IFAC papers online, Vol. 50, Issue 1, July 2017, pp. 8011-8018
DOI:10.1016/j.ifacol.2017.08.1225

Cooperative Control for a Flight Array of UAVs and an Application in Radar Jamming ⋆

Inmo Jang * **Junho Jeong** ** **Hyo-Sang Shin** *
Seungkeun Kim ** **Antonios Tsourdos** * **Jinyoung Suk** **

* *Cranfield University, Cranfield, Bedfordshire, MK430AL, UK (e-mail: {inmo.jang, h.shin, a.tsourdos}@cranfield.ac.uk).*
** *Chungnam National University, Daejeon 34134, Republic of Korea (e-mail: {junho, skim78, jsuk}@cnu.ac.kr)*

Abstract:
This paper proposes a flight array system and an integrated approach to cope with its operational issues raised in mission-planning level (i.e., task allocation) and control level (i.e., control allocation). The proposed flight array system consists of multiple ducted-fan UAVs that can assemble with each other to fly together as well as disassemble themselves to fly individually for accomplishing a given mission. To address the task allocation problem, a game-theoretical framework is developed. This framework enables agents to converge into an agreed task allocation in a decentralised and scalable manner, while guaranteeing a certain level of global optimality. In addition, this paper suggests a cooperative control scheme based on sliding mode control and weighted pseudo-inverse techniques so that the system's non-linearity and control allocation issue are effectively handled. As a proof-of-concept, a prototype simulation program is developed and validated in a cooperative jamming mission. The numerical simulations manifest the feasibility of effectiveness the proposed approach.

Keywords: Cooperative systems, Coordination of multiple vehicle systems, Multi-agent systems, Flying robots, Task allocation, Nonlinear cooperative control

1. INTRODUCTION

Multiple-agent systems consisting of a large number of small-sized vehicles, thanks to technological advances making them more intelligent, miniaturised, and affordable, have been attracting many interests from various domains (Sahin (2005)). From the perspective of aerial robotics, however, one of significant challenges is to make them fly in the presence of external disturbances, e.g., winds or gusts (Habib et al. (2011); Samad et al. (2007)), which will be likely omnipresent in many practical mission environments. Owing to the small sizes of these UAVs (denoted by micro UAVs), they are very sensitive to the unfavourable wind conditions, which could call into question the success of their operations. This issue could be exacerbated in complex environments such as an urban environment.

A potential remedy to this issue is the concept of a flight array, which consists of multiple UAVs that can physically assemble with each other and fly together. As more agents are assembled into a flight array, the whole system possesses higher control power, overcoming the external disturbances. A flight array of micro UAVs, due to their low cost and expendability, can be an appropriate candidate for cooperative jamming or communication relay missions, where precision control of each UAV would not be as vital as in other missions.

In operation of such a system, there are decision-making issues raised in mission-planning level and control level. In order to cooperatively and efficiently accomplish given tasks, the system needs to determine which agent has to be assigned to which task, i.e., *task allocation problem*, and this decision-making responsibility should reside within each vehicle. Furthermore, when the system flies as a flight array, due to its over-actuation from the actuators of multiple agents being assembled, the issue of how to distribute a desired control power amongst the actuators, i.e., *control allocation problem*, has to be addressed.

This paper proposes our flight array system and presents an approach to deal with the aforementioned issues for the system. For the task allocation problem, a game-theoretical decision-making framework is utilised that not only enables agents to converge into an agreed assignment in a decentralised and scalable manner but also provides a certain level of guaranteed global optimality. For control of the flight array system, we suggest a control scheme based on sliding mode control and weighted pseudo-inverse matrix technique, because of its computational efficiency and inherent simplicity, to address the system's nonlinearity and control allocation issues.

As one of possible applications, a cooperative *stand-in* jamming mission is formulated, where a number of micro

⋆ The authors gratefully acknowledge that this research was supported by International Joint Research Programme with Chungnam National University (No. EFA3004Z).

UAVs penetrate into enemy territory and cooperatively use their jamming resources for their allies not to be tracked by adversary radars. A prototype simulation program is implemented for this mission in order to investigate the feasibility and effectiveness of our proposed approach.

2. FLIGHT ARRAY SYSTEM DESIGN

The basic concept of a flight array system was initially presented by Oung and D'Andrea (2011), in which individual agents are able to autonomously assemble with each other on the ground, and then fly together in a coordinated fashion. Although each agent can produce sufficient thrust to lift itself into the air, however, on its own it is too unstable to fly because it has just a single propeller without any additional controller. The system may not be suitable for some missions that require its agents to fly individually, being separated from a flight array.

Our proposed flight array system consists of multiple *ducted-fan* UAV that can also individually fly and hover. The duct of each UAV is equipped with six electromagnets, by which the UAV can physically assemble with other UAVs, as shown in Figure 1. The flight array system can be reconfigured by connecting to or separating from a single vehicle or another flight array, thereby enhancing its payload capacity and controllability. Thanks to the reconfigurability, the flight array system can be utilised for various missions such as communication relay, radar jamming, reconnaissance, and wide area surveillance, as shown in Figure 2 (Jeong et al. (2016)).

A single vehicle is mainly composed of a fuselage, a propulsion system, a duct, a stator, and control flaps (Jeong et al. (2015)). The fuselage is separately located on the top and bottom of the vehicle, being equipped with necessary avionics and batteries. For the propulsion system, an electric motor is used for minimising vibration, which enables the UAV to construct a flight array even during flight. The stator reduces the anti-torque effect caused by the rotor of the propulsion system. For the attitude control of the vehicle, four control flaps are used: pitching or rolling motion can be generated by using either of two oppositely-positioned flap pairs; and yawing motion can be generated by deflecting all the flaps to the same rotational direction.

