2014 International Conference on Unmanned Aircraft Systems (ICUAS)
May 27-30, 2014. Orlando, FL, USA

The block-sharing strategy for area monitoring missions using a decentralized multi-UAV system*

L.E. Caraballo J.J. Acevedo J.M. Díaz-Báñez B.C. Arrue I. Maza A. Ollero

Abstract— **In monitoring missions, using a cooperative team of Unmanned Aerial Vehicles (UAVs), the goal is to minimize the elapsed time between two consecutive observations of any point in the area. Techniques based in area partitioning achieve the goal when the sub-area assigned to each UAV is according to its capabilities.**

In previous work of the authors [1] it was presented the *one-to-one* strategy to obtain in a decentralized way a near optimal partition from any initial grid partition. In this paper a generalization of that strategy called the *block-sharing* technique is presented. The goal in this work is to accelerate the convergence to an optimal partition with respect to previous work.

I. INTRODUCTION AND RELATED WORK

The European Project EC-SAFEMOBIL[1] is devoted to the development of sufficiently accurate common motion estimation and control methods and technologies in order to reach levels of reliability and safety to facilitate unmanned vehicle deployment in a broad range of applications. One of the applications of the project includes the distributed safe reliable cooperation and coordination of many high mobility entities. The aim is to precisely control hundreds of entities efficiently and reliably and to certify developed techniques to support the exploitation of unmanned platforms in non-restricted areas. Two scenarios have been chosen for the validation of the developments: industrial warehousing involving a large number of autonomous vehicles and surveillance also involving many mobile entities. This paper is focused on the latter scenario.

Surveillance and monitoring missions with unmanned aerial vehicles (UAVs) have been widely studied in different contexts [2]: military applications, automated inspection, search and rescue missions, etc. The application of distributed multi-UAV systems allows to accomplish them with robustness against failures and dynamism, in a way that each UAV can quickly self-adapt its sub-area, higher spatial coverage and an efficient deployment [3], [4], [5]. Therefore, the system can perform the mission, even if the communication link with the control station is broken. This paper addresses, in a distributed manner, a surveillance scenario using a large-scale team of aerial robots.

Surveillance tasks are solved from frequency-based approach, where the objective implies to optimize the refresh time or elapsed time between two consecutive visits to any position. The objectives are to minimize the maximal frequency as in [6] or to guarantee an uniform frequency of visits as in [7] and the solution is deterministic. Other authors, as in [8], address the patrolling problem in adversarial settings applying a probabilistic approach and considering intelligent intruders that could learn the strategy. In [9], different partitioning and cyclic patrolling strategies are defined and compared from a frequency-based approach. Authors of [10] also analyze the *refresh time* in area coverage problems with multiple robots using different approaches. A partitioning method is proposed in [11] to monitor a set of positions with different priorities.

Regarding distributed coordination, in [12] and [13] methods based on the *coordination variables* are proposed to perform a perimeter monitoring mission in a cooperative manner with multiple UAVs. However, the selection of these variables can be difficult for complex problems as the area monitoring. Then in [4], a rectangular area is monitored with a team of UAVs applying the *one-to-one coordination*. This technique implies that each pair of UAVs solves a coordination problem including only their own information. A similar technique is proposed in [14] to coordinate a team of video-cameras in surveillance missions. These techniques allows the system to obtain the whole coordination from local decisions and information. The resulting system is scalable, because each UAV only needs information from nearby neighbors.

In previous work of the authors [1], one-to-one coordination allowed convergence in a distributed and decentralized manner to an area partition where each UAV has an area to monitor according to its capabilities. However, in this paper it is shown how to accelerate the convergence in more general scenarios where the previous one-to-one strategy is time-consuming and slow. A new distributed method called *block sharing* is described and proposed as a generalization of the previous scheme. In the previous algorithm, the UAVs performed an area division every time they communicate with a neighbor. However, with the proposed block-sharing strategy, the UAVs wait until they get information from a set of UAVs before performing the area division. This delay in the area partitioning allows to execute fewer operations of sub-area allocation, reconstruction of boundaries and synchronization maintenance.

*This work has been developed in the framework of the the EC-SAFEMOBIL (FP7-ICT-2011-288082), the PLANET (INFSO-ICT-257649) and the CLEAR (DPI2011-28937-C02-01) projects.

Luis Evaristo Caraballo is with Departamento de Computación, Universidad de La Habana, Cuba. E-mail: luis.caraballo@iris.uh.cu

Jose Miguel Diaz-Bañez is with Dpto. de Matemática Aplicada II, Universidad de Sevilla, Spain. E-mail: dbanez@us.es

Jose Joaquin Acevedo, Begoña Arrue, Ivan Maza and Anibal Ollero are with Grupo de Robótica, Visión y Control, Universidad de Sevilla, Spain. E-mails: jacevedo@us.es, barrue@us.es, imaza@us.es, aollero@us.es

[1]http://www.ec-safemobil-project.eu/

The paper is organized as follows. In Sect. II the previously presented one-to-one strategy is revisited whereas Section III describes the new block-sharing strategy considering both the one-dimensional and two-dimensional grid partitions. The details of the distributed algorithm and the area division protocol are provided in Sect. IV. Section V shows different simulations that have been performed in order to validate the new technique. Comparisons with the previous one-to-one coordination technique are also presented. Finally, Section VI closes the paper with the conclusions.

II. THE ONE-TO-ONE STRATEGY REVISITED

Given a system of multiple UAVs $Q = \{Q_1, Q_2, \ldots, Q_N\}$ and an irregular area S with surface A, the so-called *one-to-one strategy* to monitoring the area S with a distributed algorithm has been proposed in [1]. The area is divided in N non-overlapped sub-areas S_i according to robots's sensing capabilities in a decentralized manner and the area distribution converges to an optimal partitioning where all aerial robots spend the same time to complete its surveillance.

In order to be self-contained, we borrow here the formulation of the problem given in [1]. The aerial vehicles flies at the same altitude and its sensing capabilities are defined by its maximum coverage speed $a_i^{\max}$ depending on its maximum motion speed $v_i^{\max}$ and its coverage range c_i (these values as assumed constants). Thus, the coverage speed $a_i^{\max}$ is stated as the area covered per second and can be approximated by

$$a_i^{\max} = 2c_i v_i^{\max} \ . \tag{1}$$

Therefore, a sub-area S_i which surface A_i is assigned to each aerial robot Q_i according to the following formula:

$$A_i = a_i^{\max} \frac{A}{\sum_{j=1}^{N} a_j^{\max}}, \forall i = 1, ..., n \ . \tag{2}$$

The one-to-one technique is based on the following *area partitioning paradigm*: the whole area is divided into disjoint sub-areas resulting a grid partition in which each cell is monitored by an aerial robot in a synchronized system. When two neighboring UAVs are closer that the communication range c_i they share information and they spread over the area according to their capabilities. An effective approach to ensure synchronization is proposed in [1] by scheduling the starting point of every aerial robot.

