A Fuzzy Adaptive Differential Evolution for Multi-objective 3D UAV Path Optimization

Debesh Adhikari
Department of Computer Science
University of North Dakota
Grand Forks, ND, 58202, U.S.A.
Email: debesh.adhikari@und.edu

Eunjin Kim
Department of Computer Science
University of North Dakota
Grand Forks, ND, 58202, U.S.A.
Email: ejkim@cs.und.edu

Hassan Reza
Department of Computer Science
University of North Dakota
Grand Forks, ND, 58202, U.S.A.
Email: reza@cs.und.edu

***Abstract*—This paper presents a fuzzy adaptive differential evolution (DE) for 3D UAV path planning. The path-planning problem is formulated as a multi-objective unconstrained optimization problem in the aim of minimizing the fuel and the threat cost as well as finding the shortest path. A fuzzy logic controller is used to find the parameter values of DE during this optimization process. The mutation operation of DE is modified in a way to strike a balance between the DE/rand/1 and DE/best/1 strategies. This method is compared with both DE/rand/1 and DE/best/1 and it relatively outperformed both the classical variations. As it is often tedious to find the right parameter values for DE, this method will give freedom from the process of finding the parameter values for the acceptable performance in 3D path planning optimization.**

***Index Terms*—differential evolution, fuzzy adaptive differential evolution, fuzzy logic, UAV path planning**

I. Introduction: Path Planning Problem

Path planning for UAVs is an optimization problem that plays a vital role in UAV research. Although UAVs are gaining popularity and are used for both civil and commercial purposes, their growth is still limited. The lack of complete autonomous decision-making is still seen as one of the limiting factors for the growth of these autonomous vehicles [1].

Research on unmanned aerial vehicles (UAVs) has seen rapid rise in the recent years due to their multipurpose application in various domains. UAVs can be used for wide range of issues such as transportation planning [2], natural disaster monitoring [3], civil and border security [4], agricultural surveillance [5] and many others.

A UAV has to reach the final destination safely from its initial location for its given mission traveling through the shortest path for optimal energy utilization. Thus, path planning for UAVs plays a vital role in UAV research.

As a UAV operates in the sky, UAV can travel via any path therefore it is not feasible to brute force through all the possible combinations to find an optimal route. Thus, there is a need of an efficient optimization technique to plan the shortest path.

UAVs can operate in either dynamic or static environments. A UAV must have some sensing capabilities to operate in the dynamic environment while its not required in the static environment as every information of the environment will be known beforehand. Sensors in UAV is a different field. In this initial path planning research, it is assumed that all the information about the environment, including the threat areas, is known to the UAV.

In our study, the UAV is assumed to be in an environment with different threat areas. The UAV has to avoid the threat areas; otherwise, there will be some penalty for paths passing through those threat areas. The path-planning problem can also be formulated as a three-dimensional path-planning problem or a two-dimensional path-planning problem, by assuming the constant altitude sometime after takeoff. All three dimensions are considered in this study during the entire flight.

UAV path planning is generally formulated as a multi-objective optimization problem. The most critical objective is a path length which can be directly translated to fuel cost and the risk cost of the path (i.e. risk to the threat areas). However, there are researches that take the objectives of more minimization. For example, minimization of the height cost can be taken as a separate objective [6].

In our study, the UAV path planning is considered as a bi-objective optimization problem in the aim of minimization of both the fuel cost and the risk cost. The fuel cost is directly dependent on the path length, whereas there are different ways to model the risk cost. For example, the path may be divided into a number of different segments [6]. At each segment, the distance from each obstacle at different points along the edge is measured. This is a very common approach to model the risk cost for a path. In our study, only the segment of the path that falls within the threat areas is considered for risk cost calculation.

II. Computing Methods for Optimization of Path Planning

Soft computing is a computational paradigm which may be an alternative or a support to a classical hard mathematical computing. It includes a number of computational discipline and their techniques such as fuzzy set and logic, neural network, evolutionary algorithm, swarm algorithm, probabilistic reasoning, chaos theory and rough set theory, etc. In particular, evolutionary algorithms such as genetic algorithm(GA), differential evolution (DE), evolutionary strategy(ES), etc. and swarm algorithms such as ant colony optimization (ACO) and particle swarm optimization (PSO), etc. are proven to

978-1-5090-4601-0/17/$31.00 ©2017 IEEE

be applied to find near optimal solutions for the optimization problems which are deemed beyond the scope of traditional optimization algorithms. In our study, fuzzy logic and differential evolution are applied to the minimization of the fuel cost and risk cost in UAV path planning.

