Swarm behaviour for UAV systems, search and rescue tasks

Pilar Arques *, Fidel Aznar, and Mireia Sempere

Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
{arques,fidel,mireia}@dccia.ua.es

Abstract. UAV systems are positioning themselves as an efficient and competitive solution in different areas, both military and civilian. This paper explores the concept of swarm robotics where each individual assumes a behaviour but all the system should work together to achieve a common goal. Specifically, this work takes into account the concept of swarm robotics, to use it in unmanned systems for search and rescue tasks. In this paper we present a swarm behaviour that allows agents to search exhaustively in a wide area. This behaviour is very useful in search and rescue tasks.

Keywords: UAV, swarm, lifeguard, robustness, fault-tolerance

1 Introduction

UAV systems are positioning themselves as an efficient and competitive solution in different areas, both military and civilian. The use of these systems can help to minimize physical and material risks, as these systems require fewer resources in certain tasks compared to other conventional methods. For example, these systems can carry out a wide and exhaustive search in a defined area in an efficient way. In this paper, this task is presented as a swarm behaviour for UAV systems to be used in search and rescue tasks.

Nowadays, the definition of behaviours for UAV systems is a problem widely studied as can be seen in [3], [4], [5], [6]. Some of these studies, as [3] and [6], define a totally emergent behaviour; in other works, as [4], pheromones are used to guide agents; and others, as [5], focus on communication between agents. In [7] a revision of these studies is presented.

This paper describes a swarm behaviour where agents work in a collaborative manner to reach a specific resource, within a delimited area. This behaviour is inspired by a widely studied problem in swarm robotics, flocking behaviour. Reynolds, in [1], defines the basic rules of a flocking algorithm:

* This work has been carried out by the project "SISTEMAS DE ENJAMBRE INTELIGENTES DE VEHICULOS AEREOS NO TRIPULADOS PARA TAREAS DE SEGURIDAD Y VIGILANCIA" TIN2013-40982-R. Ministerio de Economía y Competitividad (Spain). Project co-financed with FEDER funds.

- Collision avoidance: avoid collision with nearby flockmates.
- Velocity matching: attempt to match velocity with nearby flockmates.
- Flock Centering: attempt to stay close to nearby flockmates.

These rules are also known as cohesion, separation and alignment. In our algorithm agents are positioned in flock formation. In order to achieve our goal, the algorithm defines three different roles for the agents of the system. The robot that set the velocity of the group is, in our case, the guide robot; line robot follows the agent detected by its front sensors; and, finally, other agents follow agents detected by their lateral sensors.

Guide robot starts the formation movement. Cohesion, separation and alignment are guaranteed because there is an agent that set the pace of the group. For example, when velocity is too high, side robots can be close to their maximum velocity, therefore, guide robot restarts the sequence and the movement starts again, in this way, the cohesion of the group is always maintained and collision avoidance is guaranteed.

This paper focuses on the application of this algorithm for search and rescue tasks after the sinking of a vessel. The shipwreck area can be known but tides and currents can sweep along and move away objects and possible survivors and difficult rescue tasks. Our swarm of UAVs is able to locate objects and persons and send their GPS position.

Drones need fewer resources than other conventional methods such as small planes, helicopters or boats, in addition, working in swarms with this algorithm can cover a wider search area in an exhaustive way.

Next section describes the microscopic modelling of the swarm, defining the behaviour of each agent. In section 3, the macroscopic behaviour of the swarm is defined, and also robustness and fault tolerance is demonstrated. In experimental section, scalability is proved. Finally, conclusions and future lines are presented.

2 Microscopic modelling

The main goal of this paper is to define a swarm behaviour for exhaustive searching in a wide area. As we commented before, this behaviour can be useful in search and rescue tasks, for example, in the sinking of a ship. The area of shipwreck can be known but tides and currents move away objects and persons.

