Evolutionary Path Planner for UAVs in Realistic Environments

J.M. de la Cruz
jmcruz@dacya.ucm.es

E. Besada-Portas
evabes@dacya.ucm.es

L. Torre-Cubillo
ldelatorre@gmail.com

B. Andres-Toro
deandres@dacya.ucm.es

J.A. Lopez-Orozco
jalo@dacya.ucm.es

Departamento de Arquitectura de Computadores y Automática. Universidad Complutense de Madrid
Av. Complutense s/n.
Madrid, 28040, Spain
(+34) 913944477

ABSTRACT

This paper presents a path planner for Unmanned Air Vehicles (UAVs) based on Evolutionary Algorithms (EA) that can be used in realistic risky scenarios. The path returned by the algorithm fulfills and optimizes multiple criteria which (1) are calculated based on properties of real UAVs, terrains, radars and missiles, and (2) are used to rank the solutions according to the priority levels and goals selected for each mission. Developed originally to work with only one UAV, the planner currently allows us to obtain the optimal path of several UAVs that are flying simultaneously. It works globally offline and locally online to recalculate a part of the path when an unexpected threat appears. Finally, the effectiveness of the solutions given by this planner has been successfully tested against a simulator that implements a complex model of the UAV and its environment.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem solving, control methods and search – *heuristic mehods, plan execution, formation and generation*

General Terms

Algorithms

Keywords

Multiobjective Evolutionary Algorithms, path planning, UAVs.

1. INTRODUCTION

Unmanned Aerial Vehicles (UAV) are aircrafts without onboard pilots that can be remotely controlled or fly autonomously based on pre-programmed flight plans [13]. They can be used in a wide variety of fields, both civil and military, such as surveillance, reconnaissance, geophysical survey, environmental and meteorological monitoring, aerial photography, and search-and-rescue tasks.

In military missions they work in dangerous environments, where it is vital to fly along routes which keep the UAVs away from any type of threat and prohibited zones. The best routes are those which minimize the risk of destruction of the UAV and optimize some planning criteria (such as flying time and path length) while fulfilling all the physical constraints of the UAVs and its environment, plus the restrictions imposed by the selected mission. The original path obtained offline by the planner is not always valid in dynamic environments where the position of all the threats is not known before hand. Therefore, the planner must be able to also work online in order to propose a new path during the mission of the UAV when a pop-up (unknown threat) appears.

Finding the optimal solution to the route planning problem is NP-complete ([14]) and so the problem has been approached with different heuristics such as A* [17], [15], MILP [10], [6], [11], nonlinear programming [9] and others. The planners based on those techniques deal with a simplified version of the original problem (point-mass dynamics, discretising the solution space, and, in some of them, linearizing the objective function and the constraints) where the addition of new constraints or objective indexes is a difficult task.

EA has been proven to be an efficient and robust technique to deal with complex multicriteria constrained problems [1] and so, it has already been used to solve different aspects of the UAV problem, such as in [7], [8] and [19]. The first two papers focus on finding the optimal path of a single UAV that flies over simulated terrain, while [19] optimizes the path of several cooperating UAVs in military environments. Multi-UAV route optimization is also done in [9] and [10].

In those three papers the problem is formulated as finding a set of 3D points (x,y,z), that define the trajectory followed by the UAV which should minimize several criteria and fulfill a set of restrictions. Although the 3D points in [7] and [8] determine the spline curve [4] followed by the UAV and in [19] their linear segments, the codification proposed in these papers creates a big search space in real scenarios. Besides, the evaluation method used in those algorithms does not provide an easy support to modify the priorities of the objectives and their goals for different missions, and only in [7] is the concept of pareto optimality [18] used. To minimize the time response of the planner, all of them use an extremely simple model of the UAV, terrain and threats, and the performance of the planner is tested against the same model instead of a realistic complex simulator.

Our work combines the work presented in those papers, and extends it by introducing some new ideas, optimization and constraints criteria. The goodness of each possible path is evaluated according to the model of a real aircraft (F-16 model, [13]) for the UAV, a real map of the world for the terrain and the characteristics of the radars used in the simulator. In our research, not only is the position of the UAV considered, but also its velocities, which are needed for calculating the fuel consumption and risk of the trajectory. The terrain, which is a map of any part of the world as in [19], is used for checking that the UAV does not collide against it (as in the previous papers) but also to inhibit the possibility of the radar to detect the UAV when it is hiding behind mountains. The properties of the three types of radars and the missiles have been obtained analysing the data from the simulator and are used to measure the threat risk (as the probability of detection and the probability of being destroyed). The user can force the UAV to pass through several specific points (and not only one as in [7]) and some of them can be defined as refuelling points. Our planner also supports the definition of prohibited zones (No Flying Zones, NFZ), which the UAV has to avoid due to a mission restriction. Additionally, we use a relative polar coordinate codification for the waypoints of the spline curve that defines the path, which accelerates the convergence of the algorithm. Our algorithm can also obtain the path of a single UAV (as [7] and [8]) or the paths of several UAVs (as [19]) that have to be flying simultaneously without running into each other.