As a consequence, several issues have to be addressed in the strategy: the sub-areas allocation problem, the reconstruction of the boundaries and synchronization maintenance. Although the one-to-one technique ensures convergence to the optimal partition (in which all the aerial robots spend the same time to cover its own cell). In this paper a generalization of the one-to-one strategy is proposed in order to accelerate convergence by reducing the number of operations involved in the de-centralized approach. The key idea is to perform both the sub-areas and boundaries assignment on a set of UAVs instead of just two neighbors. This strategy will be called the *block sharing strategy* in the following.

III. THE BLOCK SHARING STRATEGY

Let us assume an irregular area S to monitor for which only the latitudinally and longitudinally bounds are known. Moreover, the area may contain geographical accidents or *holes* that should be avoided in the surveillance mission. Let $Q = \{Q_1, Q_2, \ldots, Q_N\}$ be a set of N UAVs to monitor the area. In the one-to-one coordination technique [1] an initial simple grid division of the area S is given for which each robot can generate its own coverage path. In this approach when two robots are close enough to establish a communication, they exchange the information and execute the *share & divide* operation, in which the robots join the two areas and divide it according to their capabilities using new vertical or horizontal lines maintaining synchronization. The convergence to the optimal partition has been tested experimentally [1].

Let us suppose that the area is partitioned by parallel strips such that at the beginning, the strips are assigned to different UAVs (see Fig. 1).

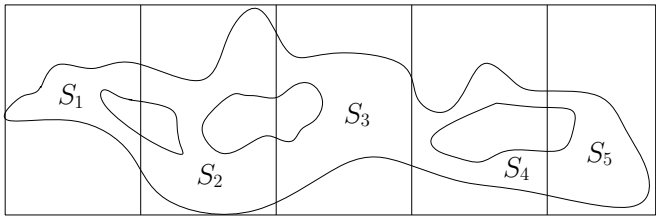


Fig. 1: Unknown irregular area with *holes*. S_i is the area to be covered by the aerial robot Q_i.

Each UAV Q_i has the established initial *link* position with its neighbors, and after a first recognition pass the UAV knows the area S_i to cover in its strip and then can generate its own coverage path. However, due to the irregularity of the area and the different capabilities of each robot, that initial division can not be efficient. Using the strategy *one-to-one* [1] the robots can achieve a near optimal area distribution according to their maximum capabilities. But this process can be slow in many cases, for example if there is a distribution where adjacent areas are very similar. Figure 2 shows an initial partition in which each operation of the *one-to-one* strategy brings little headway.

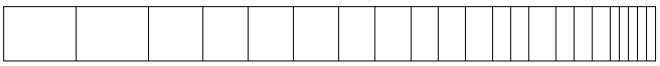


Fig. 2: An initial partition where the *one-to-one* process is slow.

A strategy to accelerate the redistribution area process is introduced in the following. It is based in the *one-to-one* principle, that is, when two robots are located within the communication link they share all the area information in their memories. However, instead of doing the *share & divide* operation at this time, they expect to have enough information to assign areas to all elements in a block. In this paper, a *block* is a set of robots forming a grid and we denote

by (g, h)-block a block of robots in a grid configuration with g rows and h columns. Note that if we use blocks of two robots this approach reduces to the one-to-one strategy.

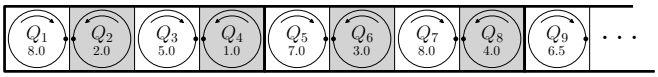
As we started above, the *one-to-one* strategy involves two issues to be solved after the area allocation. First, the boundaries of the new sub-areas have to be redefined and after that, the communication links must be established to maintain synchronization. In this section, we focus on the area allocation task. Other issues will be treated in section IV.

A. The one-dimensional grid partition

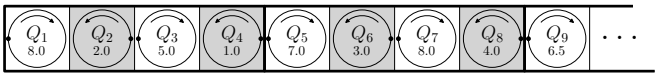
In this section we elaborate on the block sharing strategy when the initial partition is given by an one-dimensional grid as in Figure 1. Let robots in odd positions to have clockwise motion and counter-clockwise motion in other case, see Figure 3. Initially robots Q_i and Q_j only know the areas S_i and S_j, respectively. After a contact between them, Q_i and Q_j know $S_i \cup S_j$. After that, if Q_i contact to Q_k (it has information about $S_k \cup S_l$) then Q_i and Q_k know $S_i \cup S_j \cup S_k \cup S_l$. Now Q_i can allocate the new sub-areas and a new area division between Q_i, Q_j, Q_k and Q_l is performed.

Of course, if each robot wait until knowing all information in the fleet then all of them can make an exact area division. The upper bound for the time to share all information in the fleet with $m \times n$ grid configuration is $T_s \leq 5\frac{T}{4} + (n + m - 4)\frac{T}{2}$ where T is the period of the tour [15]. In many real applications it is not convenient to wait until the aerial robots know all the information about the area; robots could also have limited capacities of computing and memory, preventing to operate with a lot of information, even more in case of possible loss of vehicles.

In this scenario we have a trade-off situation and a suitable value for the time before doing the area allocation has to be decided. For a case study, we propose to work with groups (blocks) of four robots. At the first contact, the pairs of robots $< Q_1, Q_2 >$ and $< Q_3, Q_4 >$ share information, Figure 3a. After a half of period, the pair of robots $< Q_2, Q_3 >$ share information, Figure 3b, and in that moment Q_2 and Q_3 can make an area division between Q_1, Q_2, Q_3 and Q_4. However, Q_1 and Q_4 do not know yet the area assigned to them. In the next contact, Figure 3c, the first area division is completed. The process between $Q_5, \ldots, Q_8$ is similar. The next area division will be processed between the robots $Q_3, \ldots, Q_6$, starting just when the areas of pairs $< Q_3, Q_4 >$ and $< Q_5, Q_6 >$ are equal, after a half of period the pair $< Q_4, Q_5 >$ share information, Figure 3d and they know the area to cover by robots $Q_3, \ldots, Q_6$ and the area division is processed. In the next contact they will complete the area division for the block, see Figure 3e. Notice that, for this case, the area division is performed at the odd contacts. Generalizing the idea, the robots make an area division according to a $(1, 4)$-block alternating groups starting in the position $4k + 1$ and groups starting in the position $4k + 3$, see Figure 4.



