computing@computingonline.net
www.computingonline.net

Print ISSN 1727-6209
On-line ISSN 2312-5381
International Journal of Computing

PARTITIONING KNOWN ENVIRONMENTS FOR MULTI-ROBOT TASK ALLOCATION USING GENETIC ALGORITHMS

Md Maksudur R. Mazumder, Chris Phillips

School of Electronic Engineering and Computer Science,
Mile End Road, London E1 4NS, United Kingdom,
m.mazumder@qmul.ac.uk, chris.i.phillips@qmul.ac.uk

Paper history:
Received 12 June 2019
Received in revised form 25 May 2020
Accepted 08 July 2020
Available online 27 September 2020

Keywords:
task-allocation;
terrain partitioning;
known environments;
multi-robot;
exploration;
Unmanned Aerial Vehicle (UAV);
Genetic Algorithm (GA);
search optimization.

Abstract: A common challenge facing emergency services, particularly in response to fires and/or earthquakes, is the location and subsequent extraction of people from hazardous buildings in a timely manner. This usually requires emergency service workers to enter the building and put their own lives at risk, even when there may be no people to extract. However, given recent advances in autonomous robotics, drones are expected to help humans in tasks such as search and rescue, and similar tasks, where coverage and time are key parameters. The aim is to complete a comprehensive search of the environment as quickly as feasible. Using multiple drones rather than a single drone can reduce search time, although performance can be poor if the searching is non-coordinated. Therefore, partitioning a terrain is important in order to effectively distribute the drone search work so that good coverage can be achieved in a reasonable amount of time and redundant searching is eliminated. In this paper a novel Square Based Terrain Partitioning (SBTP) algorithm is presented using a genetic algorithm to partition a known environment into multiple domains in a multi-robot exploration system. In addition, a second genetic algorithm is presented to allocate the domain search workload such that, given a certain number of drones, the overall search time is minimized.

Effective task allocation is important when attempting to achieve high utilization in multi-robot exploration systems [1, 2]. In this process a group of robots collaborate to complete some complex task with better performance than using a single robot[1] or drone. The purpose of the task assignment is to minimize the task completion time whilst maximizing the search utilization of each robot.

[3] proposed a task assignment method based on social-welfare for multi-robot systems, the aim being to achieve the task completion and minimize resource consumption proportion to an acceptable amount. Zhang and Xie [4] put forward an adaptive task assignment method for multiple mobile robots based on swarm intelligence to perform cooperative tasks in unknown dynamic environments. A hierarchical architecture was applied to each robot. An alternative method based on a Genetic Algorithm (GA) was used for real-time task assignment of Multi-Unmanned Combat Aerial Vehicle in uncertain environments [5]. Irfan Younas et al. [6] also proposed a method based on GA for task assignment problems. In their approach, a group of collaborating agents work as a team to complete the tasks assigned to them.

In [1] the authors use a GA to solve a multi-robot task assignment problem. In their method the overall search area is divided into subareas and the authors assume that the size of the subarea assigned to each robot is the same and equal. However, the authors do not describe the area partitioning technique. Wei Sun et al. [7] propose a solution for the multi-robot task assignment in an environment with obstacles. The solution combines the A* algorithm with a genetic algorithm. The A* algorithm is used to find an optimal path whereas the GA is used to solve the task assignment problem. In [8] the authors

[1] In this research we freely interchange the terms drones, robots and vehicles. All of these terms refer to the same thing. Typically we use terminology that is consistent with the relevant state-of-the-art and in the given context.

introduce a Balanced Partitioned Surveillance Task (BPS) for multi-robot systems. Indeed, there are many methods to solve multi-robot task assignment problems, such as task assignment for robots with a limited communication range [9], vacancy chain scheduling for multi-robot task allocation [10], and clustering-based algorithms for multi-vehicle tasks [11]. For example, even multi-robot task allocation for cleaning public spaces has been put forward [12]. The aim is to allocate a group of robots to each cleaning zone based on status of the robot and the environment [12]. Also, dynamic task allocation is necessary process for proper management of the swarm in a robotic swarm systems. It allows the robotic swarm to do multiple tasks in such a way that a pre-defined proportion of execution of those tasks is achieved [13]. However, although there are many methods for solving Multi-Robot Task Assignment (MRTA) problems, every method has its own limitations. Different problems lend themselves to different partitioning and allocation schemes.

We propose a scheduling task allocation mechanism using GA that allocates one or more search domains to specific drones in a way that, given a certain number of drones, the overall average search time is minimized. To achieve this the given terrain needs to be partitioned into discrete search domains.

A challenging task in the field of mobile robotics is the efficient exploration of environments. Multi-robot exploration algorithms often rely on occupancy grids [14]. However, if these environments are very large, search performance can be poor. Conversely, polygonal representations do not have this limitation [14]. Partitioning of an unknown terrain into a number of regions equal to the number of available robots has been proposed using K-means clustering with an occupancy grid representation [14]. According to [15], an unknown area is partitioned into multiple sub-regions and each different region is assigned to a number of available robots. In [15], the author uses a circle partitioning method. The aim is to explore the environment as quickly as possible. Furthermore, the coordinated use of multiple cleaning robots has been considered by many researchers (e.g., [16-18]). In [16] the authors employ an area partitioning method using a balloon model to divide the cleaning area into a number of equal-size subareas, each one assigned to a specific robot. However, they do not consider the shapes and locations of obstacles. In [17] an extended partitioning method considers identifying dirty areas in the environment. However, once again, they do not consider obstacles. As an alternative, in [14], the authors use a Voronoi space partitioning algorithm to partition an obstacles free space. However, no information is provided on the partitioning process when obstacles are present. On the other hand, a circle partitioning method, based on available robots, where obstacles occupy the search space is presented in [15].

In this paper, a novel SBTP algorithm is implemented using the GA to partition a known environment into multiple sub-regions for multi-robot exploration. The aim is to achieve good search coverage in a reasonable amount of time with a given number of drones. The SBTP algorithm can partition any sized known rectangular terrain/space with or without obstacles.