Proposed algorithm has been designed taking into account its use in UAV agents, so that, we can assume that there is no possibility of collision with other objects in the environment. The agents of the swarm system have a ring of side sensors that allow them to preserve cohesion and to keep pace of the group; a lower sensor used to locate the search target; and a GPS locator and magnetometer so that we know their position and angle within the environment at all times.

2.1 Algorithm

Initially all the agents of the swarm have the same capacities and characteristics and in the first step of our algorithm a behaviour is assigned to each agent of the group.

Algorithm 1 shows the assignment of the behaviour to each individual agent of the group.

Algorithm 1: Assignment of the behaviour of each individual in the group:

```
1  for index =0.. max-Agents do
2  |  Assign a random index to each agent.
3  end
4  switch index do
5  |  case 0 Guide agent - BHVR(0)
6  |  case 1 Column agent - BHVR(1)
7  |  otherwise
8  |  |  if EVEN(index) then
9  |  |  |  Left side agent - BHVR(id)
10 |  |  else
11 |  |  |  Right side agent - BHVR(id)
12 |  |  end
13 |  end
14 endsw
```

2.2 Agents behaviour

As mentioned above, in this formation, each agent will have a predefined role that depends on the random number assigned to that agent. Next paragraph explains these behaviours.

Let $BHVR(id)$ the function which denote the behaviour of each agent respect to its assigned index.

Guide agent Its behaviour is to guide the swarm toward the target. So that, it has two possible behaviours, the first one performs spiral movements, and the second one that is used to guide the swarm to another search point inside the search environment. Algorithm 2 shows this behaviour. Where k is the maximum number of spirals to perform from a searching point, r is the position, ϵ an incremental constant for the position, α the steering angle, and θ an incremental constant for the angle, t elapsed time, $P_0(t)$ the point where agent 0 is at time t, V_{max} maximum speed for an agent, γ is a correction constant to limit the speed of the swarm in the turns and to preserve its cohesion, P_a is a random point in the searching environment, δ is a constant to limit the swarm speed in the navigation to the P_a point. The output of the algorithm is the point where the agent 0 will be at the instant $t + 1$.

Algorithm 2: BHVR(0): Guide agent behaviour

```
   input  : k, ϵ, θ, P_0 (t), V_max, γ, P_a, δ
 1 begin
 2   if number of spirals ≤ k then
 3     val r (t +1) ← r (t) + ϵ
 4     val α (t +1) ← α (t) + θ
 5     val V_0 (t +1) ← [ r (t) cos(α(t)) , r (t) sin(α(t)) ]
 6     val P_0 (t +1) ← P_0 (t) + V_0 (t +1)
 7     if  V_0 ≤ V_max + γ then  val r ← 0
 8   else
 9     val P_a ← r_a α_a
10     val D ← | P_0- P_a |
11     while  P_0 ≠ P_a do
12       val r (t +1) ← r (t) + D/ δ
13       val α(t +1) ← α (t)+D/ δ
14       val P_0 (t +1) ← [ r (t) cos(α(t)) , r (t) sin(α(t)) ]
15     end
16   end
17 end
   output: P_0 (t +1)
```

Column agent In algorithm 3, the behaviour of a column agent is shown. Where $Location(n, m)$ calculates sensor readings from angle n to m and returns the location of the nearest agent, in this case, n and m values correspond to the range of frontal sensors, and $P_1(t)$ is the location of the column agent in instant t. And the algorithm returns the location of the column agent in instant $t + 1$.

Algorithm 3: BHVR(1): Column agent behaviour

```
   input  : n,m, D_f, P_1 (t)
 1 begin
 2   val D_f ← location (n,m)
 3   val V_1 (t +1) ← D_f  −  P_1 (t)
 4   val P_1 (t +1) ← P_1 (t) + V_1 (t +1)
 5 end
   output: P_1 (t +1)
```

Side agents These behaviours are symmetric, agents with an even index will be on the right side of the column agent and agents with an odd index will be on the left side of it. In algorithm 4, the behaviour of agents which index is greater than 1 is described. In this algorithm, n and m values are referred to the range of side sensors, on right or left depending on the agent index, moreover

the steering angle of the nearest agent is needed. ρ is a constant to assure that the agent continues on the swarm direction. The output of this algorithm is the location of the side agent in instant $t + 1$.