3. COOPERATIVE STAND-IN RADAR JAMMING

As one of possible applications that exploit the proposed flight array system, this paper particularly considers a cooperative *stand-in* radar jamming mission. This mission

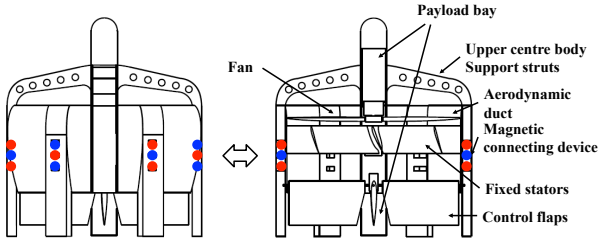

Fig. 1. Configuration design of the individual UAV (Left: side view, Right: cross-section view)

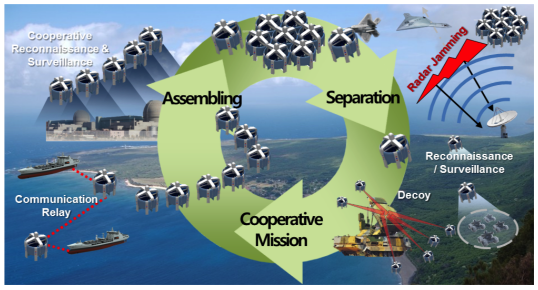

Fig. 2. The concept of our flight array system and its possible applications

can be categorised as "escort jamming", in which multiple agents have ECM (Electro Counter Measure) transmitters and provide jamming to protect their allies from adversary radars. One of key differences from existing *stand-off* jamming missions is that agents for stand-in jamming penetrate into enemy territory and neutralise targets while being located nearby.

For such a stand-in radar jamming mission, micro UAVs can be an appropriate candidate. This is because the UAVs can take advantage of proximity effect by being close enough to the adversary radars, thanks to their small RCS (Radar Cross Section) as well as fault tolerance as a multi-agent system. Although each of the UAVs has limited jamming resources, they can cooperatively exhibit enough jamming effectiveness by superimposing their signal strengths.

For now, we give a brief description of our prototype mission scenario, which was also used in the simulation experiment in Section 6. In this scenario, a set of micro UAVs (agents) are deployed to cooperatively jam a set of adversary radars (tasks). Initially, the agents fly as a flight array (in the flight-array mode) to a certain position in order to overcome winds or gusts. When reaching the position, they begin a decision-making process for task allocation in a decentralised manner. Then, according to the allocation result, the agents are disassembled into subgroups and the subgroups proceed to their assigned tasks. After the agents succeed in jamming the radars, the allies, which are armed with Air-to-Ground Missiles (AGMs) and loitering until that moment, directly fly to and attack the radars. Note that the agents are assumed to have small RCSs due to their small sizes, thus not being tracked by the radars.

The cooperative jamming effectiveness by a set of n agents toward the j-th radar is a function of the *Jamming to Signal ratio* as follows (Kim and Hespanha (2004)):

$$(J/S)_j = \frac{\sum_{i=1}^{n} J_{i,j}}{S_j}. \tag{1}$$

Here, S_j is the backscattered signal strength from one ally, which has the largest RCS than others, to the j-th radar:

$$S_j = k_j^R \cdot \sigma(\theta_{a,j})/d_{a,j}^4 \tag{2}$$

where k_j^R is the radar-dependent coefficient, which varies with the radar's characteristics such as its transmitted power, wavelength, and antenna gains; $d_{a,j}$ is the distance between the ally and the j-th radar; and σ is the RCS of the ally, which depends on its attitude with respect to the j-th radar, denoted by $\theta_{a,j}$.

The individual jamming signal strength of the i-th agent toward the j-th radar is defined by

$$J_{i,j} = k_i^A \cdot G_{i,j}^R / d_{i,j}^2 \quad (3)$$

where k_i^A is the jamming-agent-dependent coefficient; $d_{i,j}$ is the distance between the i-th agent and the j-th radar when the agent performs jamming; and $G_{i,j}^R$ is the radar antenna gain, which relies on whether or not the position of the i-th jamming agent is aligned with the j-th radar's main-lobe.

The cooperative jamming for the j-th radar is effective if $(J/S)_j$ exceeds a certain level of *burn-through value* $(J/S)_j^{burn}$. Note that $(J/S)_j^{burn}$ is determined by the radar's characteristics and signal processing method.

4. DECENTRALISED TASK ALLOCATION

4.1 A Game-theoretical Framework

In the cooperative jamming mission described in Section 3, one of the issues is how to form proper subgroups and assign the subgroups to given tasks, while optimising an objective value. This is one of the problems called *multi-robot task allocation*.

A framework to deal with this problem should be decentralised and scalable, particularly with consideration of the large cardinality of a multiple agent system. Furthermore, an agreed solution obtained by the framework is desirable to provide a certain level of guaranteed global optimality. However, there is in fact a trade-off between scalability and optimality because the problem is $\mathcal{NP}$-hard (Gerkey and Matarić (2004)).

In our previous work (Jang et al. (2016)), a decentralised game-theoretical framework called GRAPE was proposed. This framework is based on anonymous hedonic games (Ballester (2004)). A hedonic game models a conflict situation between selfish agents who want to form their preferred coalitions according to their preferences with regard to other agents. With anonymous preferences, every agent is concerned with not the identities but instead the total number of agents in the same coalition. The objective of this framework is to find a *Nash stable partition*, from which every agent does not deviate (i.e., an agreed task assignment). The framework guarantees the existence of an Nash stable partition, and provides some advantages regarding scalability and global optimality under the following condition: the preference of every agent with respect to every task is based on a monotonically-decreasing function in terms of the number of members in the same coalition. Note that, in the previous work, this condition is called SPAO (single-peaked-at-one), and this term will be used in the rest of this paper.