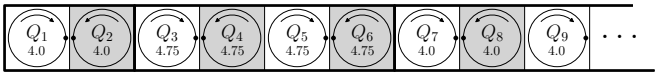
(a) Initial state, first contact.



(b) Half period, second contact.

(c) A period, third contact.

(d) Half period, fourth contact.



(e) A period, fifth contact.

Fig. 3: *Block-sharing* strategy in the one-dimensional grid partition. Robots in light strip have clockwise motion, robots in dark strip have counter-clockwise motion. The numbers inside the cells are the allocated sub-areas.

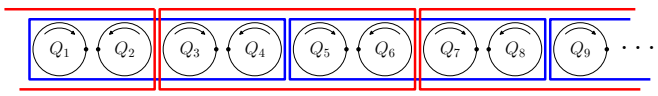


Fig. 4: Blue groups start in robots located at the $4k + 1$ position and the red ones in robots located at the $4k + 3$ position.

B. The two-dimensional grid partition

Now, let suppose we have a big area and, instead a strip initial division, we start by using a grid shape as in Figure 5. Color the grid like a chessboard, and let suppose counter-clockwise motion for the robots located at dark cells and clockwise motion in other case. In this scenario, in order to generalize the one-dimensional case, we use groups of sixteen robots, that is, $(4, 4)$-blocks. The coordination process is similar to the one in the strip partition, alternating divisions between blue groups and red groups as shown in Figure 6. Notice that, in the first step of the strategy, the first time for sub-area allocation is performed by means of the $(4, 4)$-blocks starting at the corner (blue groups in Figure 6). Now, the first division spends 1.25 periods, and after that, the following divisions can be completed one per period.

C. Computational results

We have implemented the *block sharing* strategy, using $(1, 4) - blocks$ and $(4, 4) - blocks$ for strip and grid shape configurations respectively, and we show here some tests comparing this strategy versus *one-to-one*. In this section we are not counting the time required for the location of

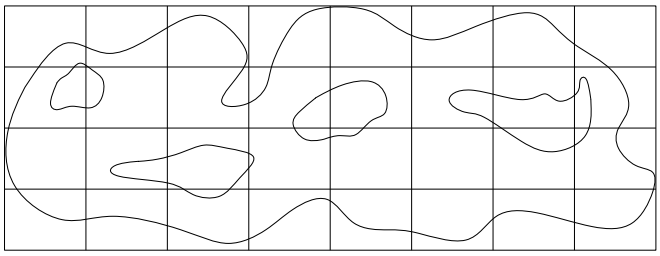


Fig. 5: Grid shape initial division in an unknown irregular area with *holes*.

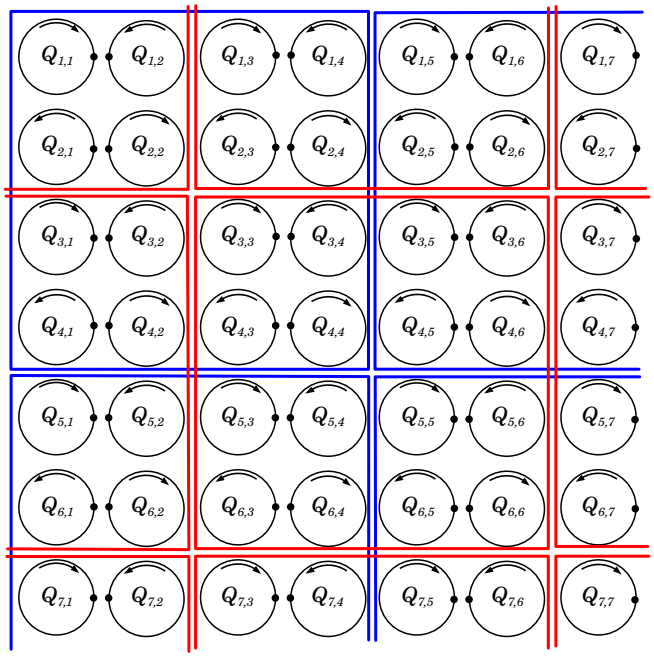


Fig. 6: Groups of sixteen robots (4x4 blocks), blue groups with top-left corner in cells $(4k+1, 4q+1)$ and red groups with top-left corner in cells $(4k+3, 4q+3)$.

new boundaries nor for the synchronization of the system. (Notice that for the one-to-one strategy these issues have to be computed more times than for the block-sharing strategy). The test consists in the generation of 100 random cases of size n in strip division or $n \times n$ in grid shape division. We applied the strategies until the maximum difference between two values become less than or equal 0.1. The diagrams in figures 7 and 8 show a comparison between the average count of needed periods for each strategy when the number of aerial robots (size of the grid) increases. Note that *block-sharing* achieves the goal using approximatively half of periods required by the *one-to-one* process.

IV. SYSTEM IMPLEMENTATION

This section describes the methods and algorithms designed to implement, over the distributed system of multiple aerial robots defined in Sect. II, the block sharing strategy explained in Sect. III. Section III defines how each aerial robot can compute its assigned area size in a distributed manner from an initial unsuitable area division and knowing

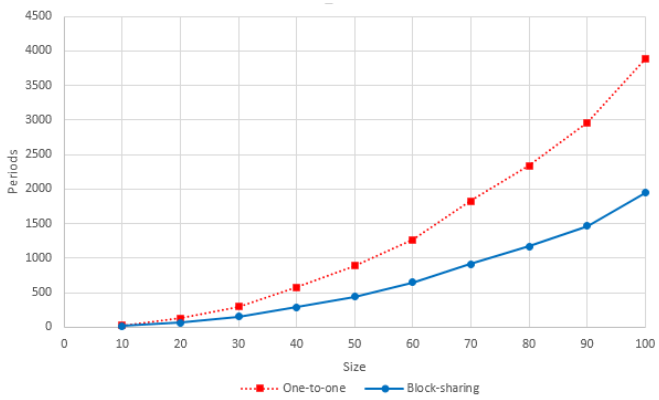


Fig. 7: This graphic shows a comparison between both strategies in a strip division, *size* is the length of the strip.