Table 1 summarizes these properties and compares our planner (last column) with the previous work ([7], [8] and [19]). The compared properties are named in the first two columns, and when there are several options in the same row the selected one can be identified by the capital letters that are shown in the second column.

Table 1. Comparisons of the properties of the planners

		[7]	[8]	[19]	Us
Codification: List of 3D points	**SPline/SEGment**	SP	SP	SEG	SP
	Absolute/Relative	A	A	A	R
	Intermediate Points (None/One/Several)	O	N	N	S
	Refuelling points				√
UAV Model	**Point/Velocity**	P	P	P	V
	Simple/Complex	S	S	S	C
	1 UAV/Several UAVs	1	1	S	S
Radar Model	**None/Simple/Complex**	N	N	S	C
Terrain	**Simulated/Map**	S	S	M	M
	Masking radars				√
	Prohibited zones				√

We also introduce several features in the EA to speed up the computation such as a local search operator used periodically over the best solutions, the possibility of flying at constant altitude, the use of hypercubes [3] in the substitution method in order to maintain a good diversification of solutions, and the initialization of the algorithm with previous solutions for a specific scenario.

It is also important to highlight that we use a generic multiobjective pareto evaluation function with goals and priorities [5] that permits the change of the priorities levels of the different objectives for different missions. Besides a realistic and complete simulator, where pop-ups can appear anywhere and make the algorithm obtain a new local plan, is used to evaluate the performance of the planner.

The rest of the paper is organized as follows. In section 2 the codification of the problem solution is presented. Section 3 presents the function used to evaluate the multicriteria solutions as well as the constraints and optimization indexes used in our experiments when using a single UAV. In section 4 the genetic operators and function used by the single UAV planner are introduced, while section 5 shows how the single planner has been modified to deal with multiple UAVs. Experimental results are given in section 6, followed by our discussion and conclusions in section 7.

2. INDIVIDUAL STRUCTURE

The first step when applying a EA to solve an optimization problem is to determine how the solutions (the individuals of the EA) are going to be codified. Our individual is a list of 3D points (waypoints) which are used to calculate a spline curve [4] that constitutes the solution of the problem: the 3D path of the UAV. So, our algorithm searches for the best list of waypoints which define a feasible and optimal spline trajectory for the UAV.

Each waypoint in the 3D space is defined by three coordinates, whose values can have different meaning depending on the selected codification. For example, in [7], [8] and [19] they are the ‘x’, ‘y’ and ‘z’ of the absolute Cartesian coordinates. This selection generates a very big search space in realistic scenarios because with that codification each point of the space can be reached from the previous waypoint.

To limit the search space our algorithm uses a relative polar coordinate system, with absolute ‘z’ (the altitude) and polar coordinates ‘r’ (radius) and ‘θ’ (angle) relative to the previous waypoint. The first θ angle is the angle formed between the initial vector direction and the vector direction defined by the start point and the first waypoint. For any other waypoint, this angle is measured according with the schema in figure 1. The values of θ are measured clockwise with respect to the (i-1,i) vector direction. This way, $\theta max = 2\pi - \theta min$. The solid line represents the spline curve (which is the path of the UAV) obtained from the waypoints.

The three parameters are codified as real numbers limited between two values. For ‘z’ the limits constitute the minimal and maximal altitude that the UAV can reach. The ‘r’ limits represent the maximal separation between two consecutive points. And the limits of the ‘θ’ are related by the UAV manoeuvrability.

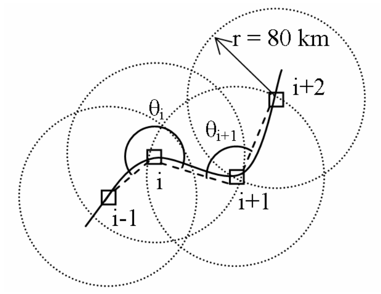

Figure 1. Relative polar coordinates for 'x' and 'y'.

The selected coordinate system forces the algorithm to have a predefined order of waypoints. This is positive because that order already exists and if it were not imposed by the codification the algorithm would have to find it. So, the selected codification lets us dramatically reduce the search space and computation time.

The main drawback of our codification (a list of 'r', 'θ' and 'z' tuples) is that the algorithm has to include a constraint that prevents the path to get outside the map limits, but the EA usually satisfies this condition in only a few generations.

3. EVOLUTIONARY EVALUATION FUNCTION OF THE SINGLE UAV PLANNER

There are: (1) constraints that the path of a single UAV must satisfy completely due to the physical restrictions of the UAV and terrain; and (2) objectives that have to be minimized according to the selected mission. Their values are calculated using the spline curve obtained from the waypoints, the terrain and the model of the UAV (constrained by the properties of the vehicle), radars and missiles (characterized after multiple simulations).