The remainder of the paper is organized as follows. In the Section 2, the Square Based Terrain Partitioning (SBTP) algorithm is proposed to partition a known environment. In Section 3 an efficient GA-based scheme is proposed for allocating drones to these search partitions. Experimental results and analysis are presented in Section 4. Finally, in Section 5, conclusions and future work are discussed.

2. TERRAIN PARTITIONING INTO DISCRETE DOMAINS

In this section, a novel SBTP algorithm is presented using a GA to partition known environments with and without fixed obstacles (such as walls) into multiple sub-regions for multi-robot exploration. By known environments we mean, the floor-plan is available ahead of exploration. The GA mechanism is explained in Section II [19].

With this approach, we first select the number of domains. The algorithm then selects valid search origin points for each domain (i.e. x, y coordinates that do not correspond to walls, water, or other illegal terrain) and evaluates their potential using a GA. The algorithm expands the domain from this origin(s) by checking whether the location ((x+i, y) OR (x-i, y)) valid or not. Initially xmax & xmin = x and i = 1. If either location (x+i, y) OR (x-i, y) is valid then the algorithm will mark those grid-squares (x, y) as belonging to the domain and set xmax = x+i or xmin = x-i. The index i is then incremented. The process continues in parallel for each domain until an obstacle or another domain is encountered. If both of the location (x+i, y) and (x-i, y) is not valid then the algorithm resets the index, i=1, and commences checking the row[(xmax, y+i) to (xmin, y+i)] and row[(xmax, y-i) to (xmin, y-i)] whether these rows are valid or not. This takes place asymmetrically, expanding the row[(xmax, y+i) to (xmin, y+i)] OR row[(xmax, y-i) to (xmin, y-i)] whilst valid until obstacle(s) or other domains are encountered. As the expansion takes place, the grid-squares (x,y) of valid row(s) are marked and the index, i, is incremented. When both of the row expansion processes cease, i.e. because row[(xmax, y+i) to (xmin, y+i)] and row[(xmax, y-i) to (xmin, y-i)] are not valid then the algorithm will stop the partitioning process for the domain(s) concerned.

The algorithm completes when all the domains have been expanded to their fullest extent [19].

As normal with GA schemes, multiple guesses are evaluated and refined to obtain a better partitioning of the terrain with or without obstacle(s). With this scheme 100% coverage is possible, but not always in a terrain that contains obstacle(s). In addition, the area of different domains can be unbalanced as the algorithm will discontinue expanding a specific domain when other domains or obstacles are encountered [19].

The SBTP algorithm flow chart is presented in Fig. 1.

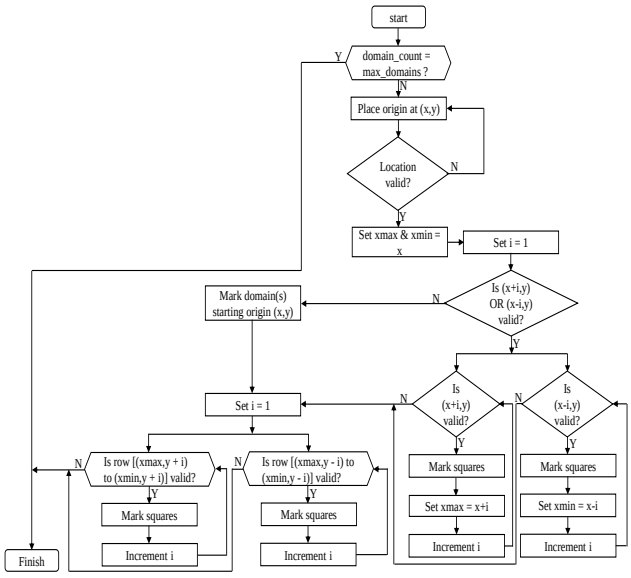
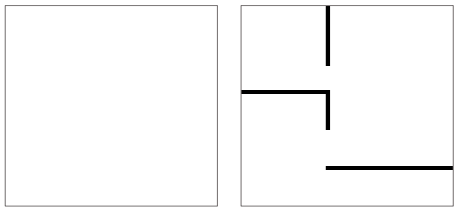


Figure 1 – Square based terrain partitioning algorithm flow chart

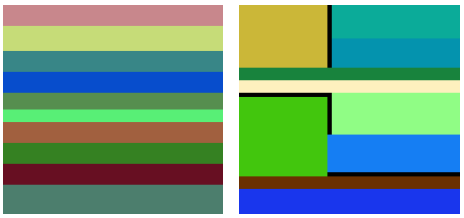
Fig. 2, provides an example of a typical 50x50 floor plan containing free space (white areas) and obstacles (black areas). The given floor-plan is partitioned using the partitioning-algorithm as described in the flow chart of Fig. 1.



a: Floor-plan (obstacle free). b: Floor-plan containing obstacles (black areas).

Figure 2 – Typical 50x50 floor-plan (before partition)

Fig. 3 gives an example of the SBTP algorithm results where each colour indicates an individual domain. The simulation is run for 100 generations, using 100 chromosomes, 10% elite, 30% kill with a crossover and mutation rate of 50% & 30% respectively.



a: Terrain partitions into multiple domains in free area. b: Terrain partitions into multiple domains in obstacles area.

Figure 3 – Typical 50x50 floor-plan after partition into discrete domains using SBTP algorithm where each colour indicates distinct domains

In Fig. 3, we can see the area of different domains is somewhat unbalanced because the SBTP algorithm expands the partition asymmetrically. As a result, it will stop expanding domain partitions when domains encounter each other, to prevent overlapping, or if obstacles prevent further growth.

3. DOMAIN EXPLORATION WORK ALLOCATION MECHANISM

The concept of a Genetic Algorithm (GA) was first proposed by John Holland [20]. A GA is a search heuristic that is used to generate solutions for optimization and search problems. It commences its search with a set of random solutions (represented by chromosomes), called a population, usually coded into binary string structures. Solutions from one population are used to create a new population. The new solutions are called the offspring, which are selected according their fitness; the more suitable they are, the more chances they have to survive and reproduce. A flow chart for a traditional GA is presented in [1].