The advantages given by the framework are described as the following theorems; please refer to (Jang et al. (2016)) for their proofs and more details. Before that, let us briefly explain the key definitions and notations that are used in the theorems as well as in the rest of this paper: (1) An instance of GRAPE is a tuple $(\mathcal{A}, \mathcal{T}, \mathcal{P})$, where $\mathcal{A} = \{\mathsf{a}_1, ..., \mathsf{a}_\mathsf{n}\}$ is a set of agents, $\mathcal{T} = \{\mathsf{t}_0, \mathsf{t}_1, ..., \mathsf{t}_\mathsf{m}\}$ is a set of tasks (t_0 means "not work any task"), and $\mathcal{P} = (\mathcal{P}_1, ..., \mathcal{P}_\mathsf{n})$ is n-tuple of the preferences of the agents. $\mathcal{P}_i$ indicates the preference of agent a_i over every task-coalition pairs $(\mathsf{t}_j, \mathsf{p}) \in \mathcal{T} \times \{1, 2, ..., \mathsf{n}\}$, which can be interpreted as "to execute task t_j with p members"; (2) $\Pi = \{\mathsf{S}_0, \mathsf{S}_1, ..., \mathsf{S}_\mathsf{m}\}$ is a (disjoint) partition such that $\cup_{j=0}^{\mathsf{m}} \mathsf{S}_j = \mathcal{A}$ and $\mathsf{S}_j \cap \mathsf{S}_k = \emptyset$ for $j \neq k$, where S_j is the coalition dedicated to execute task t_j; (3) Given a partition Π, $\Pi(i)$ is the index of the task assigned to agent a_i, for example, $\mathsf{S}_{\Pi(i)}$ is the coalition that agent a_i belongs to; (4) An instance of GRAPE can be said to be SPAO if the preference of every agent with respect to every task is SPAO.

Theorem 1. (Existence). If $(\mathcal{A}, \mathcal{T}, \mathcal{P})$ is an instance of GRAPE that is SPAO, then a Nash stable partition always exists.

Agents in an instance of GRAPE that is SPAO can converge to a Nash stable partition by utilising *merge-and-split algorithm* (Apt and Witzel (2009)) in a decentralised manner. Let *round* represent an iterative step where an arbitrary agent evaluates all the available task-coalition pairs given the current partition, and then determines which coalition to join. Note that every agent is assumed to be able to inform its status to all the other agents at every round.

Theorem 2. (Convergence). If $(\mathcal{A}, \mathcal{T}, \mathcal{P})$ is an instance of GRAPE that is SPAO, then the number of rounds to determine a Nash stable partition is at most $|\mathcal{A}|(|\mathcal{A}|+1)/2$.

Let $\mathcal{U}_i(\mathsf{t}_j, \mathsf{p})$ denote the utility of agent a_i executing task t_j with $\mathsf{p} \in \{1, 2, ..., |\mathcal{A}|\}$ of other team members including herself. The objective of a task allocation problem is to find a partition Π that maximises the aggregated utility (global utility) of all given agents, as follows:

$$\mathcal{J} = \sum_{\forall \mathsf{a}_i \in \mathcal{A}} \mathcal{U}_i(\mathsf{t}_{\Pi(i)}, |\mathsf{S}_{\Pi(i)}|) \quad (4)$$

In fact, a Nash stable partition obtained by this framework is not necessarily the optimal result. Instead, the Nash stable partition's optimality (the ratio between the objective values obtained by the Nash stable partition and by the optimal partition) can be guaranteed by the following theorem.

Theorem 3. (Global Optimality). The global optimality provided by a Nash stable partition Π in an instance of GRAPE, denoted by $\mathcal{M}_{OPT} := \mathcal{J}_{GRAPE}/\mathcal{J}_{OPT}$, is bounded as:

$$\mathcal{M}_{OPT} \geq \mathcal{J}_{GRAPE}/(\mathcal{J}_{GRAPE} + \lambda) \quad (5)$$

where

$$\lambda := \sum_{\forall \mathsf{S}_j \in \Pi} \max_{\mathsf{a}_i \in \mathcal{A}, \mathsf{p} \leq |\mathcal{A}|} \{\mathsf{p} \cdot [\mathcal{U}_i(\mathsf{t}_j, \mathsf{p}) - \mathcal{U}_i(\mathsf{t}_j, |\mathsf{S}_j \cup \{\mathsf{a}_i\}|)]\}$$

4.2 Implementation to Cooperative Radar Jamming

Consider that the agents for the cooperative jamming mission in Section 3 are desired to be distributed in proportion to the required workloads of given tasks, while reducing unnecessary costs (e.g., traveling costs). Note that we assume that the agents are homogeneous and have the same jamming resources, thus they consider only the number of other agents during a task allocation process.

In order to accommodate this desire, the objective function of this problem is (4), and the utility of agent a_i is defined as:

$$\mathcal{U}_i(\mathsf{t}_j, \mathsf{p}) = \frac{\mathcal{R}(i,j,\mathsf{p})^{1-\alpha}}{\mathcal{C}(i,j,\mathsf{p})^{\alpha}}. \tag{6}$$

$\mathcal{R}(i,j,\mathsf{p})$ is agent a_i's reward, which is set to be the same as the agent's contribution to task t_j when the required workload is equally shared with p members. The reward is bounded by the agent's maximum possible contribution (i.e., maximum jamming capacity).

$$\mathcal{R}(i,j,\mathsf{p}) = \begin{cases} J_{i,j} & \text{if } S_j \cdot (J/S)_j^{burn}/\mathsf{p} > J_{i,j} \\ S_j \cdot (J/S)_j^{burn}/\mathsf{p} & \text{otherwise.} \end{cases} \tag{7}$$

$\mathcal{C}(i,j,\mathsf{p})$ is a cost function basically based on the distance from agent a_i to task t_j. The cost increases as the number of members in a coalition, i.e., p, increases, which reflects that the more member may induce the higher maintenance costs such as communications between them.

$$\mathcal{C}(i,j,\mathsf{p}) = \beta \cdot d_{i,j} \cdot \mathsf{p}^{\gamma} \tag{8}$$

where α, β, and γ are weight factors.

In a nutshell, the utility function in (6) is monotonically decreasing in terms of the number of members. When this problem is modelled as an instance of GRAPE, every agent's preference holds SPAO. Hence, one can benefit from the theorems in Section 4.1 with regard to scalability (i.e., polynomial-time convergence) and guaranteed global optimality.