So far, we have defined for the offline planner 6 constraints and 4 objectives or optimization indexes to be minimized. Their number can be modified to include other constraints of the model or to define other types of missions. Moreover, an even more complete model, such as a realistic simulator could be used to obtain the objective values, although that would require a lot of computation time in the evaluation step of the EA.

The time needed for evaluating the solutions is especially problematic when the planner starts running online due to the presence of a sudden pop-up. In order to speed up the algorithm the online planner uses a simpler model of the problem and a different set of constraints and optimization indexes. This difference is justified because the objective of the online planner is to avoid the threat and so, minimizing some of the offline planner indexes is no longer important.

In both cases, the constraints and objective values are combined in the evaluation step of the evolutionary algorithm to obtain the fitness of each solution. In our case, we use the generic evaluation multiobjective method based on goals, priorities and Pareto sets proposed by Fonseca and Fleming [5], where the objectives are placed in priority levels and where it is possible to impose limits (goals) on each of them. In the following sections we describe the objective values and the evaluation method with more detail.

3.1 Constraints

A feasible path for the UAV must fulfill 6 constraints. The first five constraint indexes measure the number of times each of them is violated and so their value must be zero for a feasible solution, while the last one checks if an upper limit has been exceeded.

The implemented constraints are the following:

- *Terrain.* A path must not get under the terrain. This also includes avoiding collisions with mountains. The algorithm increments this value each time one of the spline trajectory points is below the altitude of the map.
- *Turning radius.* An UAV cannot follow a path where 3 consecutives points form a turning radius smaller than a minimum one, whose value depends on the physical characteristics of the UAV. Each time this constraint is not satisfied its value is incremented by one unit.
- *Map limits.* The UAV must stay inside the limits of our known map. Every point of the spline curve must stay inside the limits and as the selected codification does not ensure that this constraint is always fulfilled, this value measures how many points of the trajectory does not satisfy this restriction.
- *Maximum climbing and diving slope.* UAVs cannot reach a greater slope than the one imposed by their dynamic characteristics, which is usually different when the UAV is climbing than when it is diving. Each time this constraint is not fulfilled this value is incremented by one unit.
- *Flight prohibited zones (also called No Flying Zones, NFZs).* There can be certain zones where the UAV must not enter such as high risk zones, unknown areas, etc. The algorithm checks if any point of the spline curve is inside any of the NFZs, and if that is the case, this value is incremented by one unit.
- *Fuel.* UAVs have a limited quantity of fuel and they have to reach their final goal or a refuelling point before consuming it. We use a model that estimates the fuel consumption in three different cases: a) when the UAV is flying horizontally, b) when it is flying with maximum slope and c) when it is flying with maximum turning angle. For intermediate cases the fuel consumption is calculated as a weighted mean of the 3 previous cases. The constraint is satisfied and given a value of 0, when the fuel consumption is lower than the total fuel transported by the UAV; in other case, the value given is the extra fuel that would be needed.

In the online planner some of the constraints, such as the *Flight Prohibited Zones* and *Fuel* limit, don't need to be considered since the priority is to avoid the pop-up.

3.2 Optimal Path for the Mission

The optimality of the path is determined by the following 4 optimization indexes in the case of the offline planner:

- *Minimum length path.* Shorter paths are better than longer ones because they require less time of flight. With the purpose of defining generic limits for this cost function its value is calculated as the ratio between the real path length of the solution and the minimum possible length.

- *Minimum probability of kill (Pk).* This function depends on the model used for the Air Defence Units (ADUs), which consist of a radar and a set of missiles. Based on many simulations we have obtained a function of the probability of destruction of an UAV dependent on the type of ADU, presence of mountains, the distance, relative altitude and orientation existing between the ADU and the UAV. This objective index stores the accumulated probability of kill for the whole path.
- *Minimum flight altitude.* UAVs flying at low altitude consume less fuel and can benefit from the mask effect of the terrain (what helps them to avoid radars). This objective is the maximum difference between the 'z' coordinates of the trajectory and the altitude of the terrain.
- *Minimum radar detection probability.* UAVs that are not detected by any radar are never destroyed and so minimizing this value creates safer paths. Our model uses again a probability function obtained after many simulations that dependent on the type of radar and presence of mountains, Radar Cross Section of the UAV and the distance between the radar and the UAV at every particular point of the trajectory.

The map of the terrain is used for calculating the flight altitude and the probabilities of radar detection and kill. Selecting the appropriate map resolution is important, because high resolution increases the confidence in the values of these objective indexes at the expense of reducing dramatically the calculus speed of the algorithm.

In the case of the online planner, where the real objective is to avoid the new threat, the flight altitude and length of the path are no longer important. So, these objectives are not longer used, which minimizes the time used by the algorithm to evaluate the different solutions.