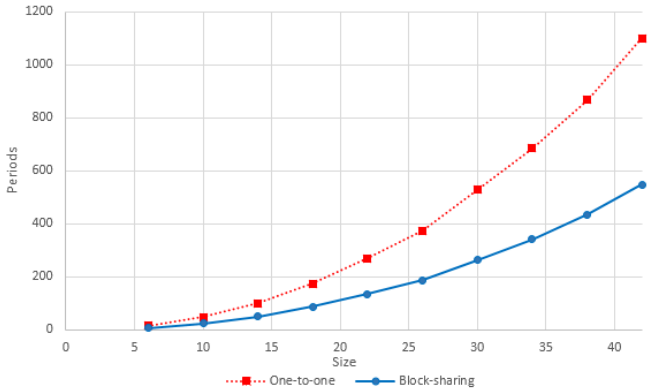


Fig. 8: This graphic shows a comparison between both strategies in a grid shape division, *size* is the length side of a square grid.

no information about the rest of UAVs. However, if we want to implement it in practical area surveillance scenarios, the robots need to decide consistently not just the area size but also the area shape. On other hand, the UAVs should be able to keep their communications links while this is running.

A. Definitions

Given an area S divided as a $m \times n$ grid of sub-areas with $N = m * n$, we can identify each UAV using a couple of indexes to define the row and the column of its sub-area into the matrix. If $ind_i = (ind_i.x, ind_i.y)$ is an index, we define $ind_i.x$ as the row component and $ind_i.y$ as the column component.

On the other hand, we define 4 *link positions* for each sub-area where the owner UAV can contact with another one. This link positions are numbered from 1 to 4 starting from the top in counter-clockwise. We say that an UAV Q_j is *neighbor* of Q_i if both share a common link position. Now, Q_j is the k-neighbor of Q_i if Q_i meet Q_j in the link position numbered as k. Applying the block sharing strategy, the UAVs are divided into groups or blocks. The *UAVs in block* of Q_i are the rest of UAVs that belong to its same block.

For instance, in Figure 9, a 4×4 grid is shown and divided in $(2, 2)$-blocks. Here, the neighbors of Q_{11} are Q_7, Q_{10}, Q_{15} and Q_{12}, its index is $(2, 3)$ and its UAVs in block are Q_{12}, Q_{15} and Q_{16}.

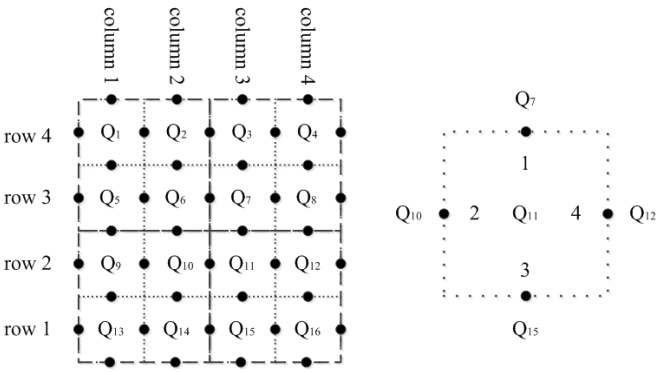


Fig. 9: On the left a rectangular area divided as a 4×4 grid is shown. Each cell is drawn by dotted lines. The dashed lines delimit the $(2, 2)$-blocks. The black points indicate the link positions. On the right, the cell assigned to UAV Q_{11} is detailed. Next to the black points into the cell is indicated the number of link position

B. Required local information

The multi-robot system behaves in a totally distributed manner. All the robots execute the same algorithms, but using their own local information. Each robot Q_i just needs to store the following information.

- The actual area $\mathbf{S}_\mathrm{i}$ that the robot has to patrol, defined as the boundary polygon (ordered list of vertexes).
- The list of the link positions $\mathbf{LP}_\mathrm{i}$. The link position number k can be accessed using $\mathbf{LP}_\mathrm{i}(k)$.
- The list of neighbors UAVs $\mathbf{LN}_\mathrm{i}$, ordered according to their link positions. A link position without associated neighbor is identified in the list with a -1. The k-neighbor is defined as $\mathbf{LN}_\mathrm{i}(k)$.
- The actual path $\mathbf{P}_\mathrm{i}$ to cover its assigned area defined as a ordered list of way-points.
- The list of link positions where there is an UAV in block $\mathbf{BLP}_\mathrm{i}$
- A list of the indexes of the UAVs in block $\mathbf{BI}_\mathrm{i}$. The element k of this list can be accessed as $\mathbf{BI}_\mathrm{i}(k)$, and the row and column components as $\mathbf{BI}_\mathrm{i}(k).x$ and $\mathbf{BI}_\mathrm{i}(k).y$, respectively.
- The list of the maximum area coverage of the UAVs in block $\mathbf{BC}_\mathrm{i}$, ordered according to the list of indexes. This list is initialized setting all the elements to -1. The element k of this list is accessed using the expression $\mathbf{BC}_\mathrm{i}(k)$.
- The list of the areas of the UAVs in block $\mathbf{BS}_\mathrm{i}$, defined as the boundary polygons. The elements of this list are initialized as empty sets. The area k of this list can be accessed as $\mathbf{BS}_\mathrm{i}(k)$
- And, finally, a vector $\mathbf{X}_\mathrm{i}$ which contains information about its actual status and capabilities: identifier i, grid index $\mathbf{ind}_\mathrm{i}$, maximum motion speed $v_i^{\max}$, coverage range c_i, maximum coverage speed $a_i^{\max}$, direction d_i in which the robot has to travel its path, next way-point to visit, next link position to visit l_i, a variable b_i to indicate the present block considered (with available values: 1 or -1), actual position $\mathbf{p}_\mathrm{i}$ and instantaneous velocity $\mathbf{v}_\mathrm{i}$.

So, the required information that each UAV needs to stores depends on the size of the block but not on the size of the grid (it means, the total amount of UAVs). Therefore, it is an scalable system.

C. Distributed algorithm

Given the area S to be covered by the UAVs, and using a simple initial division as is shown in Figure 5 and proposed in [1], each UAV Q_i is initialized with an initial assigned sub-area $\mathbf{S}_\mathrm{i}$, an initial list of link positions $\mathbf{LP}_\mathrm{i}$, a list of neighbors $\mathbf{LN}_\mathrm{i}$, its direction d_i, its index ind_i and the first link position to visit l_i. Direction and first link position are alternated between 1 and -1, and 1 and 3 respectively, such as neighbors UAVs have always different values for both parameters. The first way-point to be visited is initialized to the first link position $\mathbf{LP}_\mathrm{i}(l_i)$.

Then, each UAV can execute the algorithm 1 in a distributed manner.

