2013 International Conference on Unmanned Aircraft Systems (ICUAS)
May 28-31, 2013, Grand Hyatt Atlanta, Atlanta, GA

Particle Swarm Optimization for collision-free 4D trajectory planning in Unmanned Aerial Vehicles

D. Alejo, J. A. Cobano, G. Heredia and A. Ollero
Robotics, Vision and Control Group
Engineering School, University of Seville
41092 Seville, Spain
[dalejo,jacobano,guiller,aollero]@cartuja.us.es

***Abstract*—This paper presents a new system which automatically identifies conflicts between multiple UAVs (Unmanned Aerial Vehicles) and proposes the most effective solution considering the available computation time. The system detects conflicts using an algorithm based on axis-aligned minimum bounding box and resolves them cooperatively using a collision-free trajectory planning algorithm based on a simple one-at-a-time strategy to quickly compute a feasible but non-optimal initial solution and a stochastic optimization technique named Particle Swarm Optimization (PSO) to improve the initial solution. PSO modifies the 4D trajectories of the UAVs with an overall minimum cost. Determining optimal trajectories with short time intervals during the execution of the mission is not feasible, hence an anytime approach using PSO is applied. This approach yields trajectories whose quality improves when available computation time increases. Thus, the method could be applied in real-time depending on the available computation time. The method has been validated with simulations in scenarios with multiple UAVs in a common workspace.**

I. INTRODUCTION

Cooperative missions with multiple UAVs present important advantages to carry out different missions such as surveillance, mapping, exploration, data collection, fire detection and monitoring, etc. [1]. In these missions the problem is to maintain as much as possible the trajectories planned and meet the Estimated Time of Arrival (ETA) of each UAV to perform coordinated missions. Therefore a system of collision-free 4D trajectory planning is required to ensure the safety of the mission.

General concepts and methods on path planning could be applied in order to solve the problem. A review of these methods with a comprehensive mathematical discussion is presented in [2]. Among these methods could be highlighted: potential fields [3], graph search like A* and D* [4] and Rapidly-exploring Random Trees (RRT) [5]. Other kind of methods have been proposed such as evolutionary techniques [6] [7] [8], particle swarm optimization [9] and multi-objective evolutionary algorithms [10].

The problem of trajectory planning is NP-hard [11] [12]. Some differential constraints given by the model of the UAV should be considered to make the problem tractable. Sampling-based techniques, as opposed to combinatorial planning, are usually preferred in these NP-hard problems. These planning schemes are appropiated when the solution space is hard to model or unknown a priori because of its dynamic nature.

Furthermore, planning optimal collision-free trajectories for multiple UAV leads to optimization problems with multiple local minimum in most cases and, thus, local optimization methods as gradient-based techniques are not well suited to solve it. The application of evolutionary techniques or particle swarm optimization is an efficient and effective alternative for this problem, since they are global optimization methods.

The Conflict Detection and Resolution (CDR) problem has also been studied extensively and different types of techniques have been proposed

978-1-4799-0817-2/13/$31.00 ©2013 IEEE

[13]. One method to resolve conflicts is based on the speed assignment [14]. In this method the speed profile for all the aerial vehicles involved in a collision is computed in a centralized way to solve the conflict. A method based on mixed-integer linear programming (MILP) is presented in [15]. It resolves the conflict by changing speed to a large number of aerial vehicles subject to velocity change constraints, but some conflicts cannot be solved. Other method resolves pairwise conflicts [16] but do not consider more UAVs. More methods based on MILP to avoid collisions are presented in [17] and [18]. The method for multiple-UAV conflict avoidance proposed in [19] assumes that UAVs fly at constant altitude with varying velocities and that conflicts are resolved in the horizontal plane using heading change, velocity change, or a combination thereof. Methods based on Ant Colony Optimization (ACO) algorithms have also been proposed [20]. In [21], the application of a game theory approach to airborne conflict resolution is presented. These techniques present a disadvantage: they are not well suited for applications that require a high level of scalability for the application to many UAVs.

The ARCAS FP7 European Project [22] is developing a cooperative free-flying robot system for assembly and structure construction. The ARCAS system will use aerial vehicles (helicopters and quadrotors) with multi-link manipulators for assembly tasks [23]. The aerial vehicles carry structure parts that will be assembled at the target destination. An important part in ARCAS is cooperative assembly planning and safe trajectory generation to perform the coordinated missions, assuring that neither the aerial vehicles nor the manipulators or the objects carried collide with each other.