In this section, we describe a GA to allocate tasks between drones in a multi-robot exploration system. The purpose of this algorithm is to allocate search domains to drones so that, a given certain number of drones, the overall search time is minimized whilst maximizing the utilization of each drone. We assume M≥N; where M denotes maximum number of domains and N indicates the total number of drones. Therefore, the task scheduling assignment mechanism is implemented in such way that one or more drones will search separate multiple domains. The proposed GA differs from traditional GAs in its fitness function.

3.1. CHROMOSOME ENCODING

Genetic algorithms process chromosomes, each chromosome comprising a set of alleles. The group of chromosomes used in one generation (i.e. an iteration of an experiment) is referred to as the population. Each chromosome represents a possible solution to a problem and can be encoded in a number of ways when designing the experiment. In this paper value encoding is used where each and

every chromosome is a set of some values, i.e., we construct a chromosome as an array of x,y domain ($D_1 \ldots D_n$) origin coordinates and allocate a drone to each of 'n' domain as shown in Table 1.

Table 1. Chromosome encoding structure

	Number of Domains	D_1	D_2	D_3	...	D_n
C_i	**Drone numbers**	R_1	R_2	R_2	...	R_n

For example, we partition a 20x20 terrain/floor-plan into 4 discrete domains using the SBTP algorithm shown in Fig. 1 and obtain the domains:

$D_1(x,y) = (1,1 \ldots 5,5)$
$D_2(x,y) = (1,7 \ldots 9,9)$
$D_3(x,y) = (6,10 \ldots 10,6)$
$D_4(x,y) = (6,1 \ldots 10,5)$

Suppose we have 3 drones (R_1, R_2 & R_3) to allocate the search work to for these domains. Then one possible chromosome encoding (C_1) for each domain is represented in Table 2.

Table 2. An example of chromosome encoding

	Number of Domains	D_1	D_2	D_3	D_4
C_1	**Drone numbers**	R_3	R_1	R_3	R_2

3.2. FITNESS FUNCTION

The fitness function simply takes a candidate solution to the problem as input and produces a score based on how fit the solution is in respect to the problem under consideration. In this paper, the objective is to allocate the domain searching work in such way that the overall average search time is minimized given a specific number of drones. Therefore, the fitness function is designed as follows:

$$R_n = \alpha_{count}(D_i); \quad n = \{1,2,3 \ldots N\}, i = \{1,2,3 \ldots M\}, \quad (1)$$

where R_n represents the count of occurrences of n^{th} drone/robot that is allocated to the i^{th} domain according to the chromosome encoding. N and M denote the total number of drones and total number of domains, respectively. D_i represents the i^{th} domain and α_{count} is a method that returns the sum of the domains that are allocated to n^{th} drone.

Suppose, we have 4 drones (R_1, R_2, R_3, & R_4) and 7 domains (D_1, D_2, D_3, D_4, D_5, D_6, & D_7). One possible chromosome encoding (C_1) for each domain is:

Table 3. Example of a possible chromosome encoding to calculate the fitness function

	Number of Domains	D_1	D_2	D_3	D_4	D_5	D_6	D_7
C_1	**Drone numbers**	R_4	R_2	R_1	R_3	R_2	R_3	R_4

From Table 3, we obtain the following result using eq. (1):

$R_1 = \alpha_{count}(D_3) = 1$
$R_2 = \alpha_{count}(D_2 + D_5) = 2$
$R_3 = \alpha_{count}(D_4 + D_6) = 2$
$R_4 = \alpha_{count}(D_1 + D_7) = 2$

Separate from the actual domain allocation given by any specific chromosome, we employ eq. (2) iteratively to calculate what should be a balanced distribution of the domain exploration work among the drone(s). More precisely, suppose we have 4 domains (D_1, D_2, D_3 and D_4) and 2 drones (R_1 and R_2), then the domain allocation for R_1 and R_2 should be 2 and 2, respectively.

$$\mu_j = \left\lceil \frac{(M - \mu_{j-1} - \mu_{j-2} - \cdots - \mu_0)}{(N-(j-1))} \right\rceil, \quad (2)$$

where $j = \{1,2,3...N\}$ & $\mu_0 = 0$. μ_j, M and N represent a count of the j^{th} domain exploration workload areas, total number of domains and total number of robots, respectively. Thus, if we have 3 drones the iterative form of the eq. (2) becomes:

$$\text{Step 1: } \mu_1 = \left\lceil \frac{(M - \mu_0)}{(N)} \right\rceil \quad (3)$$

$$\text{Step 2: } \mu_2 = \left\lceil \frac{(M - \mu_1 - \mu_0)}{(N-1)} \right\rceil \quad (4)$$

$$\text{Step 3: } \mu_3 = \left\lceil \frac{(M - \mu_2 - \mu_1 - \mu_0)}{(N-2)} \right\rceil \quad (5)$$

where μ_1, μ_2, and μ_3 denote the domain exploration workload areas: 1, 2 and 3, respectively. In the case of the scenario related to the chromosome shown in Table 3, we have N = 4 and M = 7. Therefore, from eq. (2) we obtain the domain searching workload areas as: $\mu_1 = 2$, $\mu_2 = 2$, $\mu_3 = 2$ and $\mu_4 = 1$.

We then perform a comparison between each given chromosome domain allocation encoding and the analytical balanced domain assignment from eq. (1) & (2). Eq. (6) is formulated to check the equivalency of the domain count allocated to the n^{th} drone with the set of j^{th} domain searching workload area count according to the chromosomes encoding that has been calculated by using eq. (1) and (2).

$$R_n \equiv \{\mu_1, \mu_2, \mu_3, \ldots \mu_j\}; \; n = \{1,2,3 \ldots N\}, j = \{1,2,3 \ldots N\} \quad (6)$$

Basically, we are conducting a logically equivalency test between eq. (1) and (2), i.e. a count of the areas allocated to each drone in the chromosome with a count of the balanced areas calculated theoretically.