The process is as follows. First, each UAV Q_i plans a closed path $\mathbf{P}_\mathrm{i}$ to cover its initial assigned area $\mathbf{S}_\mathrm{i}$ using the `plan` function and according to the initial list of link positions $\mathbf{LP}_\mathrm{i}$. This function generates a pseudo-symmetric path such as is defined in [1]. The UAV decides the indexes $\mathbf{BI}_\mathrm{i}$ and the list of the link positions $\mathbf{BLP}_\mathrm{i}$ of the UAVs in block using to `block` function, see subsection IV-E. Then, the UAVs initialize the list of areas $\mathbf{BS}_\mathrm{i}$ and list of sensing capabilities $\mathbf{BC}_\mathrm{i}$ of UAVs in block using the `initialize` function. All the areas less its own area are initialized to empty sets. And all the capabilities less its own are initialized to -1.

Then, the robots start moving (`move` function) following the path $\mathbf{P}_\mathrm{i}$ depending on their direction d_i while monitor the area looking for area changes and informations of interest using the `monitor` function. If an UAV reaches a link position but it is not in its block, it updates (`next` function) its next link to visit l_i according to its own direction d_i and continues moving.

If the reached link position is in its list of link positions in block and there it should meet a neighbor, but the UAV can not contact with anyone, it stop and wait until the neighbor reaches using the `stop` function.

However, in case that the link position has not neighbor or has a null neighbor, the UAV updates the lists of areas and sensing capabilities using empty sets and 0 respectively in the proper positions with the `share` function. And, when the UAV reaches a link position and meet its neighbor Q_j, it receive from the neighbor all the information that it has about its UAVs in block (areas $\mathbf{BS}_\mathrm{j}$ and sensing capabilities $\mathbf{BC}_\mathrm{j}$, using the `share` function. In both cases, they updates the next link to visit l_i with the `next` function.

Algorithm 1 Distributed algorithm to generate the area to patrol according to the block sharing strategy.

Require: $\mathbf{S_i}$,$\mathbf{LP_i}$,$\mathbf{LN_i}$,$\mathbf{X_i}$
  $\mathbf{P_i} \leftarrow$ `plan`(c_i,$\mathbf{S_i}$,$\mathbf{LP_i}$)
  $b_i \leftarrow 1$
  [$\mathbf{BI_i}$,$\mathbf{BLP_i}$] $\leftarrow$ `block`(b_i,ind_i,$\mathbf{LP_i}$)
  [$\mathbf{BC_i}$,$\mathbf{BS_i}$] $\leftarrow$ `initialize()`
  while !ABORT **do**
    if $w_i \in \mathbf{BLP_i}$ **then**
      if `meet`(Q_i,Q_j)==**true then**
        [$\mathbf{BS_i}$,$\mathbf{BC_i}$] $\leftarrow$ `share`($\mathbf{BS_j}$,$\mathbf{BC_j}$)
        if $-1 \notin \mathbf{BC_i}$ **then**
          [$\mathbf{S_i}$,$\mathbf{LP_i}$] $\leftarrow$ `divide`($\mathbf{BS_i}$,$\mathbf{BC_i}$,ind_i,$\mathbf{BI_i}$,$\mathbf{LP_i}$)
          $\mathbf{P_i} \leftarrow$ `plan`(c_i,$\mathbf{S_i}$,$\mathbf{LP_i}$)
          $b_i \leftarrow -b_i$
          [$\mathbf{BI_i}$,$\mathbf{BLP_i}$] $\leftarrow$ `block`(b_i,ind_i,$\mathbf{LP_i}$)
          [$\mathbf{BC_i}$,$\mathbf{BS_i}$] $\leftarrow$ `initialize()`
        end if
        $l_i \leftarrow$ `next`(l_i,d_i)
        $\mathbf{X_i} \leftarrow$ `move`($\mathbf{X_i}$)
      else
        if $\mathbf{LN_i}(l_i)$==-1 **then**
          [$\mathbf{BS_i}$,$\mathbf{BC_i}$] $\leftarrow$ `share`($\emptyset$,0)
          if $-1 \notin \mathbf{BC_i}$ **then**
            [$\mathbf{S_i}$,$\mathbf{LP_i}$] $\leftarrow$ `divide`($\mathbf{BS_i}$,$\mathbf{BC_i}$,ind_i,$\mathbf{BI_i}$,$\mathbf{LP_i}$)
            $\mathbf{P_i} \leftarrow$ `plan`(c_i,$\mathbf{S_i}$,$\mathbf{LP_i}$)
            $b_i \leftarrow -b_i$
            [$\mathbf{BI_i}$,$\mathbf{BLP_i}$] $\leftarrow$ `block`(b_i,ind_i,$\mathbf{LP_i}$)
            [$\mathbf{BC_i}$,$\mathbf{BS_i}$] $\leftarrow$ `initialize()`
          end if
          $l_i \leftarrow$ `next`(l_i,d_i)
          $\mathbf{X_i} \leftarrow$ `move`($\mathbf{X_i}$,$\mathbf{P_i}$)
        else
          $\mathbf{X_i} \leftarrow$ `stop`($\mathbf{X_i}$)
        end if
      end if
    else
      if $w_i \in \mathbf{LP_i}$ **then**
        $l_i \leftarrow$ `next`(l_i,d_i)
      end if
      $\mathbf{X_i} \leftarrow$ `move`($\mathbf{X_i}$,$\mathbf{P_i}$)
    end if
    [$\mathbf{X_i}$,$\mathbf{S_i}$] $\leftarrow$ `monitor`(w_i,$\mathbf{X_i}$)
  end while

When an UAV has all the necessary information about its block (areas $\mathbf{BS_i}$ and sensing capabilities $\mathbf{BC_i}$), it uses the `divide` function to compute the new area to patrol $\mathbf{S_i}$, according to the sensing capabilities $\mathbf{BC_i}$ and the list of link positions $\mathbf{LP_i}$. More details about the `divide` function can be found in subsection IV-D. Then, it generates the closed pseudo-symmetric path to cover $\mathbf{P_i}$ with the `plan` function, changes the variable b_i and uses the `block` function to decide its new block (indexes $\mathbf{BI_i}$ and link positions $\mathbf{BLP_i}$) initializing the list of areas and sensing capabilities (`initialize` function).

D. Area division protocol

An area division protocol has to be defined for all the UAVs, such that they can divide the area and choose their assigned sub-area in a coherent manner. It means, there are not regions without being assigned to any UAV or regions assigned to more than one UAV.