5. CONTROL SYSTEM

The proposed flight array system is differently controlled depending on which flight mode that the system uses, i.e., the individual-flight mode and flight-array mode. For the former, a controller based on a full-state feedback LQR (linear quadratic regulator) and one based on a PID (proportional-integral-derivative) are respectively utilised to control the velocity and position of a single UAV. For the latter, a SMC (sliding mode control) method is adopted for the attitude controller of the array system, dealing with the system's nonlinearity caused by its complicated configuration. PID control techniques are also used for the velocity and position control of the array system. The following subsections particularly describe the LQR and SMC control methods.

5.1 Control of the Individual Vehicle

The dynamic model of a single vehicle can be derived from the forces and moments based on a body-fixed coordinate system (Jeong et al. (2016)). The individual vehicle has the geometric and inertial properties presented in Table 1. Note that the cross-terms of the moment of inertia are neglected since they are too small to affect the dynamics. The dynamic model is linearised at a hovering flight condition as follows:

Table 1. Specifications of a single vehicle

Properties	Parameter	Value
Mass	m	4.8 (kg)
Height	h	0.8 (m)
Radius of duct	r_{duct}	0.3 (m)
Chord of duct	c_{duct}	0.18 (m)
Moment of inertia	J_{xx}	0.9070 $(kg \cdot m^2)$
	J_{yy}	0.9070 $(kg \cdot m^2)$
	J_{zz}	0.1395 $(kg \cdot m^2)$

$$\dot{\underline{x}}_1 = A\underline{x}_1 + B\underline{u}_1 \tag{9}$$

where

$$\underline{x}_1 = [u, v, w, p, q, r, \phi, \theta, \psi]^T,$$
$$\underline{u}_1 = [\delta_{RPM}, \delta_{ail}, \delta_{ele}, \delta_{rud}]^T,$$
$$A = \begin{bmatrix} -2.08 & 0 & 0 & 0 & 0 & 0 & 0 & -9.81 & 0 \\ 0 & -2.08 & 0 & 0 & 0 & 0 & 9.81 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.005 & -0.005 & 0 \\ 0 & 0.93 & 0 & 0 & -0.009 & 0 & 0 & 0 & 0 \\ -0.93 & 0 & 0 & 0.009 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.096 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix},$$
$$B = \begin{bmatrix} 0 & 0 & 5.69 & 0 \\ 0 & -5.69 & 0 & 0 \\ -0.0052 & 0 & 0 & 0 \\ 0 & 10.74 & 0 & 0 \\ 0 & 0 & 10.74 & 0 \\ -0.002 & 0 & 0 & 70.49 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

The state vector ($\underline{x}_1$) consists of velocities (u, v, w), angular rates (p, q, r), and Euler angles (ϕ, θ, ψ) on each body axis. The input vector ($\underline{u}_1$) is composed of the difference of RPM (δ_{RPM}) and the differences of deflections of each flap combination ($\delta_{ail}, \delta_{ele}, \delta_{rud}$) from the trim values at the hovering flight condition. Note that δ_{ail}, δ_{ele}, and δ_{rud} represent control flap combinations for generating rolling, pitching, and yawing moments, respectively (see Section 2).

In order to design a LQR controller for the velocity of the vehicle, the dynamic model in (9) is rewritten with augmented states, as follows:

$$\dot{\underline{x}}_{aug} = A_{aug}\underline{x}_{aug} + B_{aug}\underline{u}_1 \tag{10}$$

where

$$A_{aug} = \begin{bmatrix} A & \mathbf{0}_{9\times4} \\ A_{add} & \mathbf{0}_{4\times4} \end{bmatrix}, B_{aug} = \begin{bmatrix} B \\ \mathbf{0}_{4\times4} \end{bmatrix},$$
$$A_{add} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Noting that the state vector ($\underline{x}_1$) is divided into two groups, i.e., tracking ($\underline{x}_t = [u, v, w, \psi]^T$) and regulating states ($\underline{x}_r = [p, q, r, \phi, \theta]^T$), we define the augmented state vector as

$$\underline{x}_{aug} = \left[\underline{x}_t^T(1:3), \underline{x}_r^T, \underline{x}_t(4), \int \underline{x}_t{}^T\right]^T. \tag{11}$$

The LQR controller is designed by using the following quadratic performance index:

$$J = \frac{1}{2}\int_0^{\infty} \left\{\underline{x}_{aug}^T Q \underline{x}_{aug} + \underline{u}_1^T R \underline{u}_1)\right\} dt \tag{12}$$

where the weighted matrices $Q \in \mathbb{R}_+^{13\times13}$ and $R \in \mathbb{R}_+^{4\times4}$ are diagonal, and heuristically chosen such that:

$$\begin{aligned} \text{diag}\{Q\} =& \{(1/0.6)^2, (1/0.6)^2, (1/0.01)^2, 1, 1, 1, 1, 1, \\ & (1/0.1)^2, (1/0.8)^2, (1/0.8)^2, (1/0.008)^2, (1/0.4)^2\}, \\ \text{diag}\{R\} =& \{(1/10)^2, (1/0.1)^2, (1/0.1)^2, (1/0.1)^2\}. \end{aligned} \tag{13}$$

The LQR control law is designed as

$$\underline{u}_1 = -K_1\left(\underline{x}_{aug} - \underline{x}_{aug,r}\right) \tag{14}$$

where

$$\underline{x}_{aug,r} = [\underline{x}_{t,r}^T(1:3), \mathbf{0}_{1\times5}, \underline{x}_{t,r}(4), \int \underline{x}_{t,r}^T]^T$$

$$K_1 = [K_t(:,1:3), K_r, K_t(:,4), K_i].$$

Note that $\underline{x}_{t,r}$ is the reference for $\underline{x}_t$. The control gain matrix ($K_1 \in \mathbb{R}^{4\times13}$) is calculated by the Matlab function *care*. Figure 3 shows the block diagram of the LQR controller.