For instance, from eq. (1) and (2), we obtain:
$R_1 = 1$, $R_2 = 2$, $R_3 = 2$, $R_4 = 2$ and
$\mu_1 = 2$, $\mu_2 = 2$, $\mu_3 = 2$, $\mu_4 = 1$.

Hence, from eq. (6) we get: $R_1 \equiv \{\mu_1, \mu_2, \mu_3, \mu_4\}$ = true because R_1 is matched or paired with μ_4. μ_4 is

then removed from the candidates set and the process repeats. Thus for the next iteration we compare $R_2 \equiv \{\mu_1,\mu_2,\mu_3\}$. A match is found with μ_1. Similarly, R_3 and R_4 are matched with μ_2 and μ_3, respectively.

Subsequently, in eq. (7) we check the logical AND with the output of the R_n comparison that has been calculated using eq. (6). If eq. (7) is logically true, we determine that the domain exploration workload, μ_j is fully matched among the drones according to the chromosome encoding and we will progress to eq. (8). Otherwise, the system will reject that chromosome encoding by assigning it the highest fitness (F_1) value, where a low score is preferable.

$$(R_1 \text{AND } R_2 \text{ AND } \ldots \text{AND } R_n) = \text{True} \quad (7)$$

Only if eq. (7) is true, then the total transit time between the domains for all the drones, T_β, is calculated as follows:

$$T_\beta = \sum_{i=1}^{N}(T_i), \quad (8)$$

where T_i indicates the transit time of the i^{th} drones. From eq. (8), the first fitness term, F_1 is:

$$F_1 = T_\beta \quad (9)$$

$$\text{Let, } A_\phi = \sum_{i=1}^{M} \text{size}(D_i), \quad (10)$$

where D_i, denotes the individual domain D_1, D_2, & D_m and A_ϕ total area of the domains and size is a method that returns the total domain's area.

From (8) and (10), the second fitness term, F_2 is:

$$F_2 = \left(\frac{A_\phi}{N}\right) + T_\beta, \quad (11)$$

where N represents the maximum number of drones.

A dynamic path finding algorithm A* is used to calculate the T_β for each drone. The A* algorithm calculates transit between any points within the two domains in order to obtain the shortest transit path between them.

The fitness term F_1 is used for drone searching work assignment in an obstacle and obstacle-free environment whereas F_2 is used to evaluate how the average search time varies in terms of the number of drones for the resultant best F_1 task allocation. The lowest F_1 value means the drone searching work allocation is adjacent for the total number of drones and the lowest F_2 value indicates the least average exploration time for that resultant best F_1 domain searching allocation.

3.3. GENETIC OPERATORS

Three genetic operators: selection, crossover and mutation are mainly used in GA. Initially; a random population is created to form a chromosome pool of n chromosomes. The fitness of each chromosome in the population is evaluated and unless the stopping criteria are met (i.e. a sufficiently superior solution has been found) the fit chromosomes are used to create new offspring chromosomes. There are several existing methods to select the best chromosomes, such as [21]. In this paper, rank selection is used with elitist selection to ensure good solutions are retained until better alternatives are found.

For the crossover, 'single point' crossover is used and for the mutation we just randomly alter allele values (i.e. x,y coordinates). In 'single point' crossover, one chromosome is selected to crossover with a second parent chromosome. For example, Table 4, shows the process before crossover. Here, two chromosomes (C_1, C_2) containing (X, Y) coordinates for four domains (D_1, D_2, D_3, and D_4) are shown. A new child chromosome C_{new} is created by performing single point crossover with chromosomes C_1 and C_2. The crossover point is randomly selected. In this instance it is from Domain 2 (D_2) onwards as shown in Table 5.

Table 4. The process of single point crossover (before)

	Coordinates	D_1	D_2	D_3	D_4
C_1	**X**	2	5	10	9
	Y	7	8	15	14
C_2	**X**	15	9	12	7
	Y	20	1	14	18

Table 5. The process of single point crossover (after)

	Coordinates	D_1	D_2	D_3	D_4
C_{new}	**X**	2	9	12	7
	Y	7	1	14	18

4. RESULTS AND ANALYSIS

In this section, some experiments are carried out to demonstrate the efficacy and feasibility of the proposed method for multi-robot exploration and work assignment in a known environment.

The purpose of the experiment shown in Fig. 4 is to determine how the coverage varies with the number of domains. In this section, the GA-based SBTP algorithm is used to achieve the efficient coverage. The given terrain (shown in Fig. 2) is partitioned into 10 domains (shown in Fig. 3). The simulation is run for 100 generations, 100 chromosomes, 10% elite, 30% kill, with the crossover and mutation rate of 50% and 30%, respectively. The simulation is repeated 5 times with computer producing different random number

generator seeds and the average percentage coverage is shown in Table 6.

Table 6. Coverage (%) versus number of domain(s) in an environment with and without obstacles

Number of domain(s)	Free space (%)	Obstacles Space (%)
D1	100	49.40
D2	100	71.80
D3	100	88.60
D4	100	99.40
D5	100	99.64
D6	100	99.68
D7	100	99.76
D8	100	99.86
D9	100	99.96
D10	100	100

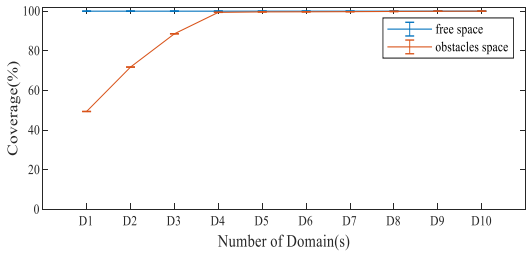


Figure 4 – Coverage (%) versus number of domain(s) in an environment with and without obstacles

