Fast Formation of Swarm of UAVs in Congested Urban Environment

Sotirios Spanogianopoulos [*] **Qian Zhang** [**] **Sarah Spurgeon** [***]

[*] *University of Kent, School of Engineering and Digital Arts, UK (e-mail:ss976@kent.ac.uk)*
[**] *Liverpool John Moores University, Department of Electronics and Electrical Engineering, UK (e-mail:Q.Zhang@ljmu.ac.uk)*
[***] *University College London, Department of Electronic and Electrical Engineering, UK (e-mail:s.spurgeon@ucl.ac.uk)*

Abstract: As Unmanned Aerial Vehicles (UAVs) become more readily available and reduce in cost, using multiple UAVs simultaneously to accomplish a task becomes increasingly attractive. Once swarms of UAVs share the same workspace (operational environment) it is necessary to have a means to *rapidly* adopt an optimal collision-free formation in the workspace. A popular approach for formation of an optimal swarm is to use Particle Swarm Optimization (PSO) techniques. A variant of PSO was recently introduced called nPSO which claims to exhibit more rapid convergence than other variants. In this paper nPSO is applied to the problem of finding optimal positions of UAVs forming a swarm in presence of large obstacles such as buildings in an urban environment. The experiments show that no more than 1000 iterations are required to obtain near optimal formation of swarm of UAVs for different maps, including maps relating to congested environments.

Keywords: Fast collision-free swarm formation, Autonomous Systems, UAV, Optimization in swarm robotics.

1. INTRODUCTION

Application of aerial robotics are gaining increasing attention. For example, swarms of UAVs may be deployed for exploration or surveillance tasks in outdoor environments. Whenever a swarm of robots are used, it is necessary to consider the optimal flight formation and co-ordination across the swarm. In this paper, rapid optimal formation of collaborative aerial robots is considered so that as the task evolves, the algorithm will re-arrange the robot formation rapidly whilst accommodating new constraints.

Existing technologies are available that enable swarms of aerial robots to move in formation. For example, in Vásárhelyi et al. (2014) authors present a decentralized multi-copter flock that can autonomously navigate based on dynamic information received across a swarm consisting of up to 10 flying agents, each which has similar behaviour as animal swarms.

Other approaches have considered PSO for aerial robots. For example in Saska et al. (2016) the task of cooperative surveillance of preselected Areas of Interest (AoI) in outdoor environments by flocks of Micro Aerial Vehicles (MAVs) is considered. In this kind of surveillance mission, the basic task is to distribute the MAVs in the environment in order to successfully cover the AoIs and to identify the correct trajectories to reach the target location.

In Saska (2015) PSO is used to stabilise the navigation of swarms of MAVs in a given environment with obstacles. Controlling large numbers of aerial vehicles, without relying on any inter-vehicle communication, the method is based on visual localization which takes place from all MAVs, and estimates the position of every neighbouring robot in the flock.

Limitations on the computation power available on aerial robots has stimulated research considering how swarms of robots can be programmed dynamically in presence of very limited communication channels and computational functionality Morgan et al. (2016). The optimal assignment problem is solved using a distributed auction assignment and collision free trajectories are generated using sequential convex programming. Model Predictive Control (MPC) is used to solve the assignment and trajectory optimization (SATO) algorithm effectively moves a swarm of robots to a predefined shape in a distributed fashion.

Improving the Concurrent Assignment and Planning of Trajectories (CAPT) for a swarm of robots is considered in Turpin et al. (2014). Two challenges are considered: the complex problem of determining a suitable assignment for each robot in order to get to the target location and the creation of a collision-free and parameterized trajectory. This approach relies on a complete and centralized algorithm that produces collision-free and optimal solution to the CAPT problem. A restriction here is that the environment must be obstacle-free.

The circumstances under which the optimal true distance between aerial robots (forming a swarm) can be reached by providing an explicit and non-asymptotic model Yu et al. (2015). The work also considered the identification of the suboptimal strategies, in cases where the number of robots cannot be freely defined. A hierarchical communication method has been developed whereby the robots are able to communicate only with the robots that belong to the same hierarchically segmented region. This hierarchical method can help to obtain an optimal solution with constant approximations of true distance under specific assumptions.