Algorithm 4: BHVR(L): Side agent behaviour

input : n,m, ρ, $\boldsymbol{P}_i$ (t)
begin
  val $\boldsymbol{D}_l \leftarrow$ `location` (n,m)
  val $\beta \leftarrow$ `angle` (n,m)
  val $\boldsymbol{P}_i$ (t +1) $\leftarrow$ [$\boldsymbol{D}_l - \rho \cos(\beta + \pi/2)$, $\boldsymbol{D}_l - \rho \sin(\beta + \pi/2)$]
end
output: $\boldsymbol{P}_i$ (t +1)

Figure 1 shows the evolution of the proposal algorithm: the first image shows agents formation placed in the centre of the searching area; the second figure shows agents after having performed their first spiral movement; next images show several spiral movements to explain the behaviour that is being developed; in the last image, swarm has been moved to another searching point. In all images, the surface explored by agents has been highlighted, and dark blue colour shows points where guide agent has passed through.

3 Macroscopic behaviour

The macroscopic behaviour, as a result of performing microscopic behaviour, consists of a swarm of agents capable of navigating in formation, carrying out a spiral movement from the centre of search area. After a certain time interval and as long as the speed of side agents do not exceed the maximum speed, the swarm will perform a new spiral movement, until k times, ensuring that all the search area is fully and exhaustively covered. If the objective is not detected, the swarm is guided to another search point within the search area, this new point is randomly calculated, and spiral movement will be restarted.

We assume that the swarm is initially formed in the centre of the search area. If the goal is detected by an agent in one of the steps of the algorithm 5, our algorithm will stop.