A. Fuzzy logic

Fuzzy logic is a soft computing method that handles uncertainty. Fuzzy logic is developed based on the fuzzy set which was invented by Zadeh in 1965 [7] . In fuzzy set, a membership of an element x to a fuzzy set A is defined by its membership degree to the set in the range of $[0, 1]$ while it is either 0 or 1 in the crisp set.

$$i.e.\ \mu(x) \in [0, 1]$$

The membership degree is further defined by a membership function such as Trapezoid function, Triangle function or Gaussian function, etc. on the universe of discourse.

If a set of tall people is defined in the universe of heights which is greater than or equal to 6 feet, for example, any one whose height is even one inch slightly less than 6 feet will not be a member of the set of tall people in the case of crisp set. In reality, however, he will still be considered tall [8]. Thus, his membership degree of the set of tall people might be close to 1 though not exactly 1. Hence, it can allow an element to be a member of different fuzzy sets with varying membership degrees at once; for instance, a membership degree in the fuzzy set of *Tall people* and and that of *Very Tall people*. Figure 1 illustrates an example for fuzzy membership functions for a person's height.

Fuzzy logic is a generalization of Boolean logic in which the input and output values are extended to a real value in the range of [0, 1] as the degree of truth, instead of the crisp values of 0 or 1 only. Fuzzy logic is designed to handle partial truths. Fuzzy logic allows the use of fuzzy predicates and can handle the imprecision in the language [9]. It can be used to develop sophisticated control systems, as it resembles human decision making like skills from the available information [10].

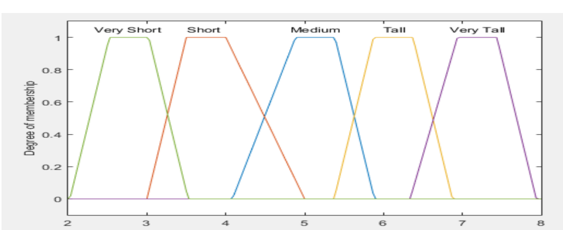

Fig. 1: Fuzzy membership functions for a person's height

A fuzzy inference system (FIS) is used to make a decision with the given inputs defined on the fuzzy sets, through an inference procedure using the fuzzy rules in the rule_base. The common types of fuzzy inference system are Mamdani's FIS and Sugeno's FIS. Hybrid fuzzy inference systems such as neuro fuzzy systems(NFS) have gained popularity due to its significant performance in various areas such as control engineering. A set of fuzzy rules are defined in the form of IF-THEN statements with the input variables defined in the fuzzy sets. The fuzzy rules for Mamdani inference types is in the following form [11]:

R: If x_1 is $\mu_R^{(1)}$ and $\dots$ and x_n is $\mu_R^{(n)}$
then y is μ_R.

where x_1, $\dots$,x_n are the fuzzy input variables of the controller, y is the fuzzy output variable and μ_R is the fuzzy membership function of the fuzzy set in which each input or output variable is defined.

The process of converting crisp inputs to a fuzzy value is called fuzzification. Normally, a fuzzy operator is used to obtain a single membership value if the antecedent of the fuzzy rule has more than one condition. A fuzzy implication operation is applied to obtain a new fuzzy set to the consequent of each rule and the antecedent value. Then, a fuzzy aggregation operator is used for the aggregation process. After aggregation, the fuzzy values are defuzzified, this process is called as defuzzification. There a number of defuzzification methods such as mean of maxima(MOM), center of the area, etc.

B. Differential Evolution

Differential evolution was first introduced by Stron *et. al* in 1995 [12]. DE is a meta-heuristic, stochastic population based search algorithm for an optimization problem. In particular, DE is a type of evolutionary algorithm inspired from biological evolution with the principle of *survival of the fittest*. Differential evolution has been successfully implemented for a wide domain of optimization problems, especially effective in continuous optimization [13]. Different variations of DE have been proposed with different modifications to the original algorithm suiting the problem with promising results [14].

Differential evolution is a population based, iterative direct search algorithm, where mutation is used as the primary operation unlike crossover is a primary operation in genetic algorithm. Differential evolution can directly work with real valued D-dimensional parameter vectors without the need of any encoding schemes. The scaled difference of random and unique population individuals are used for the new population. Differential evolution normally works in three steps.

Step 1: Mutation

The mutation operation of differential evolution is given by:

$$V_{i,G} = X_{r_1,G} + \beta \cdot (X_{r_2,G} - X_{r_3,G}) \tag{1}$$

where X is an D-dimensional individual vector, G is the current generation, $\beta \in (0, 2]$ is the differential weight, r_1, r_2 and r_3 are random numbers in $[1, 2, \dots, NP]$ to choose three random individuals where $r_1 \neq r_2 \neq r_3 \neq i$, NP is the size of population and V_i is a mutated individual i, called a *donor vector*.