There have been studies tailoring PSO to the problem of swarm of robots. For example in Adel and Songefeng (2016), a PSO is altered such that it relies on a uniform design (UD), i.e., the populations values don't change randomly. Thus a uniform design will cause an initial population of particles to be scattered uniformly over the search space. It has also been shown that with this design approach it is possible to decrease the fitness function and increase the performance compared to the standard PSO algorithm.

A new method capable of automatically determining if and where there are conflicts between a swarm of UAVs is proposed that lies in the category of Conflict Detection and Resolution (CDR) problems Alejo et al. (2013). Using an axis-aligned minimum bounding box (BB), the approach tries to resolve the conflicts one-at-a-time cooperatively by applying a collision-free trajectory planning algorithm. Then PSO refines the initial solution and alters the 4D trajectories of the UAVs using a global minimum cost.

An algorithm based on modifying the normal PSO algorithm using a random factor in order to reduce the expectation time for searching targets and removing them has been presented Cai et al. (2013). The task map is separated into specific sub-areas and each sub-area is further separated into distinct grids. The robots operate in searching phases or removing phases. When the target list of the robot does not include any detected targets, then the robot mode is changed to the searching phase else it will change its state to the removing phase. Every robot takes decisions individually by the local information it gathers and based on the communication it has with the neighbouring robots.

A new optimization algorithm based on PSO with a new 'momentum term' that is able to influence the convergence properties of the original PSO algorithm has been described Zhang and Mahfouf (2006). This new optimization method, called nPSO, can successfully deal with the problem of premature convergence. It is also capable of making particles' optimal search process adaptive by encouraging each particle to jump out of any local minima. The final version of this algorithm, called new Multi-objective PSO (nMPSO), along with nPSO was tested using a series of challenging benchmarks and it is shown that the algorithm performs better than the other optimization algorithms.

This paper applies nPSO and reports the results obtained by it with suggested parameters settings. Reasoning and techniques to improve convergence of a solution to the constrained problem are suggested. Further, the paper is organized as follows: Section 2 reviews and introduces the concept of nPSO, Section 3 describes the nPSO design and motivates the choise of a certain strategy to obtain rapid convergence, Section 4 demonstrates the results of the experiments conducted in 2D simulations and Section 5 concludes the paper.

2. REVIEW OF NPSO

Given the search space of the problem, the standard PSO adjusts the velocity on-the-fly of each particle based on the particle's behaviour as well as the behaviour of it's other companions. With tuning by the PSO algorithm the particles have a tendency to converge towards global optimum quickly. The nPSO algorithm is an improved version of PSO, where a momentum term is designed to deal with the problem of premature convergence. nPSO makes particle's optimal search process adaptive by encouraging each particle to jump out of any local minima.

2.1 Notation and Terms

The symbols and notations used for nPSO are as follows:

- D: The dimension of the particle
- N: Population size or the number of particles
- X_i**:** The ith particle in population
- x_{ij}: The jth element in the ith particle, i.e., $\mathbf{X_i} = [x_{i1}, x_{i2}, ..., x_{iD}]$
- V_i: The velocity vector of the ith particle X_i.
- v_{ij}: The jth element in the ith velocity vector, i.e., $\mathbf{V_i} = [x_{i1}, x_{i2}, ..., x_{iD}]$
- **F**: The fitness function. $F(X_i)$ generates a scalar value indicating the fitness of particle.
- P_i: The best previous position of the ith particle as determined by the fitness function F. So this variable stores the best behaviour of ith particle and hence it checks the local behaviour in search space.
- p_{ij}: The jth element in the P_i
- P_g: The particle with the best global fitness value found by the PSO algorithm
- p_{gj}: The jth element in P_g
- V_{max}: The upper limits of the achievable velocities of particles
- $v_{max,j}$: The velocity upper limit for the jth dimension
- P_t: The particle with the best fitness value among all the particles at the current generation t