This paper presents a system to plan collision-free 4D trajectories, which automatically identifies conflicts between multiple UAVs and proposes the most effective solution considering the available computation time. The detection algorithm is based on axis-aligned minimum bounding box. The cooperative 4D trajectory planning algorithm is based on a stochastic global optimization technique named Particle Swarm Optimization (PSO). Moreover, a simple one-at-a-time strategy is considered to quickly compute a feasible but non-optimal solution, which is taken as the initial point. The choice of PSO is based on its low computational overheads and faster solution convergence compared to genetic algorithms and other evolutionary algorithms [24]. Each UAV changes its trajectory to solve the detected conflicts collaborating with the rest of UAVs. The approach to solve the conflicts is to add an intermediate waypoint to the initial trajectory and/or change the speed to meet the ETA. This system presents two main advantages: its low execution time and its scalability.

On the other hand, computing time of most optimization methods is not deterministic, and most published works either do not consider computing time or simply show that in average it is much less than the available time to get a solution, thus wasting available computing time. The proposed system adopts an anytime approach to resolve the conflicts, it is able to provide a flyable solution at any time, and it will be more or less close to the optimum depending on the available time. Thus, valid trajectories whose quality improves when available computation time increases are yielded. Therefore, this system can be used with low available computation time although the quality of the solution will not be optimal.

The paper is organized into five sections. The formulation of the problem and requirements of the ARCAS project are presented in Section II. The proposed system is described in Section III. Section IV presents the simulations performed. Finally the conclusions are detailed in Section V.

II. Problem formulation

The problem considered in this paper is the collision-free 4D trajectory planning of multiple UAVs to perform the coordinated missions proposed by the ARCAS project (http://www.arcas-project.eu). Figure 1 shows the inputs and output considered in the corresponding task to perform in the project.

Firstly an initial assembly planning is generated to build the structure and spatial trajectories are computed to implement this plan. The deviations

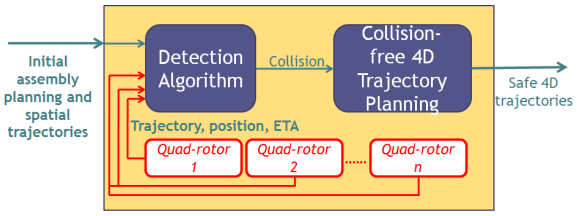

Figure 1. Cooperative assembly planning in the ARCAS project.

of aerial robots from these trajectories could be significant due to perturbations (i.e. wind). Furthermore, the planned trajectories for assembly are spatial trajectories. Then, the system should compute collision-free 4D trajectories when a conflict is detected during the execution of the mission. A trajectory is defined by a sequence of waypoints and the ETA to the last waypoint. UAVs share a common workspace and the separation between them should be greater than a given safety distance. It is assumed that heading and velocity changes are allowed to solve the conflicts. Therefore, the solution considers the addition of intermediate waypoints to the trajectory and changes of speed to meet the ETA. This latter is important to meet it in coordinated missions. The proposed system detects potential conflicts and computes a collision-free 4D trajectory for each UAV. All UAVs cooperate to solve the problem. The information that the system needs in order to solve the problem is the following:

1) Initial flight plan of each UAV.
2) Parameters of the model of each UAV
3) Location of each UAV
4) ETA of each UAV
5) The available computation time

The objective is to find collision-free 4D trajectories that minimize the probability of having a collision while minimizing the changes of waypoints and speed for each UAV. Moreover, the Estimated Time of Arrival (ETA) will be met to perform the mission of the ARCAS project.

III. Description of the System

This section describes the blocks of the proposed system to resolve the detected conflicts (see Figure 2). First, a detection algorithm based on axis-aligned minimum bounding box is implemented to detect possible conflict from all the 4D trajectories. Then, a collision-free 4D trajectory planning algorithm based on a fast initial feasible solution and PSO is implemented to solve the detected conflicts. The available computation time is considered and the quality of the solution improves when this time increases.

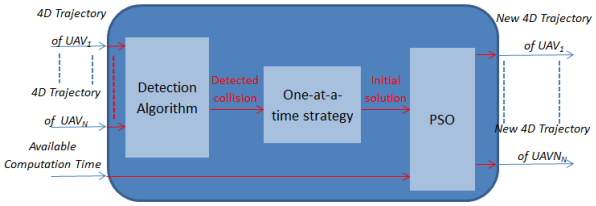

Figure 2. Description of the system.

A. Axis-aligned minimum bounding box for detection

The detection algorithm is based on axis-aligned minimum bounding box. This technique presents as advantages the low execution time and the need of few parameters to describe the system. On the other hand, it presents two disadvantages: it is not very accurate and it depends on the coordinate axes.