According to the algorithm 1, when an UAV has all the information about its actual block, it decide its new sub-area. We propose a general method to decide in a distributed but coherent manner which sub-area to choose according to the information about the block. The issue is to join the areas of all the UAVs in the block and divide it depending on UAVs sensing capabilities and UAV index with straight and parallel lines.

Therefore, the `divide` function runs as follows. A new area $\mathbf{S}_{\text{block}}$ is generated joining all the areas included in the list of areas $\mathbf{BS_i}$ and, then, the whole surface of this area A_{block} can be calculated.

Given the index of the UAV ind_i, let a_{left} be the sum of all the elements of the sensing capabilities list $\mathbf{BC_i}$ which has a column component in the list of indexes $\mathbf{BI_i}$ less than $ind_i.y$. Now, let a_{middle} be the sum of elements which column components are equal than $ind_i.y$ and, finally, define a_{right} as the sum of the rest of elements. Then, we can easily compute (4) the surface of area A_{middle} that should be assigned to all the UAVs which column component is $ind_i.y$.

$$A_{left} = \frac{a_{left}}{a_{left} + a_{middle} + a_{right}} A_{block} \tag{3}$$

$$A_{middle} = \frac{a_{middle}}{a_{left} + a_{middle} + a_{right}} A_{block} \tag{4}$$

$$A_{right} = \frac{a_{right}}{a_{left} + a_{middle} + a_{right}} A_{block} \tag{5}$$

Using two vertical straight lines, the whole area $\mathbf{S}_{\text{block}}$ is divided in three parts which surfaces are from left to right (3), (4) and (5), respectively, see Figure 10a. We define S_{middle} as the area between the two lines.

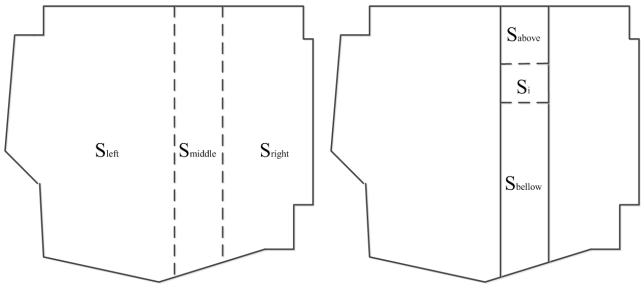
A similar process but depending on the row components of the indexes $\mathbf{BI_i}$ and $ind_i.x$ is used to obtain the assigned area $\mathbf{S_i}$ from the area obtained in the previous step S_{middle}. Parameter a_{above}, a_{below} and a_i are defined as the sum of the elements of the sensing capabilities which column component is $ind_i.y$ and which row component is less, greater or equal, respectively, than $ind_i.x$. Using the weighted average (7), we compute the surface A_i than should be assigned to the UAV Q_i from the surface A_{middle}.

$$A_{above} = \frac{a_{above}}{a_{above} + a_i + a_{below}} A_{middle} \tag{6}$$

$$A_i = \frac{a_i}{a_{above} + a_i + a_{below}} A_{middle} \tag{7}$$

$$A_{bellow} = \frac{a_{bellow}}{a_{above} + a_i + a_{below}} A_{middle} \tag{8}$$

Finally, using two horizontal straight lines, the area $\mathbf{S}_{\text{middle}}$ is divided in three parts which surfaces are from above to bellow (6), (7) and (8), respectively, see Figure 10b. We define $\mathbf{S}_{\text{i}}$ as the area between the lines. All the links positions of the list $\mathbf{LP}_{\text{middle}}$ which associated neighbors belong to the present block are updated, obtaining a common position for each pair of neighbors.



(a) (b)

Fig. 10: This figure shows the process to generate the area $\mathbf{S}_{\text{i}}$ from $\mathbf{S}_{\text{block}}$.

E. Deciding blocks

A relevant issue for each UAV to apply the block strategy is deciding which others UAVs belong to its own block at any time. According to the block strategy described in Sect. III any UAV should change its block when it has received all the information about its old block and has calculated its new area. Then, UAVs can mix the areas obtained from different blocks enabling to converge to a global solution.

The `block` function allows each UAV Q_i to decide which indexes include in the list $\mathbf{BI}_{\text{i}}$ according to a parameter b_i. Therefore, the list $\mathbf{BLP}_{\text{i}}$ is also updated. Obviously, this function depends on the size of the blocks. However, in general, this function can decide two different blocks for each UAV. The process is as follows.

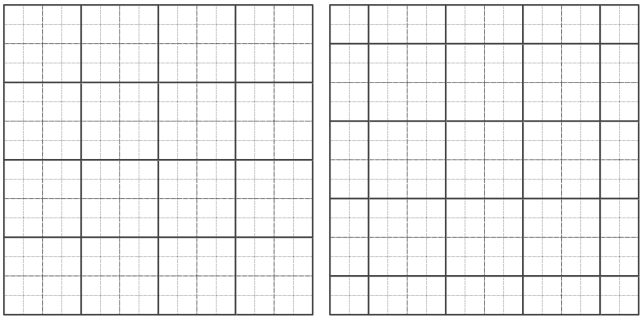
Given a $m \times n$ grid, (g, h)-blocks and assuming that g and h are even, the grid can be divided in sub-blocks of size $\frac{g}{2} \times \frac{h}{2}$ starting from the index $(1, 1)$. Each sub-block has 4 adjacent sub-blocks. Now, the adjacent sub-blocks are joined starting from the index $(1, 1)$ and forming (g, h)-blocks. These will be the blocks when $b_i = 1$. On other hand, joining each sub-block with its adjacent sub-blocks that are not joined previously, it obtained the (g, h) blocks for $b_i = -1$. Figure 11 summarizes both options.

V. VALIDATION RESULTS

As a case of study $(2, 2)$-blocks sharing strategy has been implemented using MATLAB as is described in Sect. IV. The simulations have been executed using parallel objects to test the decentralized and distributed issues. In these cases, rotary wing UAVs are considered.

A. Comparisons with the one-to-one technique

A large set of more than 60 scenarios has been executed with team of UAVs of different sizes and different areas. A communications range of 3 meters has been considered. The



(a) Blocks for $b_i = 1$. (b) Blocks for $b_i = -1$.

Fig. 11: This figure shows as a 16×16 grid should be divided in $(4, 4)$-blocks depending on the variable b_i. Sub-areas, sub-blocks and blocks are delimited by dotted, dashed and thick solid lines, respectively.

coverage range is of 3 meters for all the UAVs. However, the maximum speeds are randomly chosen for each UAV using an standard uniform distribution between 0.5 and 1 meters. Each scenario has been executed using both: the proposed $(2, 2)$-blocks sharing strategy and the one-to-one strategy described in previous work [1]. It is assumed that a system converges when the maximum difference between the actual areas and the desired ones is less than 1%. Figure 12 summarizes the results obtained.