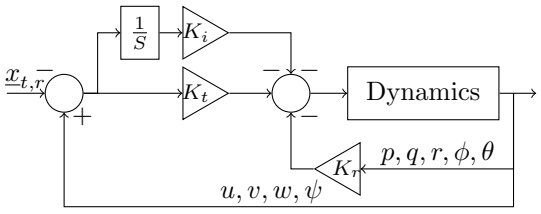

Fig. 3. The velocity controller of the individual vehicle

5.2 Control of the Flight Array

The dynamic model of a flight array includes multiple single-vehicle dynamics and its geometric effects, which vary depending on the number of individual vehicles being assembled and the resulted system configuration. The coordinate system of the array is defined as Figure 4 (Jeong et al. (2016)), where the origin is the centre of gravity of the assembled configuration. Noting that the mass change of each vehicle during a mission can be negligible, we assume that the vehicle can recognise the entire configuration and obtain its centre of gravity and moment of inertia.

For the attitude controller of the flight array system, a SMC method is utilised for the system's angular rates (p, q, r) since these states exhibit fast dynamics with much complex nonlinear characteristics. The system's Euler angles (ϕ, θ, ψ) are controlled by a PID method. The block diagram of the attitude control system is described in Figure 5.

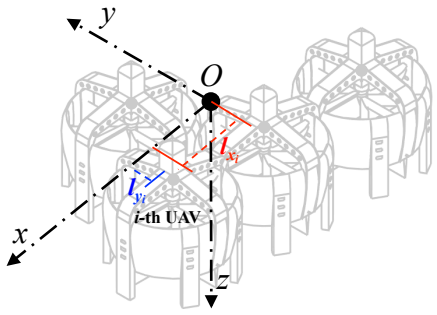

Fig. 4. Coordinate system of the flight array

To apply the nonlinear control method, an affine system of the fast dynamics is derived as follows:

$$\dot{\underline{x}}_2 = f(\underline{x}) + g(\underline{x})\underline{u}_2 \tag{15}$$

where the state vector $(\underline{x}_2)$ is $[p, q, r]^T$, and $\underline{x}$ consists of velocities (u, v, w), angular rates (p, q, r), and attitudes

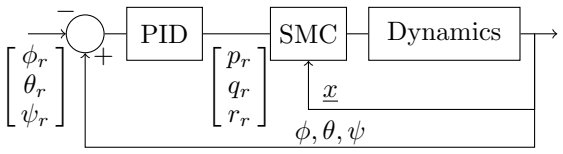

Fig. 5. The attitude controller of the flight array

(ϕ, θ, ψ) of the flight array. The input vector ($\underline{u}_2 \in \mathbb{R}^{4N\times1}$) is $[\delta_1, ..., \delta_i, ..., \delta_N]^T$, where N is the number of the vehicles being assembled for the flight array, and $\delta_i = [\delta_{RPM_i}, \delta_{ail_i}, \delta_{ele_i}, \delta_{rud_i}]$ is the vector of control surface deflections of the i-th vehicle. Each term of the dynamics is arranged as follows:

$$f(\underline{x}) = \begin{bmatrix} \{qr\left(J_{o,yy} - J_{o,zz}\right) + L_a\}/J_{o,xx} \\ \{pr\left(J_{o,zz} - J_{o,xx}\right) + M_a\}/J_{o,yy} \\ \{pq\left(J_{o,xx} - J_{o,yy}\right) + N_a\}/J_{o,zz} \end{bmatrix} \tag{16}$$

where $L_a = \sum_{i=1}^{N}\left(M_{fuse,x_i} + M_{duct,x_i} + M_{gyro,x_i} + l_{y_i}Z_{a_i}\right)$,

$M_a = \sum_{i=1}^{N}\left(M_{fuse,y_i} + M_{duct,y_i} + M_{gyro,y_i} - l_{x_i}Z_{a_i}\right)$,

$N_a = \sum_{i=1}^{N}\left(-l_{y_i}X_{a_i} + l_{x_i}Y_{a_i}\right)$,

$X_{a_i} = F_{fuse,x_i} + F_{duct,x_i} + F_{grav,x_i}$,

$Y_{a_i} = F_{fuse,y_i} + F_{duct,y_i} + F_{grav,y_i}$,

$Z_{a_i} = F_{fuse,z_i} + F_{duct,z_i} + F_{grav,z_i}$; and

$$g\left(\underline{x}\right)\underline{u}_2 = [L_b/J_{o,xx}, M_b/J_{o,yy}, N_b/J_{o,zz}]^T \tag{17}$$

where

$L_b = \sum_{i=1}^{N}\left(M_{flap,x_i} + l_{y_i}F_{rotor,z_i}\right)$,

$M_b = \sum_{i=1}^{N}\left(M_{flap,y_i} - l_{x_i}F_{rotor,z_i}\right)$,

$N_b = \sum_{i=1}^{N}\left(M_{flap,z_i} + M_{rotor,z_i} - l_{y_i}F_{flap,x_i} + l_{x_i}F_{flap,y_i}\right)$.

From (17), $g(\underline{x}) \in \mathbb{R}^{3\times4N}$ is derived as follows:

$$g(\underline{x}) = \begin{bmatrix} g(1,1) & g(1,2) & 0 & 0 & \cdots \\ g(2,1) & 0 & g(2,3) & 0 & \cdots \\ g(3,1) & g(3,2) & g(3,3) & g(3,4) & \cdots \end{bmatrix} \tag{18}$$

where

$g(1, 4(i-1)+1) = l_{y_i}c_{thr}/J_{o,xx}$,

$g(1, 4(i-1)+2) = \mathrm{sgn}(v_{ind} - w)q_f C_{L_{\delta f}} S_{ail} l_{ail}/J_{o,xx}$,

$g(2, 4(i-1)+1) = -l_{x_i}c_{thr}/J_{o,yy}$,

$g(2, 4(i-1)+3) = \mathrm{sgn}(v_{ind} - w)q_f C_{L_{\delta f}} S_{ele} l_{ele}/J_{o,yy}$,

$g(3, 4(i-1)+1) = c_{tor}/J_{o,zz}$,

$g(3, 4(i-1)+2) = -l_{x_i}\mathrm{sgn}(v_{ind} - w)q_f C_{L_{\delta f}} S_{ail}/J_{o,zz}$,

$g(3, 4(i-1)+3) = -l_{y_i}\mathrm{sgn}(v_{ind} - w)q_f C_{L_{\delta f}} S_{ele}/J_{o,zz}$,

$g(3, 4(i-1)+4) = \mathrm{sgn}(v_{ind} - w)q_f C_{L_{\delta f}} S_{rud} l_{rud}/J_{o,zz}$.

c_{thr} and c_{tor} are constant coefficients of the thrust and torque, respectively, for a trim condition when the flight array system hovers. $S_{\{\cdot\}}$ and $l_{\{\cdot\}}$ indicate the area and moment arm of each flap combination. $C_{L_{\delta f}}$ is the lift coefficient of the flap, and v_{ind} is the induced velocity through the rotor. And, q_f represents the dynamic pressure at the aerodynamic centre of the flap.