Step 2: Crossover

After mutation that generates a donor vector V_i, it undergoes crossover operation with the parent object vector. Some components of the donor vector, V_i, enter into the trial (offspring) vector (U_i) with the probability CR as follows:

$$U_{i_j,G+1} = \begin{cases} V_{i_j,G+1}, & \text{if } randb(j) \leq CR \text{ or } j = rnbr(i) \\ X_{i_j,G} & \text{if } randb(j) > CR \text{ and } j \neq rnbr(i) \end{cases} \tag{2}$$

$j \in \{1, 2, \ldots, D\}$ where
i_j is the j^{th} component of a vector i, $randb(j) \in [0, 1]$ is the j^{th} evaluation of a uniform random number generator and CR is the crossover rate in $[0, 1]$ which is predetermined.

Step 3: Selection

After the crossover, the individual of trial vector, U_i, is evaluated and will be selected if its fitness is better than its parent. If a parent's fitness is better, then the parent is retained. The selection operation is defined as below:

$$X_{i,G+1} = \begin{cases} U_{i,G+1}, & \text{if } f(U_{i,G}) \geq f(U_{i,G+1}) \\ X_{i,G} & \text{otherwise} \end{cases} \tag{3}$$

The DE algorithm is described in the Algorithm 1 [15].

Algorithm 1 General DE Algorithm

Set the generation counter, $G = 0$;
Initialize the parameters, β and CR;
Create and initialize the population of NP individuals
While stopping condition(s) not true **do**
    for each individual X_i **do**
        Evaluate the fitness, $f(X_i)$;
        Create the donor vector V_i
            by applying the mutation operator;
        Create the trial offspring U_i
            by applying the crossover operator;
        if the $f(U_i)$ is better than $f(X_i)$
        then add U_i to the next population at G+1
        else retain the parent X_i to G+1.
    end
end
Return the best individual with the best fitness as solution.

1) Effects of parameters in differential evolution performance: The parameters of DE algorithm play a crucial role in the performance as well as the search behavior of DE. They help to strike a balance between the exploration and exploitation ability in the search space. However, it depends on the nature of problem to set the parameter values of DE algorithm and it is often time consuming to tune the initial parameter values to obtain the optimal performance. Four main parameters of differential evolution are explained below.

a) population size (NP): The higher the number of initial individuals, normally the better it is for the performance of differential evolution. However, increasing population size can have unnecessary computational effects. The effect of population size for different problems and an adaptive strategy is studied in [16].

b) differential weight (β): This is the amplification factor for the difference of two vectors. This parameter helps to provide a balance between the exploration and exploitation of the search space. A high differential weight will normally provide better exploration of the search space, but it might have convergence issues. On the other hand, a low differential weight might cause a premature convergence while it provides better exploitation of the search space.

c) crossover rate (CR): The crossover operation is used to retain the property of parents in the mutated donor vector. This parameter controls which components are retained in the trial offspring vector. There can be different crossover variants which have different influence on the behavior of DE [17].

d) number of generations(GEN): This parameter is used to control the number of iterations the algorithm being executed. Running more number of iterations might not be significant if the solution has already converged; however, running fewer generations might not lead to an optimal solution.

2) Differential evolution variations: Over the years, different variations of differential evolution have been proposed. Some popular DE variations of mutation operation to generate a donor vector are listed in Table I. Note that γ is the second differential weight.

TABLE I: DE Variants

Type	Mutation Operation
DE/rand/1	$V_{i,G} = X_{r_1^i,G} + \beta \cdot (X_{r_2^i,G} - X_{r_3^i,G})$
DE/best/1	$V_{i,G} = X_{best,G} + \beta \cdot (X_{r_1^i,G} - X_{r_2^i,G})$
DE/rand/2	$V_{i,G} = X_{r_1^i,G} +$ $\beta \cdot (X_{r_2^i,G} - X_{r_3^i,G} + X_{r_4^i,G} - X_{r_5^i,G})$
DE/best/2	$V_{i,G} = X_{best,G} +$ $\beta \cdot (X_{r_1^i,G} - X_{r_2^i,G} + X_{r_3^i,G} - X_{r_4^i,G})$
DE/current-to-rand/1	$V_{i,G} = X_{i,G} + \gamma \cdot (X_{r_3^i,G} - X_{i,G}) +$ $\beta \cdot (X_{r_1^i,G} - X_{r_2^i,G})$
DE/current-to-best/1	$V_{i,G} = X_{i,G} + \gamma \cdot (X_{best,G} - X_{i,G}) +$ $\beta \cdot (X_{r_1^i,G} - X_{r_2^i,G})$

III. Related Work

Differential evolution (DE) has been successfully used in path optimization for UAVs [6][18][19]. A combination of differential evolution with other algorithms have also been studied for path planning of UAVs. For example, a hybrid algorithm of DE with Bat algorithm and that of DE with Chaotic theory are studied in [20] and [21], respectively.