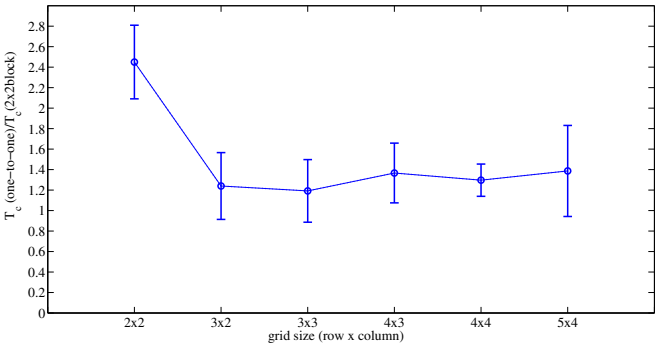
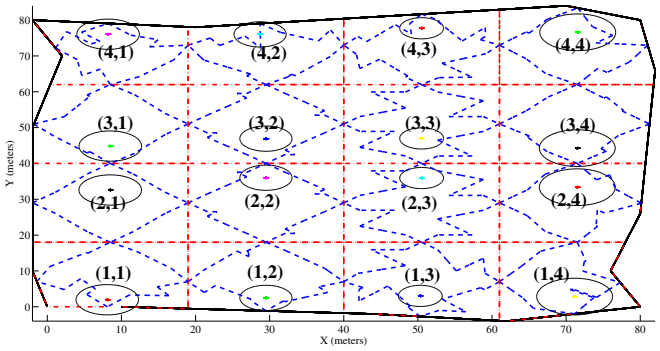


Fig. 12: This figure shows the average relation between the convergence times using the one-to-one and a $(2, 2)$-blocks sharing strategy ($\pm$ its standard deviation) depending on the size of the grid.

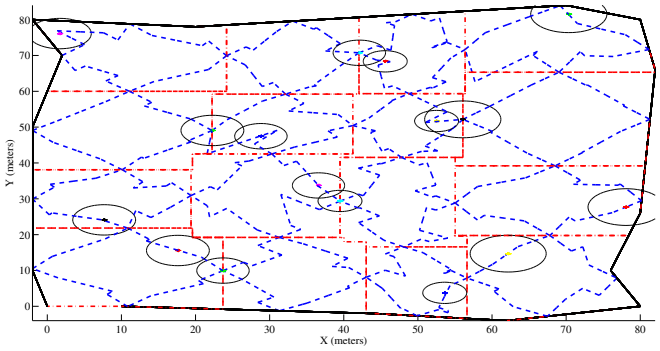
It shows as the block sharing strategy using $(2, 2)$-blocks converges about a 30% quicker than the one-to-one strategy for grid size larger than $(3, 2)$-blocks. When the size of the grid is equal to the size of the block ($(2, 2)$-blocks), the results are even more advantageous.

B. An irregular area with heterogeneous UAVs

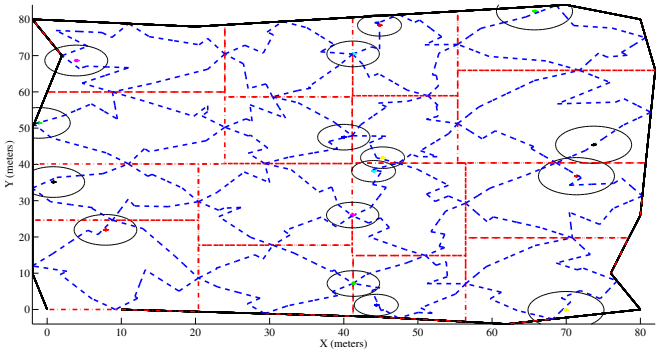
In this case, an irregular area and 16 rotary wing UAVs with different sensing capabilities and a communications range of 3 meters are considered. Figure 13a shows the scenario and the initial unsuitable area division and the table I describes the UAVs sensing capabilities and its sub-areas assigned indexes.



(a) Initial area division



(b) One-to-one strategy.



(c) $(2, 2)$ Block-sharing strategy.

Fig. 13: Scenario with an irregular area and 16 UAVs. (a) simulation initial state, (b) and (c) correspond to the area division obtained during the test at time $t = 600s$. Pairs of number between brackets are the sub-areas indexes, solid black lines indicate the irregular area, red dashed lines the initially assigned sub-areas, the blue dashed lines the covering paths and the black circles the different coverage ranges. A video from the test using the block sharing strategy can be viewed in *http://www.youtube.com/watch?v=-oKOcEirQq8*.

Figure 14 shows the maximum relative difference $(\%)$ between the actual surface assigned to each UAV and the desired one (according to expression (2)) computed during the tests (for both strategies). In Figure 13 the final area divisions (at time $t = 600\ s$) obtained for both strategies is drawn. From these results can be deducted that the block sharing converges quicker and steeper than the one-to-one strategy and obtains less concave sub-area shapes.

i	ind_i	v_i^{max} m/s	c_i m
1	(1,1)	0.77	4.28
2	(1,2)	0.75	3.58
3	(1,3)	0.79	3.00
4	(1,4)	0.61	5.20
5	(2,1)	0.88	4.28
6	(2,2)	0.66	3.58
7	(2,3)	0.65	3.00
8	(2,4)	0.95	5.20
9	(3,1)	0.61	4.28
10	(3,2)	0.98	3.58
11	(3,3)	0.97	3.00
12	(3,4)	0.70	5.20
13	(4,1)	0.95	4.28
14	(4,2)	0.83	3.58
15	(4,3)	0.66	3.00
16	(4,4)	0.68	5.20

TABLE I: UAVs sensing capabilities.

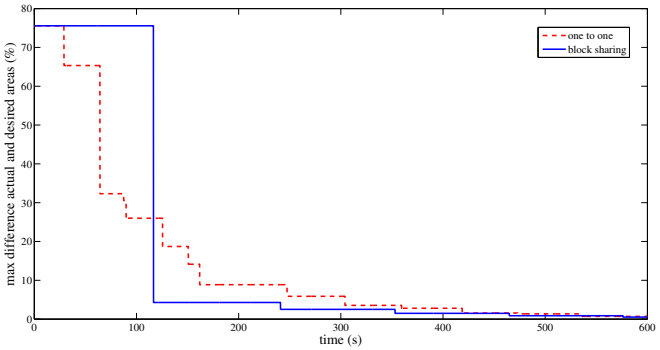


Fig. 14: This figure shows how the maximum relative difference between the actual area assigned to each UAV and the optimal one changes during the test using both the one-to-one (dashed red line) and the $(2, 2)$-blocks sharing (solid blue line) strategies.