The ARCAS project uses aerial robots (autonomous quadrotors or helicopters) with a multilink arm attached to the bottom of the aerial robot to carry out the manipulation and assembly tasks. These UAVs should grasp different structural parts and transport them to the corresponding assembly location. To avoid collisions between the UAVs (including the arms and the grasped objects), a security envelope is defined around each one. Each UAV security envelope is approximated by two boxes joined together: a horizontal box that covers the aerial robot and its rotors and a vertical box, which surrounds the arm and the transported object (see Figure 3). Each box is defined by the intersection of three intervals, one by axis. The measurement of the horizontal box is related to the minimum horizontal separation between UAV considering the arm and the grasped object, and the vertical box is related to the vertical separation.

Therefore, the minimum separation, S, between two UAVs is defined by the dimension of both joined boxes. A collision is detected when there is an overlapping between the intervals that define each box (see Figure 3). Thus, the 3D problem is reduced to three problems of overlapping, one in each coordinate axis. Let us consider the intervals in one coordinate $A = [A_i, A_e]$ and $B = [B_i, B_e]$. The condition of overlapping for this coordinate is given by:

$$(A_e > B_i) \ \Lambda \ (A_i < B_e) \tag{1}$$

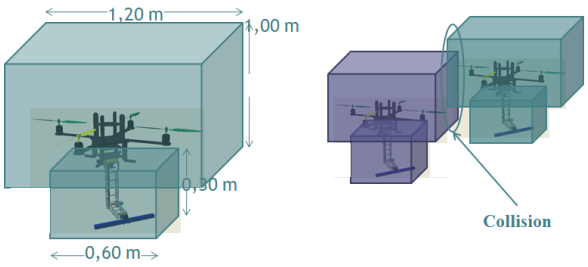

Figure 3. Detection algorithm based on axis-aligned minimum bounding box: A and B overlap (collision).

B. 4D trajectory planning algorithm

The collision-free 4D trajectory planning algorithm is based on obtaining a feasible initial solution very fast, and then optimizing the solution using the PSO method, which is an heuristic global optimization algorithm. PSO is iterative, and the solution improves with time. Thus, it is guaranteed that a feasible solution is available at any time, and that this solution will improve its quality if there is more execution time available.

The initial solution can be obtained with a simple one-at-a-time strategy: when there is a possible conflict between several UAVs, one of them moves to its destination point while the others stay hovering at the initial position, then the next UAV goes to its target position, and so on. By moving only one UAV at a time, the conflict is avoided. On the other hand, the solution is far from the optimum since the total time will be much higher than the ETA. Other velocity planning methods to quickly compute initial solutions have been also implemented using the two velocities and the greedy approach described in [25].

There are some situations in which the conflict cannot be solved only by changing the speeds of the involved UAVs. This is the case when a frontal collision is detected, for example. In these cases, the path must be changed and the roundabout technique can be a good candidate [26]. The main idea is to make the involved aircrafts circle the conflict in the same direction. The radius of the circle should be long enough to ensure collision-free trajectories and to provide flyable trajectories.

The PSO algorithm was first proposed in [27]. It is developed from swarm intelligence and is based on the research of bird and fish flock movement behavior. It works by maintaining a swarm of particles that move around in the search-space influenced by both the improvements discovered by the other particles (social behavior) and the improvements made by the particle so far (greedy behavior). Its main advantages are its simplicity, easy implementation and the existence of few parameters to tune when comparing with other evolutionary algorithms.

In this paper, we consider that the initial position is known and the final waypoint of each UAV should be the same as the one in the initial trajectory. Moreover, the ETA to the final waypoint should be met. Therefore, the goal of this algorithm is to obtain collision-free 4D trajectories by adding one intermediate waypoint in the trajectory of each UAV and changing the speed to meet the ETA while minimizing the following cost function:

$$J = \sum_{i=1}^{N} (L_i + k\Delta v_i^2) + \omega_c \tag{2}$$

where N is the total number of UAVs in the system, L_i is the total length of i^{th} trajectory, Δv_i is the change of speed of each UAV to meet its ETA, k is a factor to convert to distance, and ω_c is the collision penalty that will be added if the new trajectories still lead to collisions in the system or if at least one unfeasible plan is generated. This function can be easily modified in order to take into account energy analysis and other operational costs. Note that the approach considered is centralized.

The implemented algorithm is based on [28]. Let S be the number of particles in the swarm, each particle is defined by a state vector x_i in the search-space and a velocity vector v_i. This state vector contains the information about the location of the intermediate waypoint and the velocity in the first sector of the trajectory of each UAV. Note that the speed in the second sector is calculated so the ETA to the final waypoint is the same as in the original trajectory.