Liu *et. al* first proposed a fuzzy adaptive differential evolution in [22]. They used fuzzy logic controllers to adapt the search parameters for mutation and crossover. It was tested on the benchmark functions with standard DE algorithm and it outperformed the standard DE algorithm.

Other meta-heuristic algorithms such as ant colony optimization (ACO) and particle swarm optimization (PSO) have also been applied for 3D UAV path planning. Duan *et. al* used a hybrid ACO-DE algorithm for 3D path planning for uninhabited combat air vehicle (UCAV) in [23]. DE was used as a pheromone updating strategy while ACO was used to search the optimal path with multiple constraints. Zhang *et. al.* presented a fitness scaling adaptive chaotic particle swarm optimization (FAC-PSO) method which used a fitness-scaling, adaptive parameter mechanism, and the chaotic theory in [24]. Foo *et. al* formulated the path planning problem for UAV using PSO to minimize the risk caused from the enemy threats and fuel consumption or a combination of both in [25]. A dual population genetic ant colony algorithm was used by Qian *et. al* [26].

In [27], a hybrid mutation strategy has been proposed by Zhang *et. al*. The parameters are also adaptive but the drawing of parameter values from normal distribution might limit the adaptive capability.

IV. Fuzzy Adaptive DE Model in UAV Path Planning

A. Path formulation

A complete path for UAV is defined as a summation of a number of independent path segments. The number of sub-line segments are kept constant for the ease of use with differential evolution. Some static threat areas are also considered in the environment. The UAV should try to avoid such areas as there will be some extra penalty of traveling through them. The path planning is formulated similar to [18]. The initial location of the UAV is connected to the target location in a straight line. Then, this straight line is divided into a number of line segments. The line segments excluding the threat areas are further divided into a fixed number of discrete points which the UAV can select. For initial population, the number of solutions are constructed in such a way that the UAV from its starting location can select a point randomly in all directions in each of the line segments. The points are connected to form a complete path.

B. Objective function

The main objective for the UAV is to minimize the fuel and threat cost. The fuel cost of the UAV is defined as the path length which is calculated as the Euclidean distance. For the threat cost, the part of line segment that falls within the threat zone is considered. If the line segment is within the threat area, that part of line segment is multiplied with a penalty constant p. Equation (4) and (5) give the fuel cost and threat cost respectively:

$$F_c = \sum_{k=1}^{m} l_k \tag{4}$$

where F_c is the fuel cost, m is the number of line segments, l_k is the length of line segment k, and

$$T_c = \sum_{k=1}^{m} p \cdot S_k \tag{5}$$

where T_c is the threat cost, p is the penalty term, S_k is the length of sub segment passing through the threat area. If the path only touches the threat area, for example, tangent to a sphere, then the length is considered to be 1.

As this is a multi-objective problem, the objective function is formulated using a classical weighted sum approach which is defined as follows:

$$G = w \cdot F_c + (1 - w) \cdot T_c \tag{6}$$

where w is a weight to find a balance between the two costs. In this study, a weight, w, is taken to be 0.5, giving equal importance for both the fuel and threat cost.

In this classical approach, weights are assigned to each of the normalized objective function. The summation of the weights should be equal to one. One of the main advantages of this approach is that it produces a single solution to the problem but for multiple solutions the problem should be run multiple times with different weight combinations [28]. As the weights are needed to be predefined, it is also called as the priori approach. This is one of the most common and popular ways to solve a multi-objective problem. However, this method has some limitations such as the user needs to assign the weights and it's sometimes difficult to find the right combination of weights. Since the two objectives have relatively equal importance in this study,it is not a major limitation to assign a weight. However, this method does not necessarily produce uniformly distributed Pareto-optimal solutions even if the weights are uniformly distributed. This method does not guarantee to find all the Pareto-optimal solutions for non-convex problems, either.

C. Modified Adaptive Mutation

In our study, the mutation strategies of two of the popular DE variations DE/best/1 and DE/rand/1 are combined to form a single mutation operator, called as the *adaptive variation* in this paper. In the adaptive variation, the mutation strategy is adapted in a way to find a balance between the two of the popular DE variations. An adaptive coefficient (α) is used to strike the balance between the two variations. The adaptive mutation operator is given by equation (7):

$$\begin{aligned} V_{i,G+1} = \alpha \cdot (X_{best,G} + \beta \cdot (X_{r_1^i,G} - X_{r_2^i,G})) + \\ (1 - \alpha) \cdot (X_{r_1^i,G} + \beta \cdot (X_{r_2^i,G} - X_{r_3^i,G})) \end{aligned} \tag{7}$$

where $\alpha \in [0, 1]$ is the adaptive coefficient to adapt between the mutation operation of DE/best/1 and DE/rand/1.