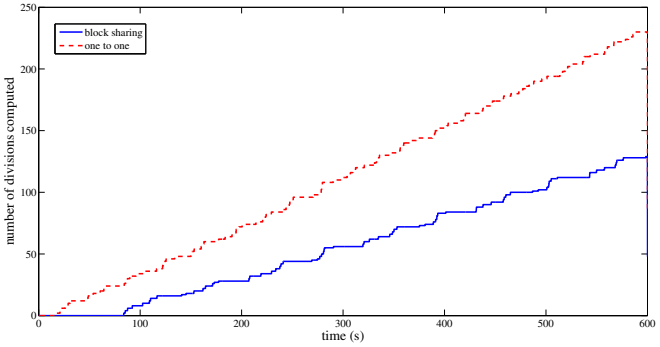


Fig. 15: This figure shows how many divisions operations performed by the UAVs is increasing during the test using both the one-to-one (dashed red line) and the $(2, 2)$-blocks sharing (solid blue line) strategies.

Finally, Figure 15 shows the total sum of divisions performed by the UAVs at any time during the tests. Note that the block sharing strategy needs many fewer operations than the one-to-one strategy to obtain a near-optimal area division.

VI. CONCLUSIONS

The one-to-one strategy was described in [1] to allow a team of multiple UAVs to converge in a distributed and decentralized way to an area partition where each robot has an area to monitoring according to it's capabilities. However, in the present paper, it is shown how to accelerate the convergence in more general scenarios where of the one-to-one strategy is time-consuming and slow.

Therefore, a new distributed strategy called *block sharing* strategy is described and proposed as one-to-one generalization to obtain a quicker convergence. While with the one-to-one strategy the UAVs perform an area division always they meet a neighbor, with the proposed block-sharing strategy the UAVs wait until getting information about a block of UAVs and then perform the area division. This delay in the area partitioning allows to execute fewer operations of sub-area allocation, reconstruction of boundaries and synchronization maintenance.

Validation results show how a $(2, 2)$-block sharing strategy can be successfully implemented in a distributed manner in mutli-UAV systems, assuming communications constraints and rotary wings UAVs with different sensing capabilities. In Section V are shown some diagrams that illustrate the improvements obtained in simulations. The block-sharing strategy is 30% faster than the one-to-one approach. Moreover, the fact of avoiding to recalculate the boundaries in each meeting leads to *nicer* boundaries and reduces the number of needed operations to achieve the optimal solution in the decentralized system. These improvements are increased with the size of the blocks, as is shown in subsection. III-C.

REFERENCES

[1] J. J. Acevedo, B. C. Arrue, J. M. Diaz-Banez, I. Ventura, I. Maza, and A. Ollero, "One-to-one coordination algorithm for decentralized area partition in surveillance missions with a team of aerial robots," *Journal of Intelligent and Robotic Systems*, vol. 74, pp. 269–285, April 2014.

[2] A. Ollero and I. Maza, eds., *Multiple heterogeneous unmanned aerial vehicles*. Springer Tracts on Advanced Robotics, Springer-Verlag, 2007.

[3] I. Maza, F. Caballero, J. Capitan, J. M. de Dios, and A. Ollero, "A distributed architecture for a robotic platform with aerial sensor transportation and self-deployment capabilities," *Journal of Field Robotics*, vol. 28, no. 3, pp. 303–328, 2011.

[4] J. Acevedo, B. Arrue, I. Maza, and A. Ollero, "Cooperative large area surveillance with a team of aerial mobile robots for long endurance missions," *Journal of Intelligent and Robotic Systems*, vol. 70, pp. 329–345, 2013.

[5] L. Merino, F. Caballero, J. M. de Dios, I. Maza, and A. Ollero, "An unmanned aircraft system for automatic forest fire monitoring and measurement," *Journal of Intelligent and Robotic Systems*, vol. 65, no. 1, pp. 533–548, 2012.

[6] M. Baseggio, A. Cenedese, P. Merlo, M. Pozzi, and L. Schenato, "Distributed perimeter patrolling and tracking for camera networks," in *Decision and Control (CDC), 2010 49th IEEE Conference on*, pp. 2093 –2098, dec. 2010.

[7] Y. Elmaliach, A. Shiloni, and G. A. Kaminka, "A realistic model of frequency-based multi-robot polyline patrolling," in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 1*, AAMAS '08, (Richland, SC), pp. 63–70, International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[8] N. Agmon, G. A. Kaminka, and S. Kraus, "Multi-robot adversarial patrolling: facing a full-knowledge opponent," *J. Artif. Int. Res.*, vol. 42, pp. 887–916, sep 2011.

[9] Y. Chevaleyre, "Theoretical analysis of the multi-agent patrolling problem," in *Intelligent Agent Technology, 2004. (IAT 2004). Proceedings. IEEE/WIC/ACM International Conference on*, pp. 302 – 308, sept. 2004.

[10] F. Pasqualetti, A. Franchi, and F. Bullo, "On cooperative patrolling: Optimal trajectories, complexity analysis, and approximation algorithms," *Robotics, IEEE Transactions on*, vol. 28, pp. 592 –606, June 2012.

[11] S. Smith and D. Rus, "Multi-robot monitoring in dynamic environments with guaranteed currency of observations," in *Decision and Control (CDC), 2010 49th IEEE Conference on*, pp. 514 –521, dec. 2010.

[12] D. Kingston, R. Beard, and R. Holt, "Decentralized perimeter surveillance using a team of UAVs," *Robotics, IEEE Transactions on*, vol. 24, pp. 1394 –1404, dec. 2008.

[13] J. Acevedo, B. Arrue, I. Maza, and A. Ollero, "Cooperative perimeter surveillance with a team of mobile robots under communication constraints," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 5067–5072, Nov 2013.

[14] R. Carli, A. Cenedese, and L. Schenato, "Distributed partitioning strategies for perimeter patrolling," in *American Control Conference (ACC), 2011*, pp. 4026 –4031, June 2011.

[15] J. Acevedo, B. Arrue, J. Diaz-Banez, I. Ventura, I. Maza, and A. Ollero, "Decentralized strategy to ensure information propagation in area monitoring missions with a team of UAVs under limited communications," in *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS 2013)*, pp. 565–574, 2013.