In first place, the swarm is initialized by randomly assigning initial locations and velocities with an uniform distribution. Then a special particle containing the initial solution is added to ensure the existence of one conflict-free solution at any time.

Let p_i be the best known state vector of particle i and let g be the best known state vector of the entire swarm. These are recalculated whenever a new iteration is obtained.

Then the exploration loop is executed. In each iteration, both the state vector and the velocity of each particle are updated by applying the expressions indicated in steps 10 and 11 (see Algorithm 1):

The most important parameters in this formula are the social weight, ϕ_g, and the local weight, ϕ_p. ω is the inertia weight. r_g and r_p are vectors where each component is generated at randomly with an uniform $U(0,1)$ distribution. Local and global best state vectors are also updated if necessary (steps 13-15).

The exploration loop can be finished by using many different termination criteria. Among these criteria a timeout condition and a convergence condition (most of the individuals lay in to a tight region of the search space) are the common approaches. In this paper, the algorithm concludes when the available computation time is reached.

The parameters ϕ_g and ϕ_p have been tuned by performing several tests with the same conditions and only changing one parameter at a time. These parameters are usually selected in the interval $[0,1]$. In our case, the best values found were $\phi_g = 0.9$ and $\phi_p = 0.1$.

Algorithm 1 Basic PSO algorithm

1. **for** Each particle **do**
2. Initialize each particle's state vector x_i with the desired probability function
3. Initialize particle best state vector $p_i \leftarrow x_i$
4. If $f(p_i) < f(g)$ update the swarm best state vector $g \leftarrow x_i$
5. Initialize each particle's velocity vector v_i. An uniform distribution is usually used.
6. **end for**
7. **repeat**
8. **for** Each particle **do**
9. Pick random numbers r_g r_p with $U(0,1)$
10. Update the particle's velocity: $v_i \leftarrow \omega v_i + \phi_p r_p(p_i - x_i) + \phi_g r_g(g - x_i)$
11. Update the particle's state vector: $x_i \leftarrow x_i + v_i$
12. **if** $f(x_i) < f(p_i)$ **then**
13. Update the particle's best known state vector
14. **if** $f(x_i) < f(g)$ **then**
15. Update the swarm's best known state vector $g \leftarrow x_i$
16. **end if**
17. **end if**
18. **end for**
19. **until** A termination criterion is met

IV. Simulations

A comprehensive set of simulations have been carried out to validate the proposed system. Also, a random generation process of the test considered has been performed to evaluate the system.

A. Test set design

The definition of a metric plays an important role to evaluate the results. In cases of motion planning problems for only one mobile object, there are some de facto benchmark standards in the academic context, like the bug trap or the alpha test [29]. However, this is not the case when dealing with planning of multiple mobile objects.

In the work presented in this paper, a test set has been developed in a given scenario to validate the ASCDR system and to provide a way to measure the properties of the system regarding time of execution, optimization and level of scalability with number of vehicles. Furthermore, the test set and the design methodology can be useful for comparison with other methods.

The considered scenario has a base of 10x10 dimensional units. Different problems are defined by considering a random generation process. Each problem is formulated as a set of entry and exit points located in one of the lateral sides of the scenario.

The adopted strategy is regressive. Random candidate solutions are generated and the problem is defined using them when they are found.

The random generation process of the tests is performed following the Algorithm 2. For each vehicle, an entry side is randomly chosen, selecting a uniformly random number between 1 and 4 (line 4). Then, the exit side is randomly selected from the resting 3 sides (line 5). Entry and exit points are randomly selected from the corresponding side (line 6). A certain number, M, of intermediate waypoints inside of the scenario along with the entry and exit points define the initial trajectory.

Algorithm 2 Random test generation algorithm

```
1:  for each test do
2:    while test is not valid do
3:      for each aircraft do
4:        Choose a random entry side
5:        Choose a random exit side from the
          resting 3
6:        Choose entry and exit points from the
          corresponding entry and exit sides
7:        Add M random intermediate way-
          points
8:        Check for the flight plan validity
9:      end for
10:   end while
11: end for
```

The algorithm should ensure the following (see line 8):

1) The solution is valid, i.e. vehicles do not collide
2) The initial trajectories should ensure that generating a conflict

The test set consists of 40,000 different problems grouped by the number of vehicles involved, from 2 to 5, in subsets of 10,000 tests. This classification, using the number of vehicles, is useful to study the scalability characteristics of the method. The number of vehicles is from 2 to 5 because is the number used in the experiments of ARCAS project.

B. Simulation results

Simulations have been carried out from the test set to check the properties of the proposed system. The tests have been performed in the same computer and under the same conditions.