There are few studies which advocate the benefits of combining different forms of mutations for more adaptive DE. In [27], the mutation operator is modified to combine two DE variants. The study in [22] used fuzzy logic controller for the adaptation of DE parameters for optimizing some of the benchmark functions.

In our study, a fuzzy logic controller is used to further get the adaptive coefficient of the mutation operator(α) in eq. (7) as well as the differential weight(β) and the crossover rate (CR).

In general, if the evolution is considered to be in exploration phase, then higher values of β and CR should be considered as it further enhances the exploration capacity of the DE. The value of adaptive coefficient(α) during this phase of exploration should be kept close to zero, as it will help to harness the power of the DE/rand/1 variation. Later, during the exploitation phase, the opposite strategy should be simply considered and the DE/Best/1 mutation operator should be given more weight(α). If it cannot be determined whether the DE is in exploration or exploitation phase, medium values for the parameters should be used from their range. The fuzzy logic controller was designed accordingly. In order to identify the phase of the DE algorithm, the measurement of fitness change(FOC) and that of parameter value change(CPV) were used.

D. Fuzzy Adaptive Controller for DE Parameters

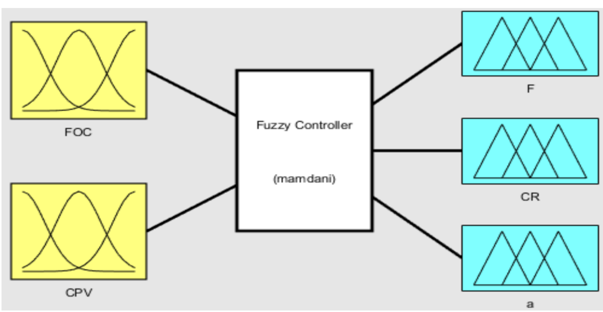

Fig. 2: Fuzzy logic controller

The parameter values for differential weight(β), crossover rate(CR) and the adaptive coefficient(α) are calculated based on the change in the parameter values and fitness values. Two terms are considered for the fitness and parameter value change measurements and are given as input to the fuzzy logic controller. The cumulative change in fitness values and change in parameter values of the entire population are considered after each generation. A similar logic to [22] is used for the fuzzy logic controller in our study.

The proposed fuzzy logic controller has two input variables and three output variables as depicted in Fig. 2. The inputs to the controller are:

1) *FOC*: *Functional Objective Change*
FOC is defined as the cumulative change of the fitness value of the entire population after each iteration. Only the members of the population whose fitness value changed in the next iteration are considered. After each iteration, the value of FOC is calculated using equation (8):

$$FOC = \frac{1}{(NP-c)} \sum_{i=1}^{(NP-c)} (\frac{S^*}{(S_{i_1})} - \frac{S^*}{(S_{i_2})})^2 \qquad (8)$$

where c is the number of individuals in the population whose fitness were unchanged from the previous iteration, NP is the size of population, S^* is the minimum fitness value over both the generations. S^* is used as the base for comparisons of change. S_{i_1} is the fitness of individual i in the previous iteration and S_{i_2} is the fitness of same individual i in the current iteration. This is done to normalize the change in fitness values to $[0, 1]$.

2) *CPV*: *Change in Parameter Values*
Manhattan distance [29] is used for the measurement of the parameter change. Similar to FOC, only those individuals whose parameters actually changed are considered. No normalization to the variable CPV is given and the change is directly given as input to the fuzzy logic controller. After each iteration, the value of CPV is calculated using equation (9):

$$CPV = \frac{1}{(NP-c)} \sum_{i=1}^{(NP-c)} Md_i \qquad (9)$$

where Md_i is the Manhattan distance between the starting and ending coordinates of each line-segments of the two generations for an individual i.

Mamdani-type fuzzy inference system is used for fuzzification. All the input and output membership functions are triangular functions. The controller uses centroid method of defuzzification to produce the crisp outputs. Fuzzy AND operator with fuzzy-*t-norm* of min was used to aggregate the inputs. The fuzzy rules are in the fuzzy *If-Then* statements. Our fuzzy rule_base of the fuzzy logic controller is given in Table II. For instance,

$Rule$ 3: **if** FOC is *small* and CPV is *big*
then β is *big* and CR is *big* and α is *medium*.