Given iteration t, the next velocity $v_{ij}(t+1)$ of the jth element x_{ij} in Particle X_i is computed using the following equation:

$$\begin{aligned} v_{ij}(t+1) =& w_{ij} \times r_1(t+1) \times v_{max,j} \\ &+ v_{ij}(t) + c_1 \times r_2(t+1) \times (p_{ij} - x_{ij}) \\ &+ c_2 \times r_3(t+1) \times (p_{gj} - x_{ij}) \end{aligned} \quad (1)$$

, where c_1 and c_2 are positive acceleration co-efficients constants of algorithm PSO, and $r_1(t+1)$, $r_2(t+1)$ and $r_3(t+1)$ are uniformly distributed random variables ranging from $(0,1)$. w_{ij} is the momentum weight of the ith particle in the jth dimension.

Once the velocity V_i of particle X_i is computed for all its element using the above formula, the next position of particle is computed as:

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (2)$$

2.2 Algorithm Process

The momentum term of the Eguation 1 can provide particles adjustable momentum to realise a balance between exploration and exploitation in the optimization search process. If a particle converges to a solution, which is judged by whether the velocity of the particle is very small, the momentum weight is set at big values to encourage the particle to jump out from the local area. When the particle does not converge, the momentum weight is dynamically adjusted according to the particles search experience, i.e. if the particle cannot find a better solution in the previous generation(s), the momentum weight is decreased to enhance the local search ability, and otherwise the momentum weight is increased to enhance the global search ability. In details, nPSO is described using Algorithm 1.

Algorithm 1 The New Structure Particle Swarm Optimization (nPSO)

```
 1: Set acceleration coefficients $c_1, c_2$, momentum decreasing and increasing factors $m_1, m_2$, convergence judgement factor $\epsilon$, population size $N$ and max. Number of iterations $K_{max}$
 2: Generate Particles $X_i$ from $i = 1$ to $N$ with random values
 3: Set Momentum weights $w_{ij} = 1$ from $i$=1 to $N$, $j$= 1 to $D$
 4: for Generation $t = 1$ to $K_{max}$ do
 5:   for i = 1 to $N$ do
 6:     Compute fitness value of particle $X_i$ using fitness function $F$
 7:   end for
 8:   for i = 1 to $N$ do
 9:     if The fitness value of the best local position $P_i$ is worse than the new current fitness value of $X_i$ then
10:       $P_i = X_i$
11:     end if
12:   end for
13:   $P_t \leftarrow$ the particle with the best fitness value among all the particles at current generation $t$
14:   if The best global fitness value $P_g$ is worse than the fitness of $P_t$ then
15:     $P_g = P_t$
16:   end if
17:   if $P_g$ is the optimal solution found then
18:     Exit with solution $P_g$
19:   end if
20:   For all the elements of velocities of particles do the following:
```

$$w_{ij} = \begin{cases} 1, & \text{if } v_{ij} < \epsilon \times v_{max,j} \\ m_1 \times w_{ij} & \text{if no improvement in} \\ & \text{fitness for new particle } X_i \\ m_2 \times w_{ij} & \text{otherwise} \end{cases} \tag{3}$$

```
21:   Compute next velocity $V_i(t+1)$ of all the particles using Equation 1
22:   Ensure next velocity $V_i(t+1)$ for all particles does not exceed set maximum velocity $V_{max}$
23:   Compute next position $X_i(t+1)$ of all the particles using Equation 2
24: end for
```

3. APPLYING NPSO TO AERIAL ROBOTICS

nPSO appears to be very suitable for the UAVs allocation problems studied in this paper, where the presence of multiple UAVs and the environment with a large number of buildings has formed a complex optimization problem including multiple local optima.

It is assume that there is a communication *base station* at which the best location of swarms of UAVs is computed. The communication base station also has a geographical map of the environment with positions of all the buildings. A typical synthetic map is shown in Figure 1(a). For simplicity and to provide a more rapidly collision checking computation, the tall buildings or obstacles as an axis aligned bounding box, shown in grey.