The algorithms have been run in a PC with a 2GHz Dual Core processor and 2 GB of RAM. The operating system used was Kubuntu Linux with kernel 2.6.32. The code was written in C++ language and compiled with the gcc-4.4. 1.

The minimum horizontal and vertical separation between UAVs is shown in Figure 3. These separation distance have to be calculated according to the dimension of each vehicle and their localization and control errors. The dimensions of the scenario are $10m \times 10m$. Two hundred tests have been performed for each subset. The number of intermediate waypoints, M, is set to 1. Therefore, each solution trajectory is composed of two segments. Each UAV should meet its ETA to perform the coordinated mission. The allowed speed for each UAV is between $0.3m/s$ and $2m/s$, and $k = 5$. The speed in the first segment is chosen randomly and in the second one is computed to meet the ETA. If a particle does not meet the ETA, it is penalized.

Table I shows the mean time of execution and the mean minimum cost considering two hundred tests and one hundred iterations in each test. The relation between the time of execution and the number of iterations performed is shown in Figure 4. The dependence with the number of UAVs shows the scalability characteristics of the method.

Table I
MEAN AND STANDARD DEVIATION OF THE TIME OF EXECUTION AND THE COST CONSIDERING 200 SIMULATIONS FOR EACH NUMBER OF VEHICLES.

UAVs	Mean Time (s)	Mean Cost (m)
2	13.843±2.104	19.935±2.907
3	23.603±3.385	28.082±4.244
4	32.599±4.691	37.092±4.934
5	38.925±7.231	46.063±6.467

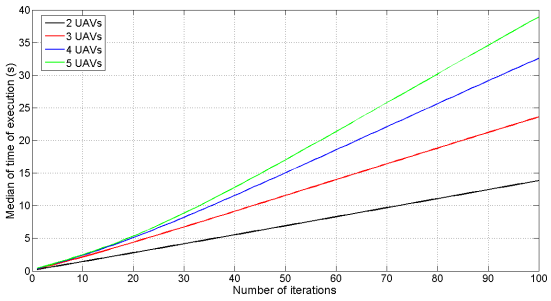

Figure 4. Time of execution vs. number of iterations depending on the number of vehicles.

Figure 5 shows the evolution of the median of the minimum cost with the number of iterations considering different number of UAVs. The proposed system finds a better solution as time passes.

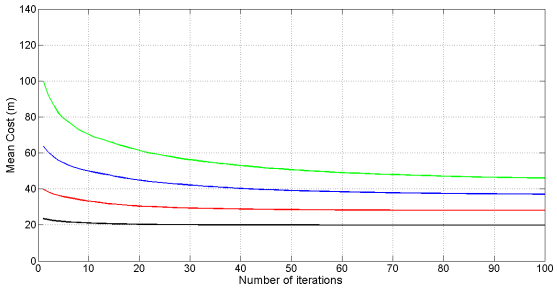

Figure 5. Median of minimum cost throughout successive iterations.

The speed should be changed in order to meet the ETA of each UAV. Figures 6, 7, 8 and 9 show the information on the speed of each UAV involved. Each box of the figure depicts statistics of the two hundred tests performed for a given number of UAVs. The central mark is the median, the edges of each box are the 25th and 75th percentiles, and the whiskers extend to the extreme data points.

Note that the mean change of speed of each UAV is low and it increases as the number of UAVs increases. Moreover, the mean change of speed of each UAV involved in a test is similar.

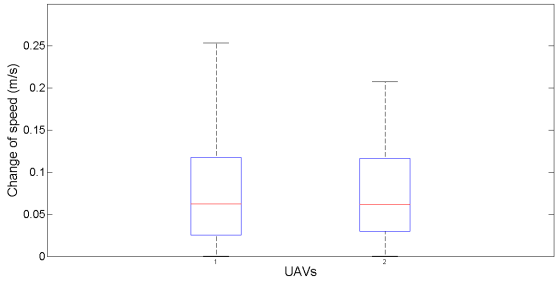

Figure 6. Speeds with two UAVs by considering 200 tests.

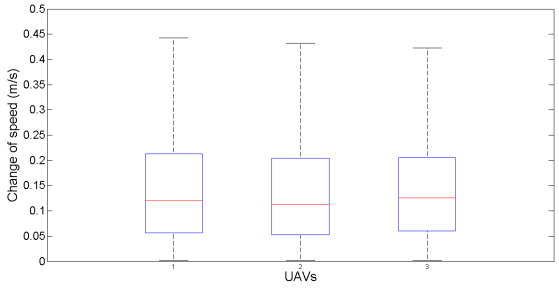

Figure 7. Speeds with three UAVs by considering 200 tests.