For the SMC, a sliding surface is defined with an integral term as

$$S = e + K_2\int_0^t e\,dt \tag{19}$$

where $e = \underline{x}_2 - \underline{x}_{2,r}$ is the error vector, i.e., the difference between the current state $\underline{x}_2$ and its reference $\underline{x}_{2,r}$; and K_2 indicates a control gain matrix.

In order to deign a control law that satisfies the Lyapunov stability, a typical Lyapunov candidate function is used, i.e., $V = 1/2 \cdot S^T S$. The time derivative of the candidate function is

$$\begin{aligned} \dot{V} &= S^T \dot{S} \\ &= S^T (f + g\underline{u}_2 - \dot{\underline{x}}_{2,r} + K_2 e). \end{aligned} \tag{20}$$

From this, a control law of the SMC can be designed as

$$\underline{u}_2 = -g^{-1} \left[f - \dot{\underline{x}}_{2,r} + K_2 e + c_1 S + c_2 \operatorname{sgn}(S) \right] \tag{21}$$

where c_1 is positive definite diagonal matrix, and c_2 is a positive-semidefinite diagonal matrix. Substituting the control law into (20) yields

$$\begin{aligned} \dot{V} &= S^T (-c_1 S - c_2 \operatorname{sgn}(S)) \\ &= -S^T c_1 S - c_2 \|S\| \leq 0. \end{aligned} \tag{22}$$

The sign function in (22) is replaced with a saturation function, by which a chattering problem can be avoided. Since the g matrix is not square, the weighted pseudo-inverse g^+ is adopted to calculate the control inputs. Thus, the final control law of the SMC is

$$\underline{u}_2 = -g^+ \left[f - \dot{\underline{x}}_{2,r} + K_2 e + c_1 S + c_2 \operatorname{sat}(S) \right] \tag{23}$$

where

$$g^+ = W^{-1} g^T \left(g W^{-1} g^T \right)^{-1}$$

and W represents a weighted matrix.

6. NUMERICAL SIMULATION RESULTS

6.1 Used Parameters and Conditions

This section presents simulation results obtained by our prototype simulation program, which was implemented for a cooperative stand-in jamming mission, as a proof-of-concept for the proposed approach. The realistic values shown in Table 2 (Kim and Hespanha (2004)) were used, which are originally from (DARPA/IXO Program "Mixed Initiative Control for Automa-teams" (2003)). Note that k^A of micro-sized UAV is assumed from consideration of other UAV types' values.

Consider that there are a set of jamming micro UAVs, two allies, and two adversary radars. $d_{a,j}$ is selected based on the distance between one ally's expected position to attack the j-th radar and the position of the radar, after the success of the cooperative jamming. We set that the ally moves straight toward and attacks the radar 20 km ahead. As the ally has been heading to the radar, $\theta_{a,j}$ is assumed to be fixed, and thus its RCS is also fixed. Particularly, each ally is considered as F-16, the RCS of which is known as 5 m^2. We assume that k_i^A is the same for all the UAVs due to their homogeneity. Note that weight factors α, β and γ are set as 0.25, 1, and 0.5, respectively.

Table 2. Parameters of Jamming UAVs and Adversary Radars (Kim and Hespanha (2004))

UAV size	k^A	Radar type	k^R
micro	0.005	short-range	$2 \cdot 10^7$
small	0.25	med.-range	$2 \cdot 10^8$
large	1	long-range	$2 \cdot 10^9$
$(J/S)^{burn}$	G^R main-lobe	G^R side-lobe	
1	1	0.001	
1	1	0.001	
1	1	0.001	

6.2 Results: Task Allocation

A Monte Carlo simulation was conducted with 100 runs to validate the scalability and the guaranteed global optimality of our proposed framework with regard to the task allocation problem. Consider that 400 of micro UAVs and two long-range adversary radars are given, and the UAVs are supposed to neutralise their assigned radars at least in the side lobe of and within a range of 100 m from the radars. At each run, the positions of radars and agents are randomly located, and the agents initially gather before starting the task allocation process. Scalability was evaluated by the number of required rounds for the agents to converge into an agreed solution.

The upper figure in Figure 6 presents that the number of the required rounds is averagely 510 and at most 564, which confirms Theorem 2. Furthermore, it has been empirically observed through additional experiments that, in general, the number of required rounds is less than two times of the number of given agents. On the other hand, the lower figure shows the guaranteed global optimality calculated by Theorem 3. Averagely, the solutions provided by our framework guarantee 75% of the global optimality.

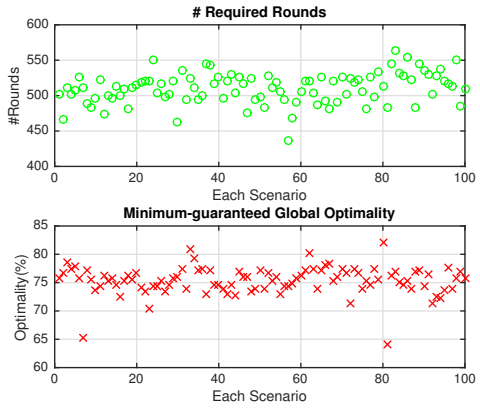

Fig. 6. Scalability (upper) and guaranteed global optimality (lower) at each scenario