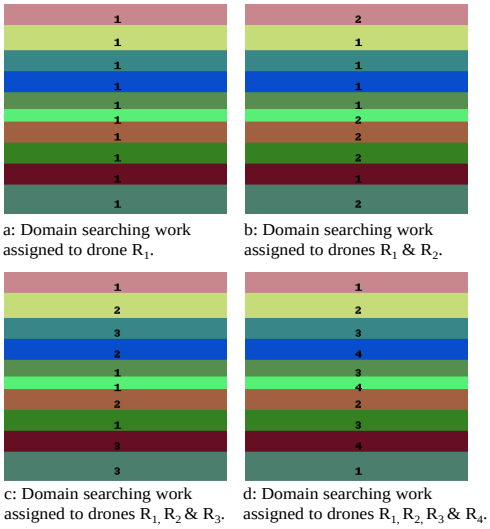
In Fig. 4, D1 to D10 represent the domains (1, 2…10) and the coverage is calculated using eq. (4) in Section II [19]. With 95% Confidence Intervals, we can see the exploration coverage gradually increases as the number of domains grow in a terrain that contains obstacles. However, in a free space environment (terrain does not contain obstacles), the exploration coverage is always 100% for the any number of domain(s) even in single generation of the chromosome population.

4.1. DRONE EXPLORATION WORK ASSIGNMENT IN ENVIRONMENT WITH AND WITHOUT OBSTACLES

In this subsection, a GA-based method is used to assign the domain searching work in a simple known environment with and without obstacles. When completing the task, the drones need to avoid colliding with obstacles. The given terrain (shown in Fig. 2a & 2b) is partitioned into 10 discrete domains (shown in Fig. 3a & 3b) and the maximum number of drones is 4. The simulation parameters for this experiment are: 200 generations, 100 chromosomes, 10% elite, 30% kill with a crossover and mutation rate of 50% & 30%, respectively. The results are shown in Figs. 5a, 5b, 6a and 6b.

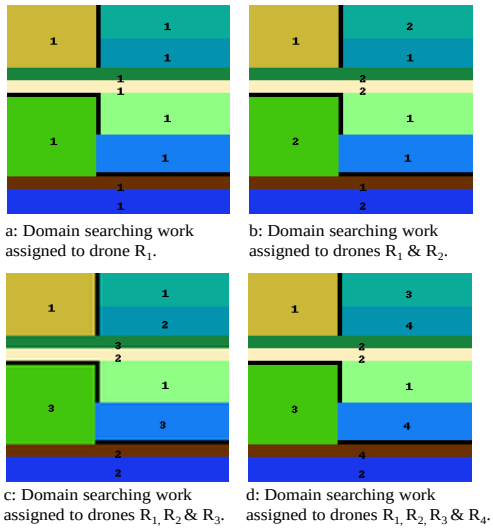
4.1.1. ROUGHLY EQUAL, NON-ADJACENT TASK ALLOCATION

In this subsection a GA-based mechanism explained in Section III [19] is used to allocate the domain exploration work among the drones. Here the task allocation is roughly equal, meaning the workload (x-y coordinates) is almost equal but not necessarily adjacent domains when assigned to specific drones. Details of the roughly equal task allocation are provided in Section III(B) [19]. The experiment is run for the partitioned floor-plan shown in Figs. 3a & 3b and the fitness performance is calculated using eq. (6) in Section III [19]. The A* dynamic path finding algorithm is used for drone inter-domain movements.



a: Domain searching work assigned to drone R_1.
b: Domain searching work assigned to drones R_1 & R_2.
c: Domain searching work assigned to drones R_1, R_2 & R_3.
d: Domain searching work assigned to drones R_1, R_2, R_3 & R_4.

Figure 5a – Roughly equal, non-adjacent domain searching work assignment among drones (R_1, R_2, R_3 & R_4) in a free space environment



a: Domain searching work assigned to drone R_1.
b: Domain searching work assigned to drones R_1 & R_2.
c: Domain searching work assigned to drones R_1, R_2 & R_3.
d: Domain searching work assigned to drones R_1, R_2, R_3 & R_4.

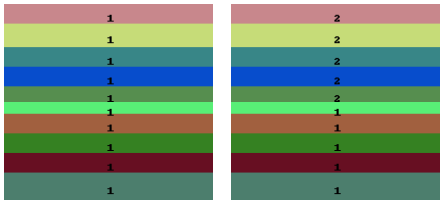
Figure 5b – Roughly equal, non-adjacent domain searching work assignment among drones (R_1, R_2, R_3 & R_4) in an environment containing obstacles (black areas)

Figs. 5a & 5b show the domain searching work assignment for different numbers of drones in a known simple environment with and without obstacles. The numbers (1-4) in Figs. 5a & 5b (a-d) denote drone 1 (R_1), drone 2 (R_2), drone 3 (R_3) & drone 4 (R_4), respectively. Basically this is an indication of which drone will search which domain. As we can see from Figs. 5a & 5b, the workload is roughly equal, meaning the area of the total partitioned domains are roughly equally distributed among the drones but they are not necessarily adjacent.

4.1.2. ADJACENT TASK ALLOCATION

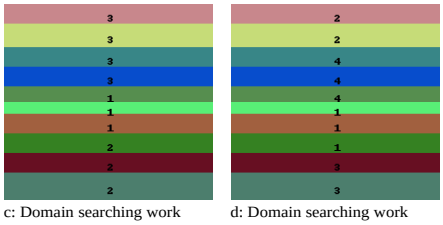
In this subsection a GA method (explained in Section 3) is used for domain searching work allocation to drones. Here the fitness function (explained in Subsection 3.2) is designed in such a way that the task allocation to drones comprises adjacent domain allocations that are balanced. Adjacent and balanced task allocation means that if there are 4 domains and 2 drones, then each drone will be allocated to explore the closest 2 domains that are adjacent to them. The experiment is run for the partitioned floor-plan shown in Figs. 3a & 3b and the fitness performance is calculated using eq. (9). The A* dynamic path finding algorithm is used for inter-domain drone movements.

