed with a cross-track error of about 15 cm. We believe that this small degradation of the performance is due to the weight of the retroreflective markers, which is not included in the dynamical model of the UAVs.

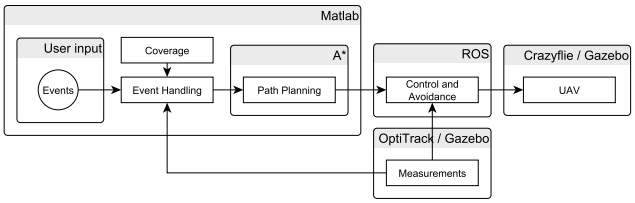

Fig. 12. Overview of the experimental setup.

Fig. 13. The Crazyflie 2.0 quadcopters.

VI. Conclusion

A control architecture has been proposed for the optimal handling of events by means of a fleet of UAVs. The architecture comprises both a group system and an individual system. The group system solves the optimal coverage problem, while taking into account that events can require multiple UAVs. For the individual system, each vehicles is steered to the desired location according to a path planner that seeks the shortest path while avoiding collisions.

VII. Acknowledgements

The authors thank Martijn Weijers for contributing, during his MSc thesis, to debugging the experimental tests.

References

[1] L. Marconi *et al.*, "Aerial service robots: An overview of the AIRobots activity," in *Proceedings of the IEEE International Conference on Applied Robotics for the Power Industry*, 2012, pp. 76–77.

[2] ——, "The SHERPA project: Smart collaboration between humans and ground-aerial robots for improving rescuing activities in alpine environments," in *Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics*, 2012, pp. 1–4.

[3] B. Stark, S. Rider, and Y. Chen, "Optimal pest management by networked unmanned cropdusters in precision agriculture: A cyber-physical system approach," in *IFAC Workshop on Research, Education and Development of Unmanned Aerial Systems*, 2013, pp. 296–302.

[4] J. Cortes, S. Martinez, and F. Bullo, "Spatially-distributed coverage optimization and control with limited-range interactions," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 11, no. 04, pp. 691–719, 2005.

[5] S. Bhattacharya, R. Ghrist, and V. Kumar, "Multi-robot coverage and exploration in non-Euclidean metric spaces," in *Algorithmic Foundations of Robotics X*. Springer, 2013, pp. 245–262.

[6] L. Pimenta, V. Kumar, R. Mesquita, and G. Pereira, "Sensing and coverage for a network of heterogeneous robots," in *IEEE Conference on Decision and Control*, 2008, pp. 3947–3952.

[7] J. Cortés, S. Martínez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Robotics and Automation Magazine*, vol. 20, no. 2, pp. 243–255, 2004.

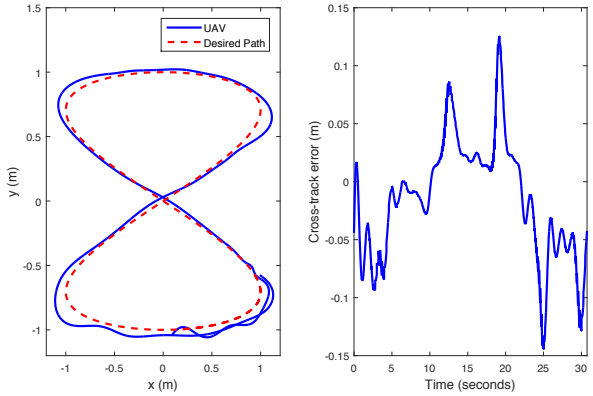

Fig. 14. Experiment: path following of a Lissajous figure, showing both the path and the cross-track error (15 cm at its largest).

[8] F. Bullo, R. Carli, and P. Frasca, "Gossip coverage control for robotic networks: Dynamical systems on the space of partitions," *SIAM Journal on Control and Optimization*, vol. 50, no. 1, pp. 419–447, 2012.

[9] D. M. Stipanović, P. F. Hokayem, M. W. Spong, and D. D. Šiljak, "Cooperative avoidance control for multiagent systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, no. 5, pp. 699–707, 2007.

[10] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *IEEE International Conference on Robotics and Automation*, 2008, pp. 1928–1935.

[11] P. Surynek, "An optimization variant of multi-robot path planning is intractable," in *AAAI Conference on Artificial Intelligence*, 2010, pp. 1261–1263.

[12] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics research*. Springer, 2011, pp. 3–19.

[13] F. Bullo, J. Cortés, and S. Martínez, *Distributed Control of Robotic Networks*. Princeton University Press, 2009.

[14] J. W. Durham, R. Carli, P. Frasca, and F. Bullo, "Discrete partitioning and coverage control for gossiping robots," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 364–378, 2012.

[15] T. Hamel, R. Mahony, R. Lozano, and J. Ostrowski, "Dynamic modelling and configuration stabilization for an X4-flyer," in *IFAC World Congress*, 2002, pp. 846–846.

[16] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: A review," *IEEE Access*, vol. 2, pp. 56–77, 2014.

[17] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[18] M. Sniedovich, "Dijkstra's algorithm revisited: the dynamic programming connexion," *Control and Cybernetics*, vol. 35, no. 3, pp. 599–620, 2006.

[19] M. Fumagalli, R. Naldi, A. Macchelli, R. Carloni, S. Stramigioli, and L. Marconi, "Modeling and control of a flying robot for contact inspection," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 3532–3537.

[20] H. W. Wopereis, M. Fumagalli, S. Stramigioli, and R. Carloni, "Bilateral human-robot control for semi-autonomous UAV navigation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 5234–5240.

[21] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-UAV testbed," *IEEE Robotics Automation Magazine*, vol. 17, no. 3, pp. 56–65, 2010.

[22] G. M. Hoffmann, S. L. Wasl, and C. J. Tomlin, "Quadrotor helicopter trajectory tracking control," in *AIAA Guidance, Navigation, and Control Conference*, 2008.

[23] W. Hanna, "Modelling and control of an unmanned aerial vehicle," Ph.D. dissertation, Charles Darwin University, 2014.

[24] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, and O. von Stryk, "Comprehensive simulation of quadrotor UAVs using ROS and Gazebo," in *International Conference on Simulation, Modeling and Programming for Autonomous Robots*, 2012, pp. 400–411.