3.3 Evaluation Function

The evaluation method used to decide which solutions or individuals are better than the others is the Fonseca and Fleming multiobjective method based on goals, priorities and Pareto sets [5], where the objectives are ordered by priority level and limits for each of them can be imposed.

This method has been used successfully in real world constrained problems [1]. The values used to measure when the constraints are satisfied are placed as high priority objectives and their values must satisfy the correct goals (be 0 in our problem). The objective indexes to be minimized are located at lower priority levels. With this division individuals that fulfill the constraints are better than the ones that do not, and the latter are organized according to the number of constraints that they satisfy.

Furthermore, as we are using a general method for evaluating the fitness of the function given the constraints and objective indexes, we can easily add or remove constraints or optimization indexes as needed. For instance, in the online planner some constraints are removed in order to reduce the computation burden.

As an offline planning example, the following table shows the priority and limits of each of the values for a particular mission. The highest priority is the 1st level, the intermediate the 2nd one and the lower the 3rd one. All the constraints are optimized to their limit to search for optimal solutions that fulfil them. The objective values are placed in the following two levels according to their importance for the mission. In this example, we want to minimize simultaneously the length of the path (whose relative value is limited) and ensure that the probability of kill of the mission is zero. When two solutions are equally good, in a Pareto sense, inside the intermediate priority level (2nd), the best solution will be the one that minimizes simultaneously the flight altitude (to use the terrain as a mask for known and unknown threats) and the probability of being detected by a radar.

Table 2. Evaluation function priority levels and limits for a given mission for the offline planner

Constraint conditions						
Name:	Terrain	Turning radius	Map limits	Maximum slope	NFZ	Fuel
Level:	1st	1st	1st	1st	1	1st
Min	0	0	0	0	0	0
Max	0	0	0	0	0	0

Optimization objectives				
Name:	Path length	Probability of kill (Pk)	Flight altitude	Radar detection probability
Level:	2nd	2nd	3rd	3rd
Min	1	0	50	0
Max	1.2	0	1000	0

4. EVOLUTIONARY OPERATORS AND FUNCTIONS OF THE SINGLE PLANNER

Our single UAV evolutionary router planner uses, among others, the following operators and functions.

4.1 Crossover

This operator is used to obtain two new individual from their parents, which are selected in our algorithm with the roulette method. We use two random cross points that divide each individual into three sections, each of them formed by a complete group of waypoints. That is, we treat the 3 parameters of each waypoint as a whole and do not mix the information of one waypoint of an individual with the information of another. The new individuals will have the first part of the route of the first parent (until the first cross point), the second part of the second parent (from the first cross point until the second one) and the third part of the first parent again (from the second cross point until the end of the individual). The other new individual or child will have the remaining information.

4.2 Mutation

The mutation operator can change (if the mutation probability is satisfied) the value of any parameter of a solution to another value obtained randomly from the interval of values for that parameter (altitude, radio or angle).

4.3 Immigration

Every new generation a number of new individuals (immigrants) are generated randomly and added to the whole population to assure the gene's diversity.

4.4 Local Search Procedure

Every 25 generations the algorithm picks up the best solutions and it applies a non evolutionary optimization method to improve them. The implemented method modifies each parameter of the individual by adding and substracting a predefined value (which varies with the generation number) and selects the best of the three different solutions (the original, the one that has a parameter incremented and the one that has it decremented). The final individual is the one improved sequentially by this procedure.

4.5 Substitution

The substitution method uses an elitist method that preserves the pareto front of solutions. Every time a set of individuals is created (by the crossover and mutation steps), we form a new population with the new individuals and the old individuals that form part of the best two pareto groups (discarding the others).

This way of proceeding can increment the number of individuals in the population with the number of generations when the number of individuals of the Pareto front grows. As more solutions need more processing time, it is convenient to limit the maximum number of individuals in the population. Although, the easiest and quicker solution is to pick the survivors randomly from the Pareto front, the selected individuals will not necessarily be representative of it. So, we have decided to use a hypercube approach, in the line of the methods proposed by Coello and Salazar [3], that divides the Pareto front into several regions and select the survivors randomly from inside the different hypercubes.

4.6 Selection of the Final Optimal Path

The evolutionary algorithm returns the last population, where the best individuals are placed along a Pareto front. So, we must make use of a final selection criterion. There are several possibilities, such as taking the one with the shorter length path or the lower probability of kill. Our choice for the offline planner when the priority levels are the ones presented in Table 2 is to divide the decision in two possible cases:

- *When there is a feasible solution with 'probability of kill' equal to zero.* This is the trivial case, because in this situation there is only one element in the Pareto front since one of the two objectives with intermediate priority (2nd) is already a minimum and the comparison of the other objective ('path length') determines which individual is the best.
- *When all the 'probability of kill' objectives are greater than zero.* First of all, we find inside the first pareto front the solution with the lowest probability of kill (Pk_{min}) and then we modify the 'probability of kill' of each individual i, dividing it by the minimum value (Pk_{min}). All the solutions with modified probability of kill greater than a predefined value (for example 1.05) are discarded. The others don't have a significant probability of kill, and so we pick among them the one with shorter path length. This can make the final solution have a probability of kill just a bit higher than the minimum as well as a reasonable path length.