TABLE II: Fuzzy Rule_Base

Inputs		Outputs		
FOC	CPV	β	CR	α
small	small	small	small	big
small	medium	medium	medium	medium
small	big	big	big	medium
medium	small	medium	medium	medium
medium	medium	medium	medium	medium
medium	big	big	big	medium
big	small	big	big	medium
big	medium	big	big	medium
big	big	big	big	small

All of two inputs and three outputs fuzzy variables are described using three fuzzy values *small, medium*, and *big* which are defined with their membership functions in the corresponding universe of discourse. The range of the universe of discourse in the membership function values are listed in Table III.

TABLE III: Fuzzy membership Functions

Inputs, Outputs	Membership functions	Range of Values
FOC	small	0 - 0.15
	medium	0.07 - 0.4
	big	0.2 - 1.0
CPV	small	0 - 15
	medium	8 - 30
	big	20 - 100
β	small	0.06 - 0.16
	medium	0.1 - 0.6
	big	0.5 - 1.0
CR	small	0.1 - 0.4
	medium	0.3 - 0.6
	big	0.5 - 0.8
α	small	0.05 - 0.2
	medium	0.1 - 0.8
	big	0.6 - 0.9

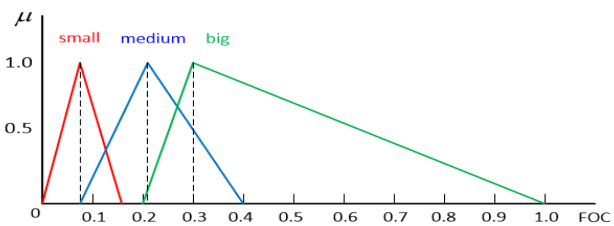

Fig. 3: Membership functions for FOC

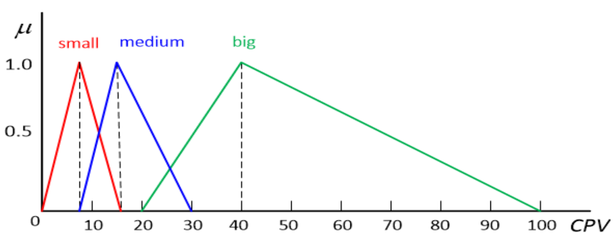

Fig. 4: Membership functions for CPV

As illustrated in Figure 2, the inputs to the controller are *FOC* and *CPV* while its outputs are the parameters β, CR and α. Figures 3, 4, 5, 6 and 7 depict the membership functions of the inputs variables (FOC and CPV) and the outputs variables (β, CR, α).

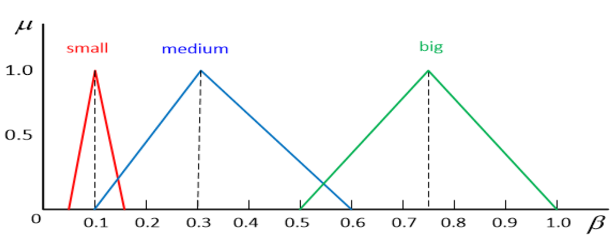

Fig. 5: Membership functions for β

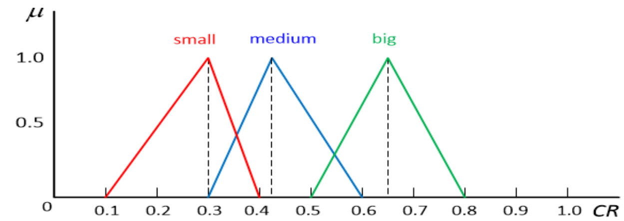

Fig. 6: Membership functions for CR

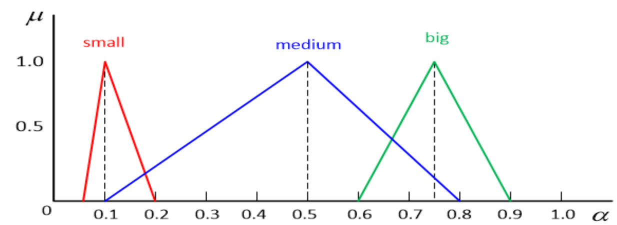

Fig. 7: Membership functions for α

V. Experiments and Results

All simulations and implementations are done in MATLAB. Matlabs fuzzy logic toolbox was used only for the design of the fuzzy logic controller.

Figure 8 shows the environment for the UAV with the paths from different DE variations. The environment had eight threat zones whose locations are provided in Table IV. All threat zones are considered spheres. The environment for UAV was consider to be of space 100 miles by 100 miles with an altitude of 100 meters.