3.1 Problem setup

Each UAV position is represented by two elements (x, y). The number of UAVs required to cover a rectangular area with no obstacles can be computed as $n-$ if A is the area of rectangular region that UAVs need to cover and a is the area of the maximum rectangular area that can be accommodated by the UAVs circular sensing region, then the least number of UAVs required to cover area A in presence of obstacles is given by:

$$n \geq \frac{A}{a} \tag{4}$$

So for n UAVs the configuration of a swarm of UAVs can be represented by $2n$ elements. The particle is designed as shown in the following equation.

$$X = [x_1, y_1, x_2, y_2, ..., x_n, y_n] \tag{5}$$

The particle is initialized with values (see Figure 1(b)) such that the overlap between UAVs is minimal in completely free space. Also the final UAV is randomly placed in the environment so as to minimise the chance of initialising the system at or close to a local optima.

3.2 Designing fitness function

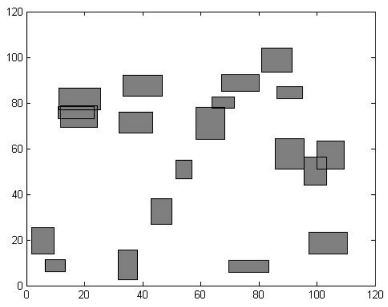
Computing the fitness function is a computationally time consuming process in nPSO algorithm. Even if the nPSO may try to converge within a few iterations, the fitness function design can dramatically affect the computational performance with respect to time.

Three parameters are used with different importance in deciding upon an appropriate fitness function:

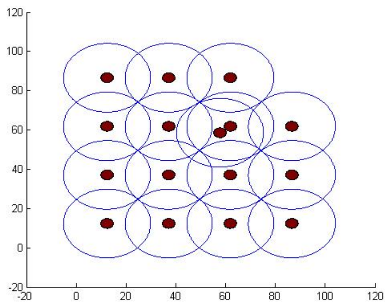
(1) Penetration extent of UAVs with axis aligned bounding boxes
(2) Sum of penetration extent of UAVs among themselves within some threshold
(3) Sum of penetration extent of all UAVs having an overlap with their sensing region

Standard intersection checking between UAVs (circle) and buildings (axis-aligned bounding box) is used for parameter 1. The penetration is computed using standard euclidean distance function for parameters 2 and 3. Matlab implements a fast euclidean distance computation mechanism computing in parallel on the arrays of elements. The fitness function is thus rapidly computed.

The threshold value in parameter 2 can be chosen by the user based on the least distance that needs to be maintained between UAVs within a neighbourhood. This



(a)



(b)

Fig. 1. (a) A synthetic map with axis aligned bounding boxes as obstacles shown in grey and (b) Initial formation of swarm of UAVs in free physical space

assists the convergence of the nPSO in the presence of obstacles as movement of a particle element is restrained by parameter 2.

4. EXPERIMENTS AND RESULTS

The nPSO code is implemented in Matlab-64bit version along-with the visualization code for environment with UAVs. The PC used for the simulations has a dual core i5-3230M processor running at 2.6GHz with 4GB RAM memory.

4.1 Environment Setup

This research assumes that the environment is a map of fixed obstacles such as buildings represented by axis aligned bounding boxes. Figure 1(a) shows an example of bounded environment ranging from [0 120] units along x-axis and [0 120] units along y-axis.

The map is generated by placing random axis aligned boxes with varied length and breadth ranging from [5 15] units. The gray coloured objects are axis-aligned bounding boxes that bound the buildings in an urban setting.

Each UAV is represented by a solid red circle with a diameter 5 units. The concentric dotted circle around a UAV represents the maximum region that can be sensed or viewed by that UAV in free space. This concentric circle diameter is 35 units. Note that we need at least 18 UAVs to fill the entire environment with given units.

4.2 Testing nPSO on different maps with increasing complexity

The main parameters of nPSO are set in Table 1. These parameters are fixed for all the experiments. Note that c_1 is purposely selected very small compared to c_2 to obtain fast convergence. A large C_2 encourages a particle to fly towards the global best particle found so far and a small C_1 makes the particle to fly towards its own historical best position found so far. Such a setting enables the particles to cooperate more with each other and concentrate the optimization search in one area rather than different local areas, which causes a faster convergence.