The selected solution is returned by the planner and the pareto front stored to be able to reuse it in the future. For instance, it can be used to find a trajectory over a slightly modified solution which will drastically reduce the computation time. The online planner can also benefit from a local use of this information.

5. MULTIPLE UAVs PLANNER

Calculating the paths of several UAVs that are flying simultaneously adds new complexity to the planner due to the space constraints that one path imposes on the others and the cooperative objectives that the set of UAVs have to minimize.

This task can be carried out using several single UAV planners in parallel, as long as they exchange some information during their evaluation step. This information is needed in order to measure the coordinating objective functions over each population, taking into account the possible paths of the remaining UAVs. Therefore, each single planner has to receive information related to the population of the remaining single planners before evaluating the cooperative constraints and objectives of each individual of its own population. As each UAV only follows the final optimal path proposed by its planner, the paths of the UAVs which are being evaluated inside each planner only need to be compared with the currently best path of each one of the remaining planners. However, the original single planners maintain a pareto front of best individuals and only selects the final optimal path when the last population is obtained. To be able to determine the final optimal path in each generation, this step has to be included in the new cooperating planners. Table 3 shows the evaluation steps in the cooperating planners.

Right now we only make use of a single cooperative objective value that checks if the path that is being evaluated can make the UAV collide against the remaining UAVs when they are flying according to the best path proposed by their own planner. As the feasibility of the plan also depends on the value of this objective, it is considered a constraint that will be placed in the 7th step of the evaluation step with the individual constraints (section 3.1) in the highest priority level.

The possibility of defining different intermediate points for each UAV lets us force cooperation in a predefined manner. However, more objectives, such as the order in which they should arrive to each objective could also be considered.

Table 3. Evaluation step in the cooperating planners

1. *Evaluating the objectives of section 3.1 and 3.2.*
2. *Calculating the path ranking with the values of step 1 and the evaluation function of section 3.3.*
3. *Sorting the individuals according to the ranking calculated in step 2.*
4. *Finding the best individual according to the final criteria of section 4.6 and sending it to the other planners.*
5. *Waiting for the best individuals of the other planners.*
6. *Calculating the coordination objectives with the received information from other planners.*
7. *Recalculating the new path ranking with the values of step 1, step 6, and evaluation function of section 3.3.*

6. EXPERIMENTAL RESULTS

The UAV planner presented in this paper is part of a complex system that we have developed to test the performance of different UAV planners for hostile environments. This system lets the user (1) define the position of the known and pop-up ADUs and intermediate points of the UAVs; and (2) plan their routes and test them against a simulator that uses a real map of the world and a more complex model for the UAVs and ADUs.

Although our planner can work globally offline to determine the original best path for each UAV and locally online to react when it observes a pop-up, the current version of the simulator needs to have the original plan and the optional routes before launching the simulation, because it is not able to launch any planner. To calculate the optional routes before hand, we check if the globally optimal path enters the detection area of any pop-up ADU, and if that is the case the online planner finds a path from the position inside the detection area to where it should leave it. During the simulation, if the pop-up ADU is detected by the UAV, the simulator gets the optional path and makes the UAV follow it.

The model used by the planner for obtaining the probability of kill is conservative enough to ensure that when this probability is null the UAV will be able to reach the final destination if pop-up ADUs do not appear. When the value is non-zero, the UAV already knows that it is assuming some risk, and depending on the simulation it will succeed or fail.

The planner has been tested with success in multiple scenarios and in this paper we present some examples that reflect important characteristics of the algorithm.

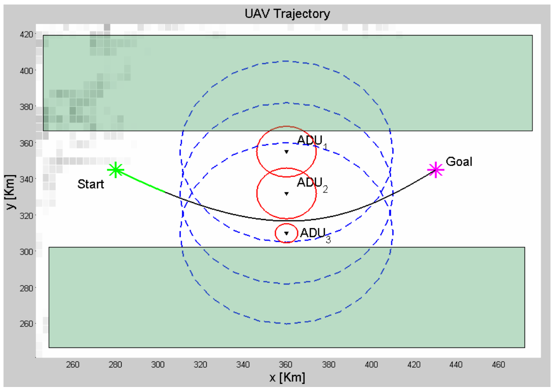

Figure 2. Eagle eye view of experiment A

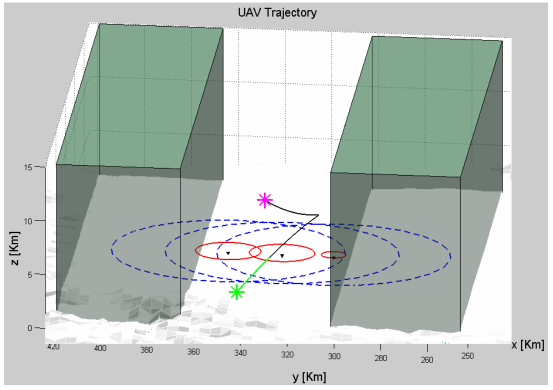

Figure 3. Isometric view of experiment A

6.1 Example A

This example, presented in figures 2 and 3, shows the ability of the single UAV planner to find no risky paths between the Start and Goal points in a static environment with 3 ADUs and 2 NFZs.