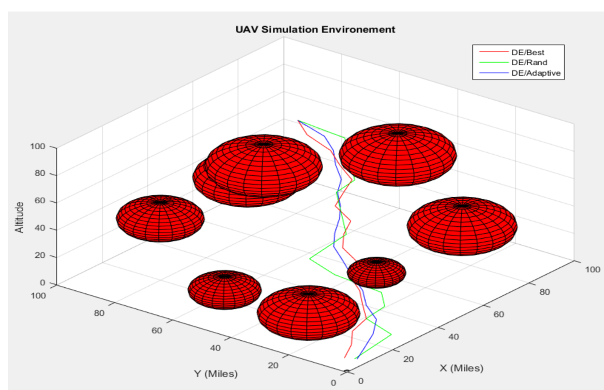

Fig. 8: UAV path planning results

TABLE IV: Threat locations

No.	X	Y	Z	Radius
1	15	25	15	14
2	10	50	20	10
3	20	80	45	12
4	40	60	80	16
5	65	85	35	15
6	80	45	65	16
7	45	25	20	8
8	70	15	40	15

TABLE V: Comparison of Results

No.	β	CR	No. of gen	DE/Best/1	DE/Rand/1	FA-DE
1	1.2	0.6	100	87.7566	87.7566	77.0141
2	0.9	0.6	100	82.2552	89.9539	77.5284
3	0.8	0.7	100	79.9424	88.8052	78.4914
4	0.6	0.4	100	78.7735	85.146	78.4083
5	0.25	0.6	100	76.9872	76.9152	77.3149
6	0.2	0.6	100	81.6830	79.7885	78.3502
7	0.15	0.6	100	79.6605	80.9921	79.2697

Our fuzzy adaptive DE (FA-DE) approach is compared with DE/best/1 and DE/rand/1 versions. Our fuzzy adaptive DE, in general, showed better or relatively similar performance than both variations of DE. DE/best/1 was the second best performing algorithm for this problem. Different values of the initial parameters were tested. For our fuzzy adaptive DE, the initial values did not have much impact on the final performance as the parameters would be adapted after each iteration. In the experiment, the size of initial population ($NP = 15$) and the number of sub-paths was kept constant ($m = 18$). The results are summarized in Table V. Table VI gives the average parameter values during the evolution process for 100 generations.

TABLE VI: Average Parameter Values in Fuzzy Adaptive DE

Parameters	Average Values
β	0.4410
CR	0.4861
α	0.5714

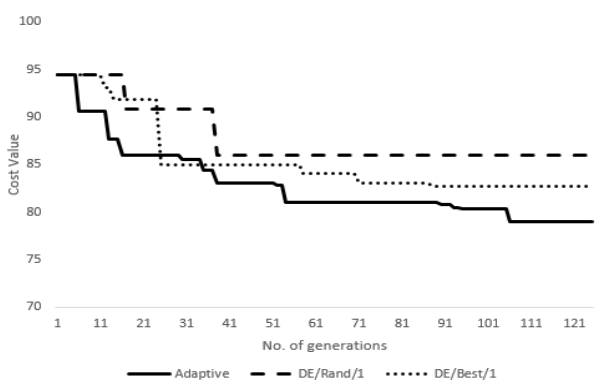

Fig. 9: The evolution curves of three algorithms

VI. Conclusion

With the fuzzy adaptive DE, the search space will be better explored and exploited. The search space will be better explored because the parameters will be adapted to facilitate the exploration of the search space for all the members of the population. After the members are unable to further explore for better solutions, the parameters will be adjusted to focus on the exploitation of the current solutions. It could have an impact on the speed of convergence to give all the individuals of the population a chance to explore and exploit the solution. Chances of premature convergence are reduced by using this approach, however, it is still not fully overcome. The relative change of the parameters and fitness values after each successive generation is taken into consideration, if no member of the population shows any significant progress in the solution for a generation. Then, this method can get stuck in the local minimum.

The fuzzy adaptive differential evolution was able to perform relatively better than other classical versions of DE. With fuzzy adaptive DE, in addition, there will be no need of the tedious process of finding acceptable parameter values, as the parameters would be automatically adapted. Path planning for UAVs with dynamic obstacles is considered for the future study. Further generalization of this adaptive method for other problems will be also studied. In addition, fuzzy adaptive particle swarm optimization (PSO) for the optimization of UAV path planning will be designed for the comparisons of performance.

References

[1] J. S. Zelek, "Dynamic path planning," in *Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on*, vol. 2. IEEE, 1995, pp. 1285–1290.

[2] B. Coifman, M. McCord, M. Mishalani, and K. Redmill, "Surface transportation surveillance from unmanned aerial vehicles," in *Proc. of the 83rd Annual Meeting of the Transportation Research Board*, 2004.