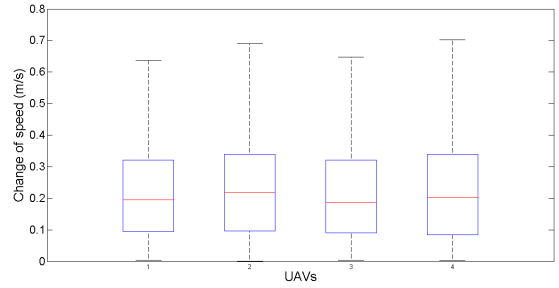

Figure 8. Speeds with four UAVs by considering 200 tests.

The median of the minimum costs computed in all the tests has been chosen as statistical indicator. This indicator indicates how much time it would cost to achieve a solution with certain level of optimality. This relates the cost in a given iteration to the obtained minimum cost in the corresponding problem.

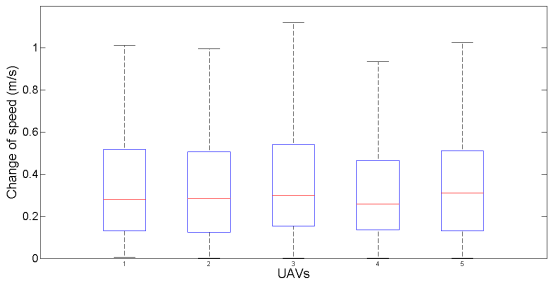

Figure 9. Speeds with five UAVs by considering 200 tests.

Figure 10 shows a normalization of the cost against the number of iterations. A line that marks the required number of iterations to compute for a 90% level of optimality is drawn. If the test set is executed in the same computer where the user has installed the proposed method, Figure 4 will provide an estimation of the time needed for that number of iterations, and therefore, that level of optimality.

For the cost normalization, a linear transformation, $f(x) = ax + b$, is applied to the actual cost values to set them in the range [0,1]. Therefore a and b are chosen in such a way that the maximum cost equals to 1 and the minimum cost equals to 0. Therefore,

$$a = \frac{1}{Cost_{max} - Cost_{min}} \tag{3}$$

$$b = \frac{Cost_{min}}{Cost_{min} - Cost_{max}} \tag{4}$$

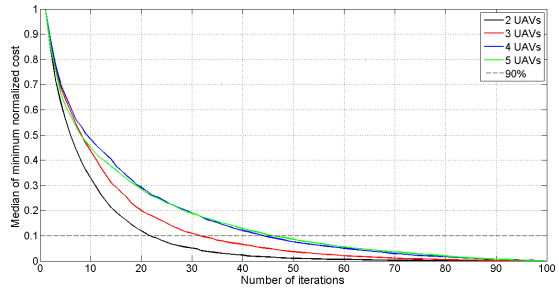

Figure 10. Normalized cost throughout successive iterations. The line marks the 90% optimality.

Depending on the number of UAVs, a solution of great quality, 90%, is computed between 20 or 47 iterations. This means that this kind of solutions can be computed between three and sixteen seconds by depending on the number of UAVs. This characteristic is important to apply this collision-free 4D trajectory planning algorithm in real-time applications. Moreover, the algorithm based on PSO presents better results than the algorithm based on genetic algorithms presented in [30].

Next, an anytime approach is considered. Figures 11, 12, 13 and 14 show how the quality of the solution improves as the time increases by using PSO. Two hundred simulations are considered for each number of UAVs. The quality of the solution is reported each ten iterations. The advantage of this approach can be noted because a good quality of solution of 75% is achieved on two or six seconds for 2 and 4 UAVs, and on eight seconds for 5 UAVs. Obviously, the quality of solution improves as the time increases.

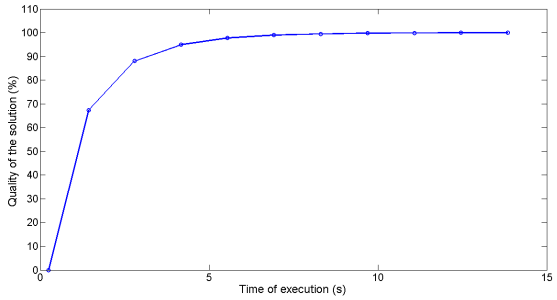

Figure 11. Anytime approach with two UAVs.

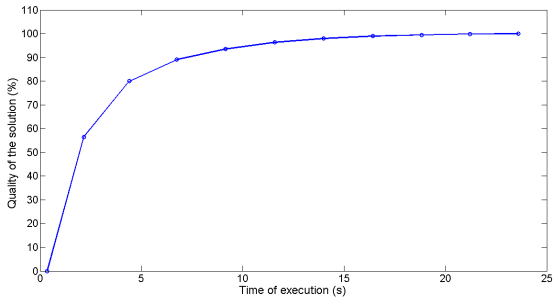

Figure 12. Anytime approach with three UAVs.