The big dotted blue circles mark the maximum distance of detection of each ADU while the small red ones enclose the zones where Pk>0. NFZs are represented as rectangular green zones. There is a little corridor (of 5 kms) where Pk=0 and the algorithm has been able to find it even though the Pk and the path length objectives are in the same priority level. Figure 3 shows that the altitude is also being minimized and that the trajectory is always below the Goal altitude, which is higher than the Start point. The proposed path is obtained with only 200 generations and 14 pairs of parents to cross in each generation. As the path selected by the algorithm is feasible and safely optimal, all the tests carried out in the simulator have succeeded to reach the Goal.

6.2 Example B

This example, presented in figure 4, shows the ability of the algorithm to find low-risk paths in areas with a high number of ADUs, where the Pk=0 area has a complex shape: The ADUs are placed building a spiral around a non risky zone and the UAV is asked to go from the Start point outside the spiral to the Goal which is placed in the centre of the spiral.

Our planner is able to find the safe zone and also minimize its path inside it by finding a solution which is tangent to the Pk>0 red areas of the exterior and interior ADUs of the spiral. The proposed solution is found with 800 generations and 14 pairs of parents to cross in each generation. As the selected path is feasible and has a null Pk, the UAV is able to find its Goal in all the simulations.

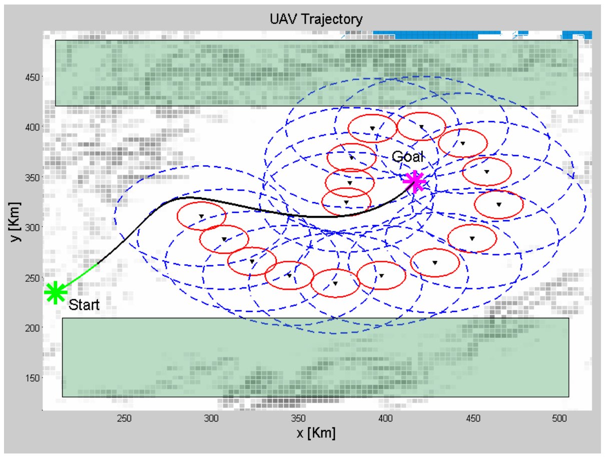

Figure 4. Eagle eye view of experiment B

6.3 Example C

This example, presented in figures 5 and 6, illustrates how the algorithm works with pop-ups, obligatory intermediate points and roundtrip paths (that are a special case of a path with at least an intermediate point and the Goal in the same position as the Start point). There is 1 UAV in a scenario with 3 intermediate points (WP_1, WP_2, WP_3), 1 known radar (ADU_1), 1 NFZ and 1 pop-up radar (POP_2) near the Start point.

The black solid line is the original generated path, while the dots define the path calculated by the algorithm when the pop-up appears (note there are not dots before the route passes near the pop-up). In figure 6 we can see how the replanning permits the evasion of the pop-up. Without it, the UAV could have been killed as it was originally going to pass inside the PKill area of the ADU.

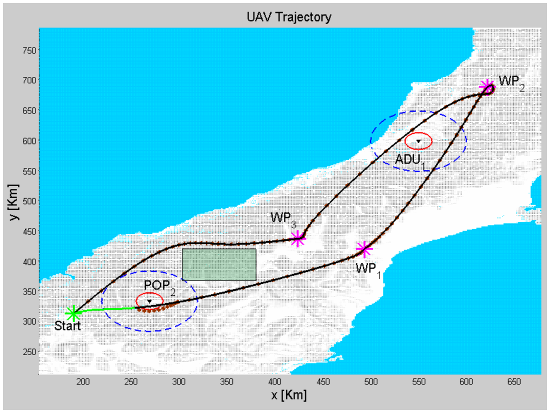

Figure 5. Eagle eye view of experiment C

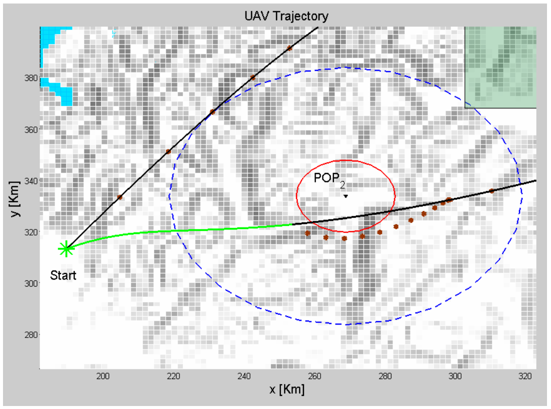

Figure 6. Eagle view of experiment C near the pop-up

6.4 Example D

Finally, this example, presented in figure 7, shows the advantages of considering the terrain inhibition and the results of the multiple UAVs planner for two UAVs flying at different velocities.