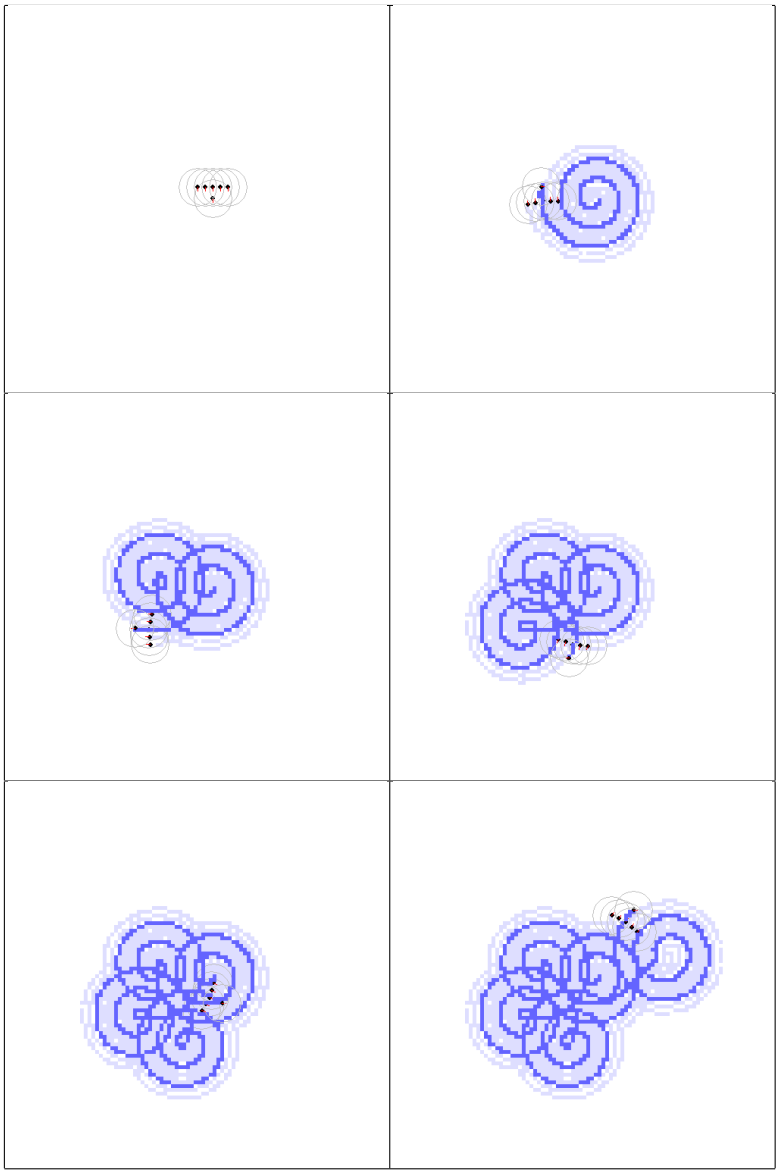

Fig. 1. Evolution of spiral sweep. Starting from the initial position and performing several spirals. In last picture the starting point has been moved to another point, and spiral motion restarts again.

Algorithm 5: Swarm behaviour

```
while Goal-Found == FALSE do
    Spiral-Movement
    if Speed-Passed then
        Spiral-Restarted
    end
    if number of spirals ≥ k then
        Random point-Selection
        Searching-Point-Movement
    end
end
```

Fault tolerance In swarm systems robustness and fault tolerance are needed in each step of the algorithm. A swarm system must ensure, at all times, that the loss of an agent does not affect the behaviour of the swarm and, therefore, it does not imply a failure to achieve the objective. In this case, all agents have the same capabilities and the behaviour of each one is determined by the random number assigned at he beginning of the algorithm. Therefore, in view of a possible loss of an agent of the swarm, to ensure the robustness of the algorithm, we only have to adjust the assigned index to each agent.

In our algorithm, the different possibilities are:

- Loss of a side agent. Whether it is a right or left side agent it is not an issue of particular relevance, the swarm automatically regroups and the only consequence is that the scanning surface decreases.
- Loss of the column agent. Supposing that the agent identified with value 1 is lost, the agent that has the ID 2 will be reassigned as 1, and the identifiers of even agents will be reordered, and agent with number 4 will have index 2, and so on.
- Loss of guide agent. Apparently, guide agent is the most important agent of the swarm, since this agent leads the swarm. If guide agent is lost, column agent (with index 1) will change to index 0 (guide agent), and other agents continue with their behaviours. In this case, the formation is different because there is not an agent ahead of the rest of the swarm. That is, because it is easier to continue the behaviour than change the behaviour of all the agents.

4 Experimentation

The experimentation of proposed behaviour has been carried out with MASON simulator [2]. Tests have consisted of the following:

- A random point, within the search area, is generated. That point will be our goal.
- Agents are located at the centre of the search area.
- Algorithm starts.
- Time needed to find the goal and covered area are measured.

We have compared the proposed behaviour with a random behaviour. In both cases, number of agents, the initial configuration and the initial location of agents are the same. Although it should be noted that the goal point is located in a different position in each simulation. This position is chosen randomly.

In figure 2, a comparison between both behaviours is shown. The measures obtained are:

- Proposed Behaviour: Iterations[1] needed to find the goal: 1346. Covered area: 1203m.
- Random Behaviour: Iterations needed to find the goal: 1909. Covered area: 3233m.

Thus, it is demonstrated that the stochastic behaviour of a random algorithm is less efficient than the proposed behaviour.