V. Conclusions

In this paper, we have presented a system to plan collision-free 4D trajectories based on an anytime

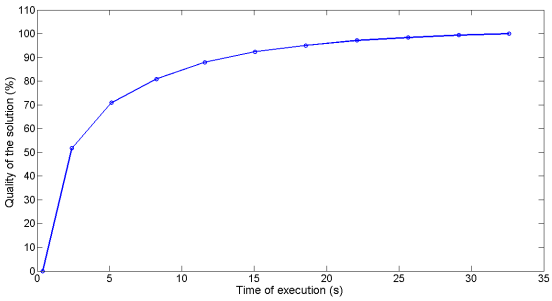

Figure 13. Anytime approach with four UAVs.

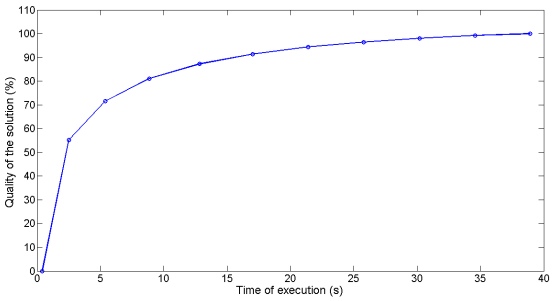

Figure 14. Anytime approach with five UAVs.

stochastic optimization approach. The proposed system detects conflicts in trajectories of multiple UAVs using an algorithm based on axis-aligned minimum bounding box, and resolves them cooperatively using a collision-free 4D trajectory planning algorithm based on a simple one-at-a-time strategy to quickly compute a feasible but non-optimal initial solution and Particle Swarm Optimization to improve incrementally the initial solution. The system provides a valid collision-free set of 4D trajectories at any time, so it can be used with low available computation times, although a sub-optimal solution will be obtained. Thus, the proposed system is well suited to situations with variable available computation times, depending on the number of UAVs and the distance to potential conflicts.

Other requirement to meet in coordinated missions is the ETA. The proposed system computes solution trajectories meeting the ETA of each to perform the mission.

The system has been validated with many simulations performed in different scenarios and several studies to analyze the characteristics of the system.

The main advantages of the proposed system with respect to the one presented in [9] are: it considers mulitple vehicles in the space and could be applied in real-time.

Future work will include the validation of the proposed system experimentally with UAVs.

VI. Acknowledgement

This work was supported by the European Commission FP7 ICT Programme under the project ARCAS 287617 and the CLEAR project (DPI2011-28937-C02-01) funded by the Spanish Research and Development Program. Also it has been partially funded by the Junta de Andalucia project P09-TEP- 5120. David Alejo has been granted with a FPU fellowship by the Ministerio de Educación y Ciencia of the Spanish Government.

References

[1] J. A. Cobano, J. R. Martínez-de Dios, R. Conde, J. M. Sánchez-Matamoros, and A. Ollero, "Data retrieving from heterogeneous wireless sensor network nodes using uavs," *Journal of Intelligent and Robotic Systems*, vol. 60, no. 1, pp. 133–151, 2010. [Online]. Available: http://www.springerlink.com/index/10.1007/s10846-010-9414-y

[2] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006, available at http://planning.cs.uiuc.edu/.

[3] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Rob. Res.*, vol. 5, no. 1, pp. 90–98, Apr. 1986. [Online]. Available: http://dx.doi.org/10.1177/027836498600500106

[4] A. Stentz and I. C. Mellon, "Optimal and Efficient Path Planning for Unknown and Dynamic Environments," *International Journal of Robotics and Automation*, vol. 10, pp. 89–100, 1993.

[5] S. M. Lavalle, J. J. Kuffner, and Jr., "Rapidly-Exploring Random Trees: Progress and Prospects," in *Algorithmic and Computational Robotics: New Directions*, 2000, pp. 293–308.

[6] J. A. Cobano, R. Conde, D. Alejo, and A. Ollero, "Path planning based on genetic algorithms and the monte-carlo method to avoid aerial vehicle collisions under uncertainties," in *Proc. IEEE Int Robotics and Automation (ICRA) Conf*, 2011, pp. 4429–4434.

[7] R. Vivona, D. Karr, and D.Roscoe, "Pattern-based genetic algorithm for airborne conflict resolution," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, Keystone, Colorado, August 2006.

[8] N. Durand and J. Alliot, "Optimization resolution of en route conflicts," in *First USA/Europe Air Traffic Management Research and Development Seminar (ATM1997)*, Saclay (France), 17-19 June 1997.