Also note m_1 is very small compared to m_2 so as to benefit the particle that experiences more positive behaviour. A large m_2 provides particles with more vigour, which will speed up the search process for most continuous optimization problems. Note that V_{max} value is mentioned in grid units per sec. A smaller value is chosen so that the initial UAV positions adopted do not change drastically and move to nearby spaces only. Such an assumption is practical when UAVs are distributed in real environment as the next optimal position change should be nearby to save energy and time of flight for UAVs. The significance of ϵ is to help the search process to progress if the particle's velocity by at least two decimal places. These factors were observed while performing experiments.

Table 1. Parameter settings of nPSO

Parameter	c_1	c_2	m_1	m_2	ϵ	V_{max}
Value	0.01	3.8	0.5	10	0.01	19

Table 2 contains the results of experiments conducted. The weights W_1, W_2 and W_3 are used for each parameters configuration of fitness function as orderly mentioned in section 3.2. Experiment 1 contains a small number of obstacles and as can be seen in Figure 2(a) the optimal solution is obtained very fast (within 408 iterations). We double the obstacles keeping the same number of UAVs and iteration rises to 747 for Experiment 2 (see Figure 2(b)).

The numbers of obstacles are kept same but the number of UAVs is increased in Experiment 3. Note the change in weight W_2 to give higher significance to keep UAVs away from each other at least maintaining some threshold distance. We make the environment further crowded with UAVs in Experiment 4 and we discover that iterations are nearly same as that of Experiment 3.

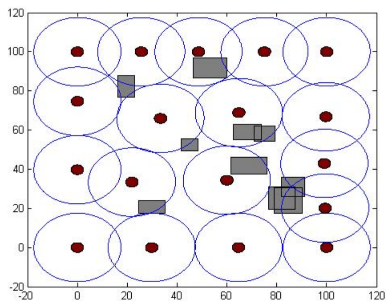
Experiments 5 and 6 tested nPSO in a large number of obstacles, that form a congested environment. And we notice from results that we obtain the nearly optimal solution within 1000 iterations (see Figure 2(e) and Figure 2(f)).

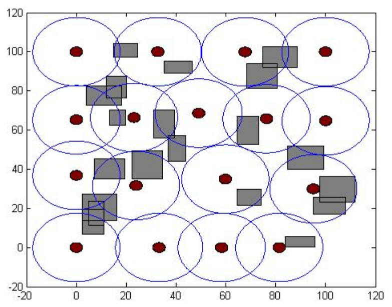
5. CONCLUSION

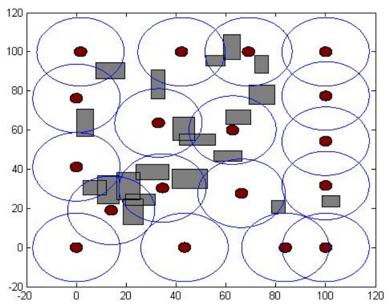
This paper addresses the basic problem of fast formation of swarm of UAVs using a recently developed algorithm Zhang and Mahfouf (2006)called New Structure Particle Swarm Optimization (nPSO). The nPSO algorithm was applied successfully to achieve optimal separation between UAVs in the presence of obstacles. The default settings of the algorithm were modified to suit the problem under

