A multi-modal optimization approach to single path planning for unmanned aerial vehicle

Yang, Peng; Lu, Guanzhou; Tang, Ke; Yao, Xin

DOI:
10.1109/CEC.2016.7743998

A Multi-Modal Optimization Approach to Single Path Planning for Unmanned Aerial Vehicle

Peng Yang[1], Guanzhou Lu[1*], Ke Tang[1], and Xin Yao[2]
[1]USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications (UBRI),
School of Computer Science and Technology, University of Science and Technology of China, Hefei, China, 230027.
[2]The Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA),
School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K.
* Corresponding Author.
Emails: trevor@mail.ustc.edu.cn; lrlgz@ustc.edu.cn; ketang@ustc.edu.cn; x.yao@cs.bham.ac.uk

Abstract

In the past few years, Evolutionary Algorithms (EAs) based UAV path planners have drawn increasing research interests. However, they are not scalable to large-scale problems, i.e., lots of waypoints. Recently, we have proposed a novel EA-based framework, named Separately Evolving Waypoints (SEW), that can deal with large-scale problems. However, the difficulty of UAV path planning depends not only on the number of waypoints, but on the number of constraints it has to satisfy, especially the number of obstacles. In particular, the number of waypoints required is also partly determined by the number of constraints. Hence, it is critical to further improve SEW with respect to large number of obstacles. Originally, a state-of-the-art global optimization approach is employed. In this work, we discuss how the increasing number of obstacles will deteriorate the performance of the global optimizer, then we propose multi-modal optimization approaches that facilitates the performance of SEW against large number of obstacles.

Index Terms

Multi-Modal Optimization, Unmanned Aerial Vehicle, Path Planning, Separately Evolving Waypoints.

I. Introduction

Autonomous path planning techniques have shown increasingly importance to Unmanned Aerial Vehicles (UAVs), since the traditional remotely piloted control can hardly offer sufficient accuracy and perfect timing for complex missions. Generally, the path planning problem for a UAV is to find a sequence of waypoints to optimize a number of objectives subjecting to various constraints [1], [2].

In the past few years, Evolutionary Algorithms (EAs) based UAV path planners have drawn increasing research interests as they are more flexible and effective than the traditional mathematical programming methods [2]. Nevertheless, existing EA-based planners deteriorated rapidly as the number of waypoints increased. This is caused by regarding an integrated path as a candidate solution [3]. Generally, a path is feasible only if all its waypoints are feasible. However, it is usually the case that waypoints in a candidate path may be of different qualities, especially when the number of waypoints is large. Unfortunately, the existing EA-based planners cannot identify such differences of the waypoint qualities during the search. As a consequence, all waypoints of an infeasible path will be regarded as infeasible waypoints and vice versa. Eventually, the lack of capability of exploiting high quality waypoints leads ineffective search when the number of waypoints increases in the existing EA-based planners..

To solve that drawback, we have recently proposed a novel EA-based framework, named Separately Evolving Waypoints (SEW) [3]. The main idea of SEW is to decompose the original problem that finds an optimal path into a set of sub-problems, each of which aims to find an optimal waypoint of the path separately. On this basis, the waypoints of a path will be evolved separately by EAs and high quality waypoints can be explicitly exploited to guide further evolution. Simulation results have verified the significant advantages of SEW over the existing EA-based planners on several tested scenarios, in terms of solution quality and computational time.

However, the difficulty of UAV path planning depends not only on the number of waypoints, but on the number of constraints, especially the number of obstacles. This is because the increase of the number of obstacles will constrain the search space with more infeasible areas. Besides, more waypoints may be required to keep the path sufficiently smooth and flyable as the feasible passageway for the path becomes narrower and more zigzagged. On this basis, it is of importance to further improve SEW against the scenarios with large number of obstacles. In this work, we will restrict our studies on the evolution of sub-problems. In [3], a state-of-the-art Global Optimization Approach (GOA), i.e., JADE [4], is chosen. Here we discuss how the increase in the number of obstacles will deteriorate the performance of GOAs and propose that Multi-Modal Optimization Approaches (MMOAs) can solve the sub-problems more effectively.

As discussed above, the increase of obstacles will result in the increase of infeasible areas, where it is more difficult for the population to identify the decreased feasible areas, and usually insufficient useful fitness information. As the evolution of

GOAs is mostly driven by fitness differences of candidate solutions, the GOAs will easily converge or stagnate in the infeasible areas of some sub-problems, due to the lack of evolution motivation. Consequently, the whole search will be ineffective.

Distinct from GOAs whose evolution is only motivated by fitness differences of candidate solutions, the evolution of MMOAs is also encouraged by some diversity maintenance techniques, e.g., niching [5], [6], despite some potential issues [7]. This is because MMOAs aim to search different areas of the solution space to locate multiple diverse optimal solutions simultaneously. On this basis, the explicit force, which pushes the candidate waypoints away from each other to avoid overlapping search, can be utilized as alternative motivation for guiding the evolution when insufficient fitness information can be provided. To verify the proposed viewpoint, we select a recently proposed MMOA, named Negatively Correlated Search (NCS) [8], to evolve the waypoints in each sub-problem. The proposed planner is named SEW-NCS-C. A set of simulation studies shows that SEW-NCS-C can outperform the original SEW-JADE in [3] significantly.