Figs. 6a & 6b show the domain searching work assignment for different numbers of drones in an environment with and without obstacles. The numbers (1-4) denote drone 1 (R_1), drone 2 (R_2), drone 3 (R_3) & drone 4 (R_4), respectively, in Figs. 6a & 6b (a-d). As we can see from Figs. 6a and 6b the domain searching work assignment for each drone is adjacent and the domain allocation is balanced. However, as expected, the workload is slightly imbalanced when the number of domains is not wholly divisible by the number of drones.



a: Domain searching work assigned to drone R_1.

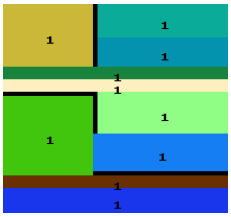
b: Domain searching work assigned to drones R_1 & R_2.



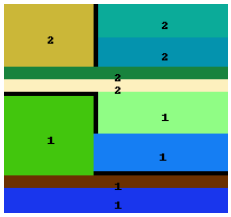
c: Domain searching work assigned to drones R_1, R_2 & R_3.

d: Domain searching work assigned to drones R_1, R_2, R_3 & R_4.

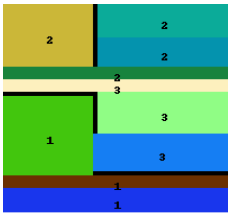
Figure 6a – Adjacent domain searching work assignment among drones (R_1, R_2, R_3 & R_4) in a free space environment



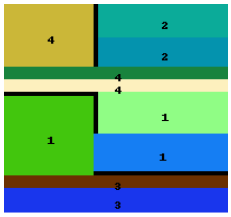
a: Domain searching work assigned to drone R_1.



b: Domain searching work assigned to drones R_1 & R_2.



c: Domain searching work assigned to drones R_1, R_2 & R_3.



d: Domain searching work assigned to drones R_1, R_2, R_3 & R_4.

Figure 6b – Adjacent domain searching work assignment among drones (R_1, R_2, R_3 & R_4) in an environment containing obstacles (black areas).

4.2. TASK ALLOCATION PERFORMANCE IN A FREE SPACE ENVIRONMENT

In this subsection, a performance comparison is conducted in an obstacle-free environment based on two task allocation methods: roughly equal, non-adjacent and adjacent task allocation, as shown in Figs. 5a & 6a. We can see from Fig. 5a, the workload is roughly equal, meaning the area of the total partitioned domain workload among the drones is distributed fairly equally but the allocation of the domains is not necessarily adjacent for each drone. On the other hand, we can see from Fig. 6a, the work assignment for each drone is adjacent and domain allocation is balanced as well. Suppose, if there are 8 domains and 2 drones; then the domain allocation for each drone will be the closest 4 domains that are adjacent to them. In addition, Table 7 gives a comparison of the average search time between the two domain searching allocation mechanisms shown in Fig. 5a in Subsection 4.1.1 & in Fig. 6a in Subsection 4.1.2 for different numbers of drones. In Fig. 7, we can see that the overall domain searching time is reduced when the number of drones increases.

Table 7. Comparison of average search time between the non-adjacent and adjacent task allocation for different numbers of drones in an obstacle-free environment

Number of drone(s)	Average search time (seconds)	
	Task allocation non-adjacent	Task allocation adjacent
1	2556.70	2556.70
2	1323.5	1295.20
3	917.60	869
4	709.70	652.10

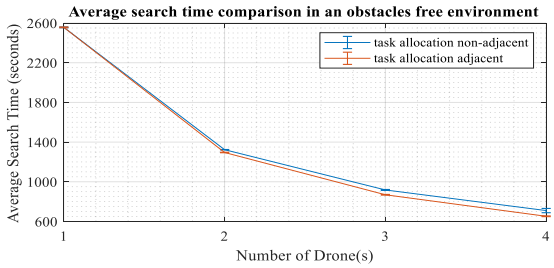


Figure 7 – Number of drones versus average search time in a free space environment. The experiment is repeated 10 times, based on the drone searching work allocation shown in Figs. 5a & 6a, with different random number generator seeds. The fitness performance is assessed using eq. (9) [19] and eq. (11) for the non-adjacent and adjacent task allocation respectively. With the 95% CIs, we can see the average domain exploration time is reduced as the number of drones increases, assuming, the exploration of 1 grid-square equals 1 time-unit (1second).

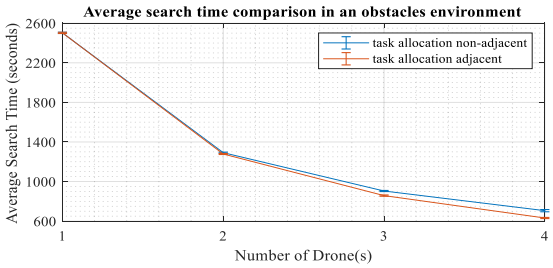


Figure 8 - Number of drones versus average search time in an environment with obstacles. The experiment is repeated 10 times, based on the drone searching work allocation shown in Figs. 5b & 6b, with different random number generator seeds. The fitness performance is evaluated using eq. (9) [19] and eq. (11) for the non-adjacent and adjacent task allocation respectively. With the 95% CIs, we can see the average domain exploration time is reduced as the number of drones increases, assuming, the exploration of 1 grid-square equals 1 time-unit (1second).

In Fig. 7, we can see when the task allocation is adjacent, the overall domain searching time is less than when the task allocation is non-adjacent for a given number of drones due to the reduced inter-domain transit time.

4.3. TASK ALLOCATION PERFORMANCE IN ENVIRONMENT WITH OBSTACLES

In this subsection, a performance comparison is conducted in an environment with obstacles based on two task allocation methods: roughly equal, non-adjacent and adjacent task allocation as shown in Figs. 5b & 6b. We can see from Figs. 5b and 6b, the workload is non-adjacent and adjacent for each drone, respectively. In addition, we can see from Fig. 5b & 6b, that 4 drones can complete the whole domain exploration task in approximately 705.70 and 631.50 seconds, respectively (shown in Table 8) without colliding with the obstacles.