[9] M. Pontani and B. A. Conway, "Particle Swarm Optimization Applied to Space Trajectories," *Journal of Guidance Control and Dynamics*, vol. 33, pp. 1429–1441, 2010.

[10] A. J. Pohl and G. B. Lamont, "Multi-objective UAV mission planning using evolutionary computation," in *Winter Simulation Conference*, 2008, pp. 1268–1279. [Online]. Available: http://dx.doi.org/10.1109/WSC.2008.4736199

[11] J. F. Gilmore, "Autonomous vehicle planning analysis methodology," in *AIAAA Guidance Navigation Control Conference*, 1991, pp. 2000–4370.

[12] R. J. Szczerba, "Threat netting for real-time, intelligent route planners," in *IEEE Symp. Inf., Decis. Control*, 1999, pp. 377–382.

[13] J. K. Kuchar and L. C. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, pp. 179–189, 2000.

[14] J. A. Cobano, D. Alejo, A. Ollero, and A. Viguria, "Efficient conflict resolution method in air traffic management based on the speed assignment," in *Proceedings of the 2nd International Conference on Application and Theory of Automation in Command and Control Systems*, ser. ATACCS '12. Toulouse, France, France: IRIT Press, 2012, pp. 54–61. [Online]. Available: http://dl.acm.org/citation.cfm?id=2325676.2325684

[15] A. Vela, S. Solak, W. Singhose, and J.-P. Clarke, "A mixed integer program for flight-level assignment and speed control for conflict resolution," in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, dec. 2009, pp. 5219 –5226.

[16] H. Erzberger, "Automated conflict resolution for air traffic control," in *Proceeding International Congress Aeronautical Sciences*, 2006, pp. 179–189.

[17] A. Richards and J. P. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," in *In Proc. ACC*, 2002, pp. 1936–1941.

[18] L. Pallottino, E. Feron, and A. Bicchi, "Conflict resolution problems for air traffic management systems solved with mixed integer programming," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 3, no. 1, pp. 3 –11, mar 2002.

[19] I. Hwang, C. J. Tomlin, I. Hwang, and C. Tomlin, "C.: Protocol-based conflict resolution for air traffic control. air traffic control quarterly 15(1," 2007.

[20] N. Durand and J. Alliot, "Ant colony optimization for air traffic conflict resolution," in *Proceedings of the Eighth USA/Europe Air Traffic Management Research and Development Seminar (ATM2009)*, Napa, (CA, USA), 2009.

[21] P. Masci and A. Tedeschi, "Modelling and evaluation of a game-theory approach for airborne conflict resolution in omnet++," in *Second International Conference on Dependability*, June 2009.

[22] A. Ollero, "Aerial robotics cooperative assembly system (arcas): First results," in *Aerial Physically Acting Robots (AIRPHARO) workshop, IROS 2012*, Vilamoura, Portugal, October 7-12 2012.

[23] A. E. Jimenez-Cano, J. Martin, G. Heredia, R. Cano, and A. Ollero, "Control of an aerial robot with multi-link arm for assembly tasks," in *IEEE Int. Conf. Robotics and Automation (ICRA)*, Karlsruhe, Germany, MAy 6-10 2013.

[24] K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," *Natural Computing, Springer*, vol. 1, pp. 235–306, 202.

[25] D. Alejo, J. M. Díaz-Báñez, J. A. Cobano, P. Pérez-Lantero, and A. Ollero, "The velocity assignment problem for conflict resolution with multiple aerial vehicles sharing airspace," *Journal of Intelligent and Robotic Systems*, vol. 69, no. 1-4, pp. 331–346, 2013.

[26] L. Pallotino, V. G. Scordio, A. Bicchi, and E. Frazzoli, "Decentralized cooperative policy for conflict resolution in multi-vehicle systems," *IEEE Transactions on Robotics*, vol. 23, no. 6, pp. 1170 –1183, December 2007.

[27] J. Kennedy and R. Eberhat, "Particle swarm optimization," in *IEEE Internatioal Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948.

[28] M. E. H. Pedersen, "Good parameters for particle swarm optimization," in *Hvass Laboratories, Technical Report no. HL1001*, 2010.

[29] "Alpha test:," http://parasol.tamu.edu/groups/amatogroup/benchmarks/mp.

[30] R. Conde, D. Alejo, J. A. Cobano, A. Viguria, and A. Ollero, "Conflict detection and resolution method for cooperating unmanned aerial vehicles," *Journal of Intelligent & Robotic Systems*, vol. 65, pp. 495–505, 2012, 10.1007/s10846-011-9564-6.