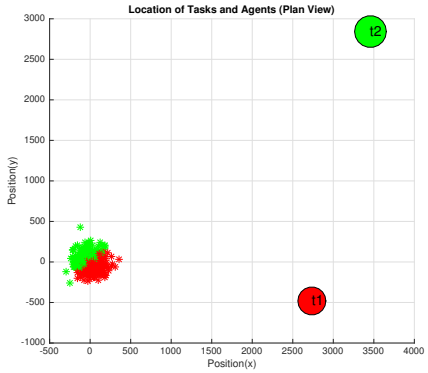

Fig. 7. An example of the visualised task allocation results

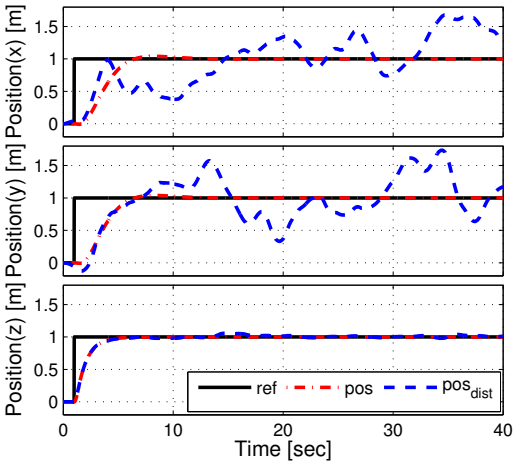

Fig. 8. Position tracking error of the single UAV

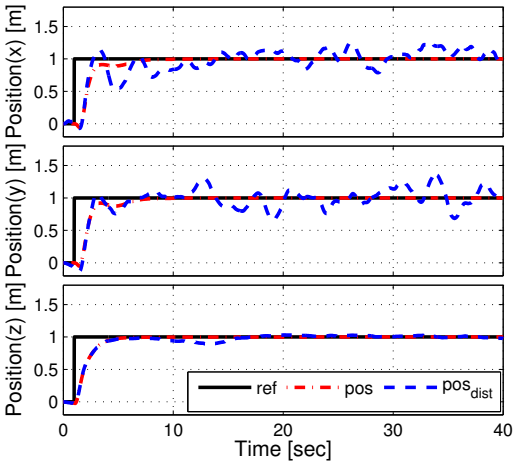

Fig. 9. Position tracking error of the flight array

Figure 7 portrays the visualised task allocation result at one instance of the Monte Carlo simulation conducted. The asterisks and the circles indicate the positions of agents and tasks, respectively. The coloured agents are assigned to the same coloured task. This shows that agents are partitioned into two subgroups based on their current positions with respect to the positions of given tasks, because the distances between an agent and tasks were considered as a cost in the utility function of the agent

6.3 Results: Control

This subsection shows numerical simulations to investigate the performance of the control systems derived in Section 5 under wind disturbance. The wind disturbance, generated by the Dryden wind turbulence model, was set to be blown randomly along the x-, y-, and z-axes, and its maximum strength was approximately at most 2 m/s for each axis.

Figure 8 and 9 present position tracking results of the single UAV and those of the flight array system consisting of 10 UAVs, respectively. Note that the configuration of the flight array is hexagonal as shown in Figure 10(a). The black bold line represents a position reference. The blue dashed line shows the tracking results with the wind disturbance, and the red dash-dot line indicates the results without the disturbance.

The simulation results show that the both designed control systems, the controller for a single vehicle and one for the array, exhibit proper performance under windless condition. However, under the wind disturbance, the single UAV has higher root-mean-square(RMS) errors (about 0.40 m) in comparison with the flight array (about 0.20 m) on x- and y- axes, because the UAV has not enough control power to overcome the disturbance.

6.4 Demonstration of Our Prototype Integrated Simulation

Our prototype simulation program for a cooperative jamming mission provides the feasibility of the proposed flight array system and coordination approach. Due to the lack of the computational resources for visualisation, the number of agents was set to be only 10. Since it is not practically possible for such a small number of micro UAVs to neutralise a radar, we suppose that each agent has a higher jamming coefficient, i.e., $k_i^A = 0.05$, and the adversary radars are short-range-type. Note that one radar was set to require more jamming resource than the other one, i.e., $(k_1^R = 3.33 \cdot 10^7,\ k_2^R = 2 \cdot 10^7)$, and the agents are assumed to jam the radars 500 m ahead of the radars. In this experiment, a steady wind (1.5 m/s), which is strong enough to make a single UAV drift despite its control power, was set to be blown along the x-axis from $(-)$ to $(+)$.

Figure 10(a) to 10(d) present four key scenes during the mission in chronological order. Two adversary radars are represented by the red (task 1) and blue (task 2) cylinders, and the size of each cylinder indicates the required jamming resources for the associated radar to be jammed. Initially, the agents are deployed and fly as a flight array, and then proceed to the enemy territory up to a certain position (Figure 10(a) and 10(b)). The position is where the flight array will be disassembled, and was set to be $(850, 1700)$ m because each agent, when flying individually, was expected to drift by the wind. At that position, they assign themselves to proper adversary radars (Figure 10(c)), and individually continue to move forward to the assigned radars (Figure 10(d)). Note that there must be the time required for the task allocation process, but it is not counted in the time frame of this visualisation result.

The task allocation result is sensible because more agents are assigned to the higher demanded task, while reducing the costs (i.e., the distances to the radars). The global optimality of the result obtained is guaranteed by 78.9% from Theorem 3, and in fact exhibits 99.6% compared with the optimal result from a brute-force search. In addition, the number of rounds required for the agents to converge the result is just 10.