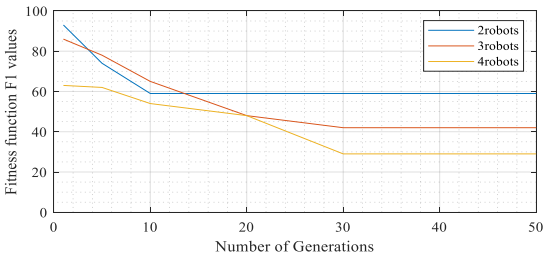
Table 8. Comparison of average search time between non-adjacent and adjacent task allocation for different numbers of drones in an environment with obstacles

Number of drone(s)	Average search time (seconds)	
	Task allocation non-adjacent	Task allocation adjacent
1	2502.70	2502.70
2	1291.80	1278.30
3	904.30	857.80
4	705.70	631.50

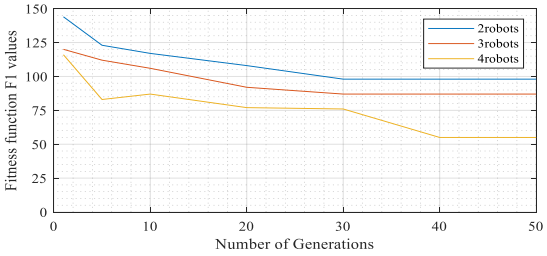
If we compare Figs. 5b & 6b we can see the drones can finish the domain exploration work in less time when task allocation is adjacent (shown in Fig. 8) among the drones (R_1, R_2, R_3 & R_4).

4.4. TASK ALLOCATION ALOGORITHM PERFORMANCE IN AN ENVIRONMENT WITH AND WITHOUT OBSTACLES

In this subsection, the performance of the GA method explained in Section 3, is tested. The experiment is run for 1, 5, 10, 20, 30, 40 and 50 generations with the simulation parameters; 50 chromosomes, 10% elite, 30% kill and the crossover and mutation rate of 50% and 30%, respectively. According to the fitness function shown in eq. (9), the best adjacent task allocation is found in a free space environment, as shown in Fig. 6a and an obstacles space shown in Fig. 6b. However, the F_1 values vary depending upon the drone inter-domain movement. This is due to where a drone is within one domain when it commences its transit to the next domain. In Fig. 9a, we can see that the best adjacent task assignment is found within 10 generations in an obstacle free environment for 2 robots and within 30 generations for 3 and 4 robots. On the other hand, in an environment with obstacles, as shown in Fig. 9b, we can see the finest solution is found for 2 robots in 30 generations. However, for 3 and 4 robots, the best solution is found within 30 and 40 generations, respectively. Overall, we can see, based on both scenarios, the GA converges to the best solution very quickly in a free space environment compared to one with obstacles environment.



a: Free space.



b: Obstacles.

Figure 9 - No. of generations vs fitness function (F1) in (a) Free space, and (b) Obstacles space. Here task allocation algorithm performance is checked based on drone searching work assignment shown in Figs. 6a & 6b. We can see the best solution is found within 10 to 40 generations in both scenarios; free space and obstacles space.

5. CONCLUSION AND FUTURE WORK

In this paper, a novel Square-Based Terrain Partitioning (SBTP) scheme employing a Genetic Algorithm (GA) is used to partition known environments into multiple sub-areas in a multi-robot exploration system where N robots are employed to survey a given area with some static obstacles. Also, a GA has been presented to solve Multi-Robot Task Assignment (MRTA) problems where one or more drones search multiple domains in such a way that for the given number of drones the overall search time is minimized. In applied GA, an appropriate fitness function is provided based on traditional GAs.

As shown in Section III [19] and Section 3 of this paper, we can see the terrain exploration coverage is increased for increasing number of domains. In addition, the utilization of each robot is increased by taking into account the transit time between domains within the F_2 fitness function, which can reduce the overall search time. However, the new fitness function explained in eq. (1-9) in Subsection 3.2 performs better than the fitness function described in eq. (5-6) in Section III(B) [19]. In both scenarios (terrain with and without obstacles), the overall domain searching time is effectively reduced when domain exploration work is adjacent compared to the non-adjacent work allocation for a given number of drones due to the reduced inter-domain transit time.

In the future work, the communication continuity constraint will be incorporated. This means that drones can no longer search independently but must regulate/coordinate their actions to remain in contact with each other, as shown in Fig. 10.

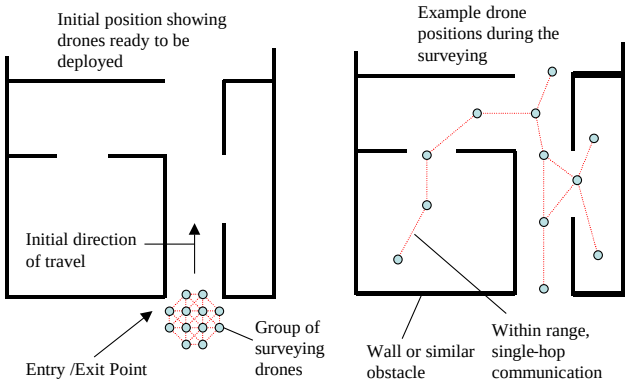


Figure 10 – An example snapshot of future work surveying operation.

A group of drones with sensors are sent into a hostile enclosed space typically comprising a number of rooms and corridors containing various obstacles. The drones must spread out to survey the terrain (In some instances the terrain is completely unknown; however, we initially focus on scenarios where a floor-plan is available) in a timely manner. However, they are required to remain in contact with each other to relay data and ensure the group functions as a single orchestrated unit. At various strategic points some of the drones may assume the role of a relay station whilst others actually perform the surveying. If the group becomes partitioned, remedial steps are taken to promptly re-establish contact.

6. REFERENCES

[1] S. Li, X. Xu, and L. Zuo, "Task assignment of multi-robot systems based on improved genetic algorithm," *Proceedings of the IEEE International Conference on Mechatronics and Automation*, Beijing, China, August 2-5, 2015, pp. 1430-1435.