[3] J. M. M. Neto, R. A. da Paixão, L. R. L. Rodrigues, E. M. Moreira, J. C. J. dos Santos, and P. F. F. Rosa, "A surveillance task for a uav in a natural disaster scenario," in *Industrial Electronics (ISIE), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 1516–1522.

[4] C. Bolkcom, "Homeland security: Unmanned aerial vehicles and border surveillance." DTIC Document, 2004.

[5] S. Herwitz, L. Johnson, S. Dunagan, R. Higgins, D. Sullivan, J. Zheng, B. Lobitz, J. Leung, B. Gallmeyer, M. Aoyagi *et al.*, "Imaging from an unmanned aerial vehicle: agricultural surveillance and decision support," *Computers and electronics in agriculture*, vol. 44, no. 1, pp. 49–61, 2004.

[6] G. Wang, L. Guo, H. Duan, H. Wang, L. Liu, and M. Shao, "A hybrid metaheuristic de/cs algorithm for ucav three-dimension path planning," *The Scientific World Journal*, vol. 2012, 2012.

[7] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353, 1965.

[8] T. J. Ross, *Fuzzy logic with engineering applications*. John Wiley & Sons, 2009.

[9] G. Klir and B. Yuan, *Fuzzy sets and fuzzy logic*. Prentice hall New Jersey, 1995, vol. 4.

[10] H. Singh, M. M. Gupta, T. Meitzler, Z.-G. Hou, K. K. Garg, A. M. Solo, and L. A. Zadeh, "Real-life applications of fuzzy logic." *Adv. Fuzzy Systems*, vol. 2013, pp. 581 879–1, 2013.

[11] R. Kruse, C. Borgelt, F. Klawonn, C. Moewes, G. Ruß, M. Steinbrecher *et al.*, *Computational Intelligence*. Springer, 2011.

[12] R. Storn and K. Price, "Differential evolutiona simple and efficient adaptive scheme for global optimization over continuous spaces, berkeley," *CA: International Computer Science Institute*, 1995.

[13] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 61–106, 2010.

[14] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE transactions on evolutionary computation*, vol. 15, no. 1, pp. 4–31, 2011.

[15] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[16] V. Gonuguntla, R. Mallipeddi, and K. C. Veluvolu, "Differential evolution with population and strategy parameter adaptation," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[17] D. Zaharie, "Influence of crossover on the behavior of differential evolution algorithms," *Applied Soft Computing*, vol. 9, no. 3, pp. 1126–1138, 2009.

[18] K. Y. Kok and P. Rajendran, "Differential-evolution control parameter optimization for unmanned aerial vehicle path planning," *PloS one*, vol. 11, no. 3, p. e0150558, 2016.

[19] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, and M. Shao, "Path planning for uninhabited combat aerial vehicle using hybrid meta-heuristic de/bbo algorithm," *Advanced Science, Engineering and Medicine*, vol. 4, no. 6, pp. 550–564, 2012.

[20] G.-G. Wang, H. E. Chu, and S. Mirjalili, "Three-dimensional path planning for ucav using an improved bat algorithm," *Aerospace Science and Technology*, vol. 49, pp. 231–238, 2016.

[21] Z. Zhou, H. Duan, P. Li, and B. Di, "Chaotic differential evolution approach for 3d trajectory planning of unmanned aerial vehicle," in *Control and Automation (ICCA), 2013 10th IEEE International Conference on*. IEEE, 2013, pp. 368–372.

[22] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Computing*, vol. 9, no. 6, pp. 448–462, 2005.

[23] H. Duan, Y. Yu, X. Zhang, and S. Shao, "Three-dimension path planning for ucav using hybrid meta-heuristic aco-de algorithm," *Simulation Modelling Practice and Theory*, vol. 18, no. 8, pp. 1104–1115, 2010.

[24] Y. Zhang, L. Wu, and S. Wang, "Ucav path planning by fitness-scaling adaptive chaotic particle swarm optimization," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[25] J. L. Foo, J. Knutzon, J. Oliver, and E. Winer, "Three-dimensional path planning of unmanned aerial vehicles using particle swarm optimization," in *11th AIAA/ISSMO multidisciplinary analysis and optimization conference*, 2006, p. 6995.

[26] Z. Qian, J. Wang, G. Wang, and Y. Shi, "Route planning of uav based on improved ant colony algorithm," 2015.

[27] X. Zhang, J. Chen, B. Xin, and H. Fang, "Online path planning for uav using an improved differential evolution algorithm," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 6349–6354, 2011.

[28] A. Konak, D. W. Coit, and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability Engineering & System Safety*, vol. 91, no. 9, pp. 992–1007, 2006.

[29] P. E. Black, "Manhattan distance," *Dictionary of Algorithms and Data Structures*, vol. 18, p. 2012, 2006.