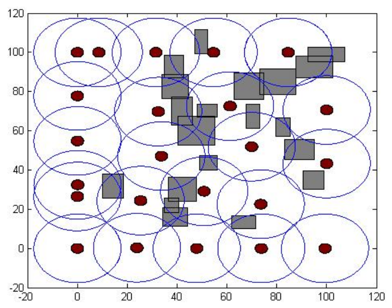
Table 2. Experiments and Results

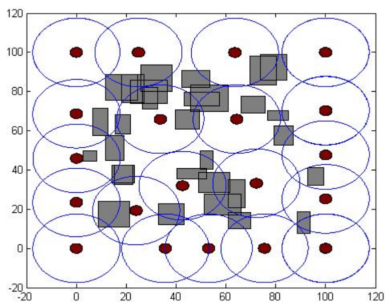
Exp. #	**# of iterations**	**Fitness Value**	# Obstacles	# UAVs	$[W_1, W_2, W_3]$
1	408	29.1272	10	18	[3, 0.175, 0.1025]
2	747	30.2558	20	18	[3, 0.175, 0.1025]
3	616	34.1902	20	18	[3, 0.375, 0.1025]
4	664	67.8993	20	25	[3, 0.375, 0.1025]
5	820	70.4825	30	25	[3, 0.375, 0.1025]
6	946	45.0974	40	18	[3, 0.375, 0.1025]


(a) Exp #1: Fitness value=29.1272


(b) Exp #2: Fitness value=30.2558


(c) Exp #3: Fitness value=34.1902


(d) Exp #4: Fitness value=67.8993


(e) Exp #5: Fitness value=70.4825

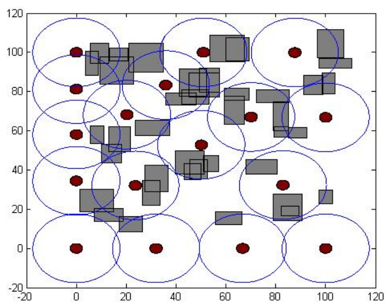

(f) Exp #6: Fitness value=45.0974

Fig. 2. Six experiments conducted to measure the performance of nPSO

consideration and the reasoning for the parameter settings adopted has been presented. In general, the experimental results presented have shown that nPSO results in faster convergence to nearly optimal solution with a straightforward implementation.

Future work will involve investigating repeated calling of nPSO to seek an optimal solution when the task constraints change incrementally from previous constraints. The case where a Manned Aerial Vehicle (MAV) with its own sensing region covers some area of environment independently of the UAVs will be considered. As the MAV moves continuously the neighbouring UAVs need to change their positions accordingly.

REFERENCES

Adel, H. and Songefeng, L. (2016). A particle swarm optimization algorithm based on uniform design. *The International Journal of Data Mining and Knowledge Management Process*, 6(1), 29–36.

Alejo, D., Cobano, J., Heredia, G., and Ollero, A. (2013). Particle swarm optimization for collision-free 4d trajectory planning in unmanned aerial vehicles. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, 298–307. IEEE.

Cai, Y., Chen, Z., Li, J., Li, Q., and Min, H. (2013). An adaptive particle swarm optimization algorithm for distributed search and collective cleanup in complex environment. *International Journal of Distributed Sensor Networks*, 2013.

Morgan, D., Subramanian, G.P., Chung, S.J., and Hadaegh, F.Y. (2016). Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming. *The International Journal of Robotics Research*, 0278364916632065.

Saska, M. (2015). Mav-swarms: Unmanned aerial vehicles stabilized along a given path using onboard relative localization. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, 894–903. IEEE.

Saska, M., Vonásek, V., Chudoba, J., Thomas, J., Loianno, G., and Kumar, V. (2016). Swarm distribution and deployment for cooperative surveillance by micro-aerial vehicles. *Journal of Intelligent & Robotic Systems*, 1–24.

Turpin, M., Michael, N., and Kumar, V. (2014). Capt: Concurrent assignment and planning of trajectories for multiple robots. *The International Journal of Robotics*

Research, 33(1), 98–112.

Vásárhelyi, G., Virágh, C., Somorjai, G., Tarcai, N., Szorenyi, T., Nepusz, T., and Vicsek, T. (2014). Outdoor flocking and formation flight with autonomous aerial robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, 3866–3873. IEEE.

Yu, J., Chung, S.J., and Voulgaris, P.G. (2015). Target assignment in robotic networks: Distance optimality guarantees and hierarchical strategies. *IEEE Transactions on Automatic Control*, 60(2), 327–341.

Zhang, Q. and Mahfouf, M. (2006). A new structure for particle swarm optimization (npso) applicable to single objective and multiobjective problems. In *IEEE International Conference on Intelligent Systems*, 176–181. IEEE.