Figure 11 presents the position tracking results of the flight array system from the time associated with Figure 10.(a) to one with 10.(b). It is shown that the array properly reached to the disassembling position despite the wind disturbance. Figure 12 shows the position tracking error of one of the UAVs (i.e., the first one, which is marked in Figure 10.(c)) after being disassembled. Although the UAV attempted to fly to and stay around its assigned target, it is presented that its tracking error with regard to x-axis

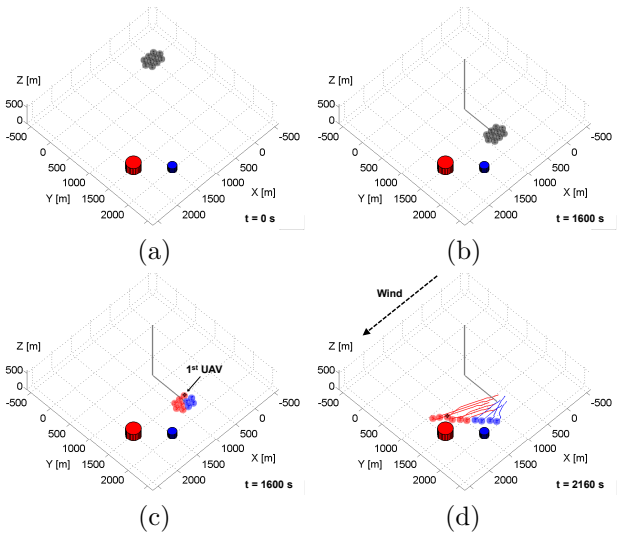

Fig. 10. Visualisation result obtained by our prototype simulation program under wind disturbances

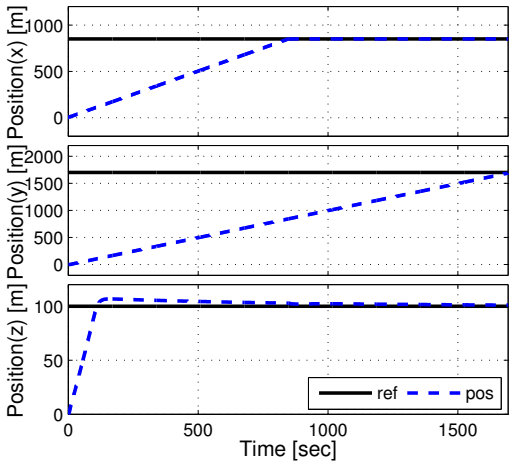

Fig. 11. Position tracking error of the flight array in the integrated simulation

steadily increased due to the strong wind. Thus, the UAVs should be landed on proper locations to accomplish the jamming mission.

7. CONCLUSIONS AND FUTURE WORKS

This paper presented a new flight array concept of micro UAVs and our approaches to coordinate them in terms of mission planning level as well as control level. As a possible application, we suggested a cooperative stand-in jamming mission and implemented our proposed approach into a prototype simulation program as a proof-of-concept.

In the future, we plan to extend our work to include heterogeneity of agents. Although agents are manufactured homogeneously, they may have different resources after they finish flying as a flight array. For the task allocation, it is necessary to additionally consider individual agents' remaining resources, unless the resource for flying and one for jamming are mutually independent. Furthermore, the control allocation problem will be further studied with consideration of workload balance amongst individual agents of a flight array system.

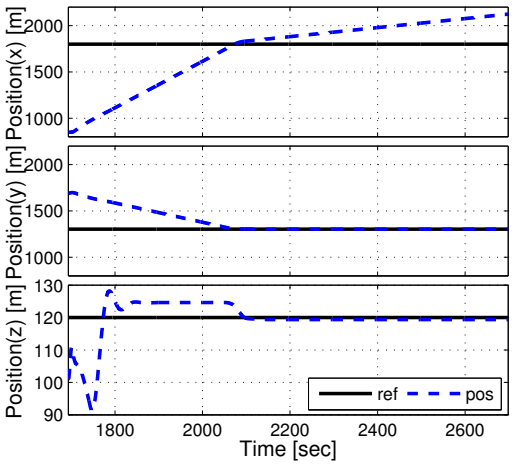

Fig. 12. Position tracking error of the first UAV after being disassembled in the integrated simulation

REFERENCES

Apt, K.R. and Witzel, A. (2009). A Generic Approach to Coalition Formation. *International Game Theory Review*, 11(03), 347–367.

Ballester, C. (2004). NP-completeness in Hedonic Games. *Games and Economic Behavior*, 49(1), 1–30.

DARPA/IXO Program "Mixed Initiative Control for Automa-teams" (2003). Challenge Problem Version 1.3, MICA OEP Software Version 1.3.

Gerkey, B.P. and Matarić, M.J. (2004). A formal analysis and taxonomy of task allocation in multi-robot systems. *International Journal of Robotics Research*, 23(9), 939–954.

Habib, M., Quimby, P.W., Chang, S., Jackson, K., and Cummings, M.L. (2011). Wind gust alerting for supervisory control of a micro aerial vehicle. *IEEE Aerospace Conference Proceedings*, 1–7.

Jang, I., Shin, H.S., and Tsourdos, A. (2016). Anonymous hedonic game for task allocation in a large-scale multiple agent system. *IEEE Transactions on Robotics (Submitted)*.

Jeong, J., Kim, S., and Suk, J. (2015). Control system design for a ducted-fan unmanned aerial vehicle using linear quadratic tracker. *International Journal of Aerospace Engineering*.

Jeong, J., Kim, S., and Suk, J. (2016). Adaptive sliding mode controller design for reconfiguration of organic flight array. In *30th Congress of the International Council of the Aeronautical Sciences*. Daejeon, Republic of Korea.

Kim, J. and Hespanha, P. (2004). Cooperative radar jamming for groups of unmanned air vehicles. In *IEEE Conference on Decision and Control*, 632–637. Atlantis, Bahamas.

Oung, R. and D'Andrea, R. (2011). The distributed flight array. *Mechatronics*, 21(6), 908–917.

Sahin, E. (2005). Swarm robotics: from sources of inspiration to domains of application. In *Swarm Robotics*, 10–20. Springer, Berlin.

Samad, T., Bay, J.S., and Godbole, D. (2007). Network-centric systems for military operations in urban terrain: The role of UAVs. *Proceedings of the IEEE*, 95(1), 92–107.

2017-10-18

Cooperative control for a flight array of UAVs and an application in radar jamming

Jang, Inmo

Elsevier

Jang I, Jeong J, Shin H-S, Kim S, Tsourdos A, Suk J, Cooperative control for a flight array of UAVs and an application in radar jamming, IFAC papers online, Vol. 50, Issue 1, July 2017, pp. 8011-8018
http://dx.doi.org/10.1016/j.ifacol.2017.08.1225