In order to prove the validity of the proposed behaviour, 50 tests for each type of behaviour, both the proposed and the random, have been carried out. These results

[1] 2 Iterations = 1 second

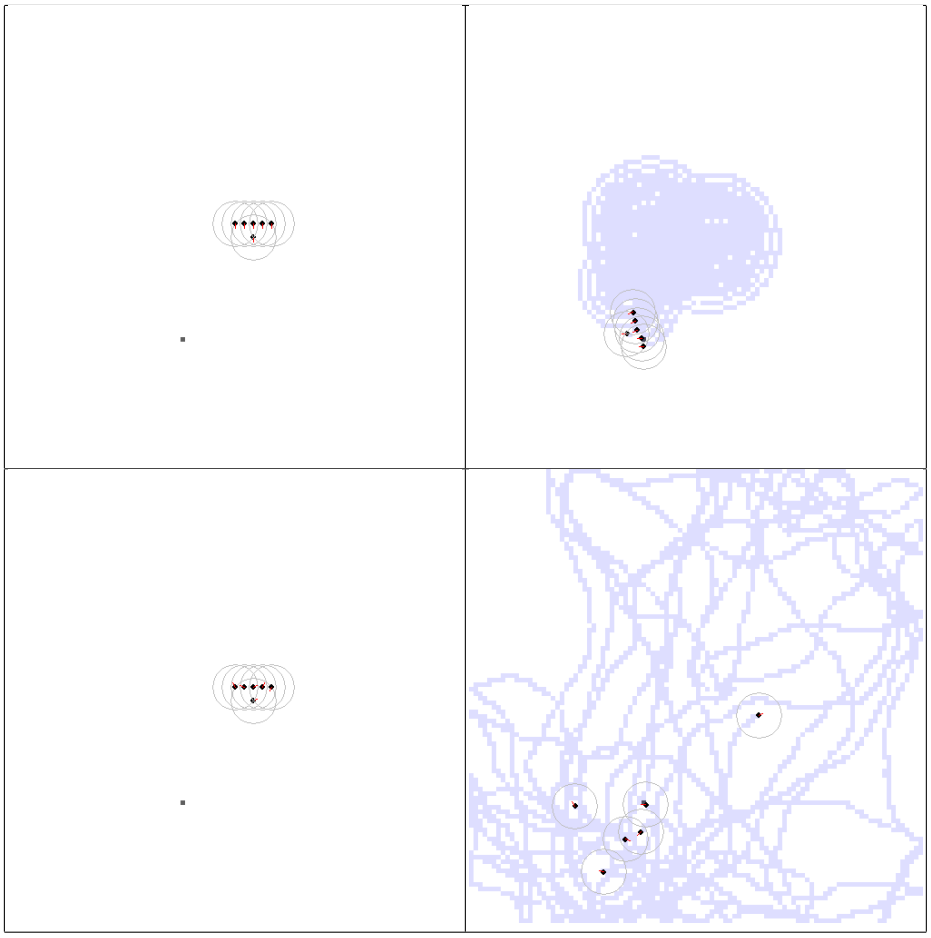

Fig. 2. Comparison of behaviours: images on the top show the proposed behaviour. Images on the bottom show the random behaviour.

are shown in Table 1. As can be seen statistical values obtained with the proposed algorithm are better than the values obtained with a random method.

In figure 3 the results of the tests have been grouped into 25 different categories depending on the number of iterations needed to achieve the goal. As can be seen, nearly 90% of test results obtained with our behaviour are grouped in the first 5 categories, that is, need fewer iterations, whereas 90% of test results obtained with a random behaviour are distributed along 9 categories, requiring, on average, a greater number of iterations to achieve the objective.

4.1 Algorithm scalability

Due to the nature of the described algorithm, scalability is simply to implement. It consists in adding more agents with side behaviour. Thus, the larger the number of

	Spiral	Random
Average	1629,38	3466,6
Variance	5.813.498	14.801.594
Median	935,5	2.516,5
Minimum	7	22
Maximum	13340	17080

Table 1. Statistical measures of 50 tests, comparing proposed behaviour versus a random behaviour.