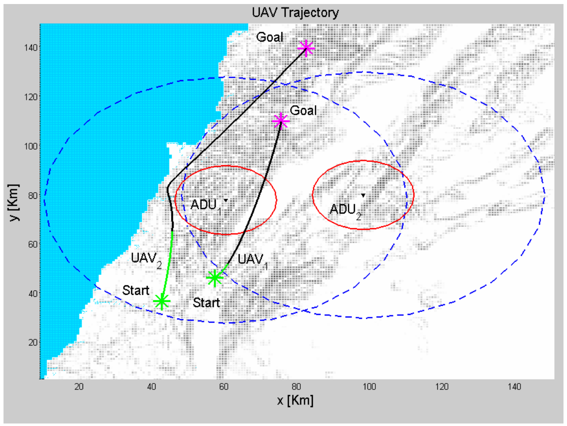

Figure 7. Eagle eye view of example D

The planner finds a path for the UAV$_1$ inside the original PKill area of ADU$_1$ because behind its southern mountains there is a valley that the UAV$_1$ can use to hide from the ADU$_1$, inhibiting its detection effects. The UAV$_2$ needs to evade the Pkill area of the ADU$_1$ because the topography in its northern area doesn't let the UAV$_2$ benefit from the inhabitation possibilities of the southern region. The two trajectories don't collide because they have been calculated by the multiple UAVs planner and have been tested with success against our simulator.

7. CONCLUSIONS

The paper shows an EA planner that calculates with success the trajectory of one or several UAVs flying simultaneously in hostile complex dynamic environments. The mission is defined by the intermediate points that the UAV has to visit, as well as the priorities and goals of the selected optimization criteria.

The evaluation of the different individuals of the population uses a real model of the UAV, a real map of the world and models of the ADUs related to the properties of those presented in the simulator.

Our model of the UAV calculates its position and velocities to be able to calculate the fuel consumption and risk of the trajectory. The continuity of the trajectories proposed by the planner is supported by its spline origin and its smoothness is due to the constraint related to the turning angle. The UAV model lets us also measure properly the climbing and diving slopes. When the planner returns a feasible path, it can actually be followed by the simulated UAV. As a matter of fact, the model of the UAV used by the planner to evaluate the trajectories proposed by the EA works so well (as we checked with the simulator) that in the absence of threats, we know before hand that the UAV will be able to follow the path quite closely.

The terrain, which is a real map of the selected region of the world is used for calculating the altitude of the UAV and the probability of detection and kill, so the planner can use it to mask the UAVs that are flying behind a mountain. Taking advantage of the masking possibility has let the planner find some solutions we didn't expect, at the expense of increasing the computational time of the planner. As this extra time is significant for online planning, the calculations to test if the ADU and UAV are masked should not be used.

The models of the three types of ADUs (radar plus sets of missiles) have been obtained analysing the data from the simulator, and are used to measure the threat risk (as the probability of detection and the probability of being destroyed). These models are conservative enough to ensure that when the probability of kill is null during the whole trajectory the UAV will be able to reach the final destination if there is not any pop-up. When the value is non-zero, the UAV already knows that it is assuming some risk, and depending on the simulation it will succeed or fail. The planner also supports the definition of prohibited zones (NFZ), which the UAV have to avoid due to a mission restriction. Defining unexplored areas of the terrain as NFZs can increase the survival probability of the UAV.

The spline path returned by the algorithm is codified by its list of waypoints in a relative polar coordinate system, which accelerates the convergence of the algorithm without loss of generality. The speed of the algorithm is also improved with the periodic use of a local search procedure, the possibility of flying at constant altitude, the use of hypercubes in the substitution method to maintain a good diversification of solutions, and the initialization of the algorithm with previous solutions for a specific scenario.

To evaluate the individuals of the population, the generic Fonseca and Fleming multicriteria method is used, putting the constraints in high priority levels and the optimitation indexes in lower ones. With the choice of this method, adding new objectives and constraints is straightforward, and so the planner can be easily extended to face more complex scenarios and flight missions.

Finally, it is important to point out that other UAVs planning tasks such as target recognition and tracking ([12], [16]) are also being solved with EA. Although our research is currently focused in the problem of finding the best path that evades the risk, we are analysing the possibilities of widening the types of tasks that our EA planner can optimize.

8. ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions.

This research was funded by the Community of Madrid, project "COSICOLOGI" S-0505/DPI-0391, by the Spanish Ministry of Education and Science, project "Planning, simulation and control for cooperation of multiple UAVs and MAVs" DPI2006-15661-C02-01, and by EADS (CASA), project 353/2005.

The evolutionary algorithm used for solving the problem is implemented with the Matlab Toolbox EVOCOM [2].

9. REFERENCES

[1] Andres-Toro, B., Besada-Portas, E., Fernandez-Blanco, P., Lopez Orozco, J.A., and de la Cruz, J.M. 2002. Multiobjective optimization of dynamic processes by evolutionary algorithms. In Proceeding of the 15th Triennial World Congress of the IFAC. (Barcelona, Spain, July 2002)

[2] Besada-Portas, E., Lopez-Orozco, J.A., Andres-Toro, B. 2002. A versatile toolbox for solving industrial problems with several evolutionary techniques. Evolutionary methods for Design, Optimization and Control. International Center for Numerical Methods in Engineering. March 2002.

[3] Coello Coello, C.A., and Salazar Lechuga, M. 2002. MOPSO: A proposal for multiple objective particle swarm optimization. In Proceeding of Congress on Evolutionary Computation (CEC'2002). (Piscataway, New Jersey, May 2002).

[4] Farin, G. 1988. Curves and Surfaces for Computer Aided Geometric Design. A practical Guide. New York:Academic.

[5] Fonseca, C.M., Fleming, P.J. 1998. Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms- Part I: Unificied Formulation. Transactions on Systems, Man and Cybernetics. Part A: Systems and Humans. Vol 28, no. 1, (January 1998). 26-37.

[6] Kuwata,Y. and How J. 2004. Three Dimensional Receding Horizon Control for UAVs. In Proceedings of AIAA Guidance, Navigation, and Control Conference and Exhibit, (August 2004). AIAA 2004-5144.

[7] Mittal, S., and Deb, K. 2004. Three-Dimensional Offline Path Planning for UAVs Using Multiobjective Evolutionary Algorithms. Kangal Report no. 2004-008

[8] Nikolos, I.K., Valavanis, K.P., Tsourveloudis, N.C. and Kostaras, A.N. 2003. Evolutionary Algorithm Based Offline/Online Path Planner for UAV Navigation, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, Vol 33, no. 6 (December 2003).

[9] Raghunathan, A., Gopal, V., Subramanian, D., Biegler, L. and Samad, T. 2004. Dynamic Optimization Strategies for 3D Conflict Resolution of Multiple Aircraft. AIAA Journal of Guidance, Control and Dynamics, 27 (4). (2004), 586-594.

[10] Richards, A.G., How, J.P. 2002. Aircraft Trajectory Planning with Collision Avoidance Using Mixed-Integer Linear Programming. In Proceeding of the American Control Conference (May 2002).

[11] Ruz, J.J., Arévalo, O., de la Cruz, J.M, and Pajares, G. 2006. Using MILP for UAVs Trajectory Optimization under Radar Detection Risk. In Proceedings of the 11th IEEE International Conference on Emerging Technologies and Factory Automation (Prague, September 2006).

[12] Shima, T., Schumachery, C. 2005. Assignment of Cooperating UAVs to Simultaneous Tasks using Genetic Algorithms. In Proceeding of the AIAA Guidance, Navigation, and Control Conference and Exhibit. 15-18 (San Francisco, California, August 2005).

[13] Stevens, B., Lewis F. 2004. Aircraft Control and Simulation. 2nd Edition. Wiley. 2004.

[14] Szczerba, R.J. 1999. Thread netting for real-time, intelligent route planners. In Proceeding of the IEEE Symp. Inf., Decision and Control. (Adelaida, Autralia, 1999)

[15] Szczerba, R.J., Galkowski, P., Glicktein, I.S., Ternullo, N. 2000. Robust algorithm for real-time route planning. IEEE Transactions on Aerospace and Electronic Systems, vol. 36, 3, (July 2000), 869-878.

[16] Tian, J., Shen, L., Zheng, Y. 2006. Genetic Algorithm Based Approach for Multi-UAV Cooperative Reconnaissance Mission Planning Problem. Lectures notes in Computer Science. Springer Berlin. Volume 4203/2006.

[17] Trovato, K.I. 1996. A* Planning in Discrete Configuration Spaces of Autonomous Systems. PhD thesis, Amsterdam University, 1996.

[18] Veldhuizen, D. A. V. and G. B. Lamont.1998. Evolutionary computation and convergence to a pareto front. In Proceeding of the Genetic Programming 1998 Conference.

[19] Zheng, C., Li, L., Xu, F., Sun F., and Ding, M., Evolutionary Route Planner for Unmanned Air Vehicles, IEEE Transaction on Robotics, Vol. 21, no. 4, (August 2005), 609-620.