[2] T. Nagarajan, and A. Thondiyath, "Heuristic based task allocation algorithm for multiple robots using agents," *Proceedings of the International Conference on Design and Manufacturing, IConDM*, 2013, pp. 844-853.

[3] M.-H. Kim, S.-P. Kim, and S. Lee, "Social-welfare based task allocation for multi-robot

systems with resource constraints," *Journal Computer and Industrial Engineering*, vol. 63, no. 4, pp. 994-1002, 2012.

[4] D. Zhang, G. Xie, J. Yu, and L. Wang, "Adaptive task assignment for multiple mobile robots via swarm intelligence approach," *Journal Robotics and Autonomous Systems*, vol. 55, no. 7, pp. 572-588, 2007.

[5] X. Chen, T. Tang, and L. Li, "Study on the real-time task assignment of multi-UCAV in uncertain environment based on genetic algorithm," *Proceedings of the IEEE 5th International Conference on Robotics, Automation and Mechatronics (RAM)*, 2011, pp. 73-76.

[6] I. Younas, F. Kamrani, C. Schulte, and R. Ayani, "Optimization of task assignment to collaborating agents," *Proceedings of the IEEE Symposium on Computational Intelligence in Scheduling (SCIS)*, 11-15 April, 2011, pp. 17-24.

[7] W. Sun, W. Hu, A. Lin, H. Tang, and W. Wu, "Multi-robot task assignment in obstacle environment," *Proceedings of the 36th Chinese Control Conference*, Dalian, China, July 26-28, 2017, pp. 6608-6613.

[8] R. Calvo, A.A. Constantino, and M. Figueiredo, "Individual distinguishing pheromone in a multi-robot system for a balanced partitioned surveillance task," *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 24-29 July, 2016, pp. 4346-4353.

[9] X. Bai, W. Yan, M. Cao, and J. Huang, "Task assignment for robots with limited communication," *Proceedings of the 36th Chinese Control Conference*, Dalian, China, July 26-28, 2017, pp. 6934-6939.

[10] T.S. Dahl, M. Mataric, and G.S. Sukhatme, "Multi-robot task allocation through vacancy chain scheduling," *Robotics and Autonomous Systems*, vol. 57, pp. 674-687, 2009.

[11] X. Bai, W. Yan, and M. Cao, "Clustering-based algorithms for multivehicle task assignment in a time-invariant drift field," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 2166-2173, October 2017.

[12] S. Jeon, M. Jang, D. Lee, Y.-J. Cho, and J. Lee, "Multiple robots task allocation for cleaning a large public space," Proceedings of the SAI Intelligent Conference, London, UK, November 10-11, 2015, pp. 315-319.

[13] N. Nedjah, R.M. de Mendonca, and L. de Macedo Mourelle, "PSO-based distributed algorithm for dynamic task allocation in a robotic swarm," *Journal International Conference on Computational Science (ICCS)*, vol. 51, pp. 326-335, 2015.

[14] W. Ling, G.M. Angel, P. Domenec, S. Albert, "Voronoi-based space partitioning for coordinated multi-robot exploration," *Journal of Physical Agents*, vol. 1, no. 1, pp. 37-44, 2007.

[15] U. Jain, R. Tiwari, S. Majumder, S. Sharma, "Multi robot area exploration using circle partitioning method," *Proceedings of the International Symposium on Robotics and Intelligent Sensors (IRIS'2012)*, 2012, vol. 41, pp. 383-387.

[16] Y. Elor and A.M. Bruckstain, "Multi-agent graph patrolling and partitioning," *Proceedings of the 2009 IEEE/WIC/ACM Int. Joint Conf. on Web Intelligence and Intelligent Agent Technologies*, 2009, pp. 52-57.

[17] C. Kato and T. Sugawara, "Decentralized area partitioning for a cooperative cleaning task," *Proceedings of the 16th International Conference on Principles and Practice of Multi-Agent Systems, PRIMA 2013, Dunedin, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8291, pp. 470-477, 2013.

[18] K. Yoneda, C. Kato, and T. Sugawara, "Autonomous learning of target decision strategies without communications for continuous coordinated cleaning tasks," *Proceedings of the IEEE/WIC/ACM Int. Conf. of Web Intelligence and Intelligent Agent Technologies*, 2013, pp. 216-223.

[19] Md.M.R. Mazumder, and C. Phillips, "Exploration of known environments with a multi-robot system using a genetic algorithm with a novel terrain partitioning algorithm," *Proceedings of the 9th IEEE International Conference on Intelligent Systems (IS)*, Funchal-Madeira, Portugal, September 25-27, 2018, pp. 371-378.

[20] K. Deb, "Genetic algorithm in search and optimization: the technique and applications," *Proceedings of the International Workshop on Soft Computing and Intelligent Systems*, 1997, pp. 58-87.

[21] http://www.obitko.com/tutorials/genetic-algorithms/index.php (last accessed on April 14, 2018).

Md Maksudur R. Mazumder *has been researcher of the Networks Research Lab of Artificial Intelligence at the School of Electronic Engineering & Computer Science (Queen Mary University of London) since 2015. He holds MSc degree from the Queen Mary University of London and BSc degree from the Stamford University Bangladesh. He has research interest in areas of Artificial Intelligence, UAV, Multi-robot exploration & Communication, Internet of Things (IoT), Vehicular Networks & Security, Cloud Computing, Network Security, Wireless Communications and Millimeter Wave Communications for 5G.*

Chris Phillips *received the B.Eng. degree in telecommunications engineering in 1987 and the Ph.D. degree in concurrent discrete event-driven simulation from the Queen Mary University of London, London, U.K. He then worked in industry for nine years as a Hardware and Systems Engineer with Bell Northern Research, Siemens Roke Manor Research, and Nortel Networks. In 2000, he returned to the Queen Mary. A common theme that underpins his research is how mechanisms can be developed to enable limited resources to be used effectively in different environments.*