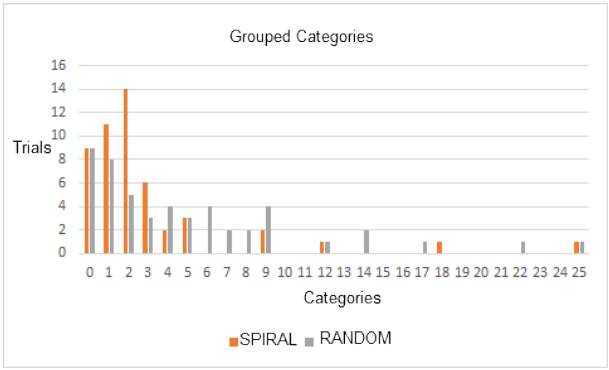

Fig. 3. Proposed algorithm vs Random. Grouped Categories

agents, the covered area will be greater. It is important to highlight that agents must be incorporated to the system in a balanced way. Furthermore, the maximum speed allowed to each agent must be considered, since the speed of the agents at the ends of the formation will necessarily be grater than central agents.

In table 2, the average and standard deviation from 10 tests performed with 8 agents and with 10 agents are shown. As can be seen the proposed algorithm improves its results respect average and standard deviation in the iterations needed to achieve the goal, whereas random behaviour remains with higher values.

	8 Agents		10 Agents	
	Spiral	Random	Spiral	Random
Average	949,7	3201,2	789,5	2066
Standard deviation	892,87	2331,16	671,32	2328,18

Table 2. Statistical values of 10 tests: proposed behaviour and random behaviour with 8 and 10 agents each one.

5 Conclusions and Future Work

In this paper we presented a swarm behaviour algorithm, decentralized, distributed, scalable and fault-tolerant, where all agents have the same initial capabilities, but each of them has a determined role inside the swarm behaviour. This algorithm could be adapted for UAV systems.

The defined behaviour performs spiral movements, which allows the system to carry out an exhaustive search in a wide area. It has been verified that the defined behaviour achieves the objective, furthermore, substantially in less time than a completely random behaviour.

It has been demonstrated that the algorithm is robust and fault-tolerant, where the loss of a component does not interfere with the main objective.

In addition we have studied scalability of the algorithm, which can be performed easily and efficiently.

As future work, we consider to ensure that a covered area will not be looked over again. If at first scan the goal is not reached, the new position must be outside the covered area. We also want to prove the behaviour using real UAV systems, searching a real object in a wide area.

References

1. Reynolds, C. W. Flocks, Herds, and Schools: A Distributes Behavioural Model. Computer Graphics, Volume 21, Number 4, (1987) 25-34
2. Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K.: A new multi-agent simulation toolkit. Proceedings of the 2004 SwarmFest Workshop (2004)
3. Michael A. Kovacina, Daniel Palmer, Guang Yang , Ravi Vaidyanathan Multi-Agent Control Algorithms for Chemical Cloud Detection and Mapping Using Unmanned Air Vehicles Proceedings of the 2002 IEEE International Conference on Intelligent Robots and Systems (IROS)
4. Bryan Walter, Adrian Sannier, Dirk Reiners, James Oliver UAV Swarm Control: Calculating Digital Pheromone Fields with the GPU Interservice/Industry Training, Simulation and Education Conference (IITSEC),Orlando, FL, 2005
5. A. Sivakumar and C. K. Y. Tan, UAV Swarm Coordination using Cooperative Control for establishing a Wire less Communications Backbone Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010), van der Hoek, Kaminka, Lesperance, Luck and Sen (eds.), May, 10-14, 2010, Toronto, Canada, pp. 1157-1164
6. Madey, Alexander G., and Gregory R. Madey. Design and evaluation of UAV swarm command and control strategies. Proceedings of the Agent-Directed Simulation Symposium. Society for Computer Simulation International, 2013.
7. Grzegorz Chmaj, Henry Selvaraj Distributed processing applications for UAV/drones: a survey Proceedings of the 23rd International Conference on Systems Engineering ICSEng 2014