The rest of this paper is organized as follows. In Section II, the UAV path planning problem is briefly introduced. In Section III, the SEW framework is presented. After that, the proposed SEW-NCS-C is detailed in IV. Simulated results and analyses are given in Section IV. Section V provides the conclusions and discussions of this work.

II. UAV Path Planning

The UAV path planning problem can be formulated as an optimization problem that finds out a sequence of waypoints to optimize a number of objectives while subjecting to various constraints. Here we briefly list corresponding criteria functions as follows to describe the problem. More details of descriptions and technical justifications can be referred to [1]–[3].

A. Objective Functions

1) *Minimal Path Length Ratio (MPLR)*: For many missions, the paths are required to be short enough to satisfy the fuel or time-window constraints. Hence, it needs to be minimized.

$$o_1 = \frac{\sum_{i=2}^{N_w} \|\mathbf{W}_i - \mathbf{W}_{i-1}\|}{\|\mathbf{W}_n - \mathbf{W}_1\|}, \tag{1}$$

where N_w indicates the number of waypoints of a path, which is fixed during the path planning. $\mathbf{W}_i$ is the 3-D position of the i^{th}, $i = 2, 3, ..., N_w - 1$, waypoint in Cartesian coordinate system, which can be also expressed as (x_i, y_i, z_i).

2) *Minimal Risk of Kill (MRK)*: The hostile missiles will impose risk on UAVs once the UAVs have entered into the their range. Hence, to guarantee a safe flight, the path should avoid the range of hostile missiles, i.e., minimized the risk of kill.

$$o_2 = \sum_{i=2}^{N_w} \sum_{j=1}^{N_d} \sum_{k=1}^{M} RK_{ij}^k \qquad \text{with}$$
$$RK_{ij}^k = \begin{cases} \frac{1}{((dis_{ij}^k)^4} & \textbf{if} \quad dis_{ij}^k \leq R_{RKmax}^k \\ 0 & \textbf{otherwise} \end{cases} \tag{2}$$

R_{RKmax}^k is the maximal risk distance of the k^{th} enemy missiles, where $k = 1, 2, ..., M$, and M is the number of enemy missiles. dis_{ij}^k indicates the distance between a dividing point $\mathbf{D}_{ij}$ and the k^{th} missile $\mathbf{M}_k$, $j = 1, 2, ..., 6, k = 1, 2, ..., M$. The dividing point $\mathbf{D}_{ij}$ is used to estimate the behaviors of segments between two adjacent waypoints [3] and is defined as:

$$\mathbf{D}_{ij} = \mathbf{W}_{i-1} + j * (\mathbf{W}_i - \mathbf{W}_{i-1})/N_d, \tag{3}$$

3) *Minimal Risk of Radar Detection (MRRD)*: A UAV is usually asked to keep stealthy during the missions. Therefore, the risk of radar detection should be minimized.

$$o_3 = \sum_{i=2}^{N_w} \sum_{j=1}^{N_d} \sum_{k=1}^{R} RRD_{ij}^k \qquad \text{with}$$
$$RRD_{ij}^k = \begin{cases} \frac{1}{((dis_{ij}^k)^4} & \textbf{if} \quad dis_{ij}^k \leq R_{RRDmax}^k \\ 0 & \textbf{otherwise} \end{cases} \tag{4}$$

where R^k_{RRDmax} is the maximal risk distance of the k^{th} enemy radar, where $k = 1, 2, ..., R$, and R is the number of enemy missiles. dis^k_{ij} indicates the distance between a dividing point $\mathbf{D}_{ij}$ and the k^{th} radar $\mathbf{R}_k$,$j = 1, 2, ..., 6, k = 1, 2, ..., R$.
4) *Minimal Flight Altitude (MFA)*: A UAV sometimes needs to fly at a low altitude to be a greater threat to the enemy on the ground. On this basis, the flight altitude should be minimized.

$$o_4 = \sum_{i=2}^{N_w-1} FA_i \qquad \text{with}$$
$$FA_i = \begin{cases} 0 & \textbf{if} \quad z_i \leq \text{map}(x_i, y_i) \\ \frac{z_i - \text{map}(x_i, y_i)}{N_w - 2} & \textbf{otherwise} \end{cases} \tag{5}$$

where map(x_i, y_i) is a function that returns the height of the position (x_i, y_i).

B. Constraint Functions

1) *Minimal Turning Angle (MTA)*: Subject to the maneuverability of a UAV, a path should be sufficiently smooth. This requires the turning angle of the UAV at any waypoint to be smaller than a predefined threshold.

$$c_1 = 0 \quad \text{where} \quad c_1 = \sum_{i=2}^{N_w-1} TA_i \quad \text{with}$$
$$TA_i = \begin{cases} e^{\frac{\theta_i - \theta_{max}}{\pi - \theta_{max}}} & \textbf{if} \quad \theta_i > \theta_{max} \\ 0 & \textbf{otherwise} \end{cases} \tag{6}$$

where θ_{max} is the upper bound of the turning angle. θ_i is the turning angle at the i^{th} waypoint $\mathbf{W}_i$, and can be calculated as follow,

$$\theta_i = \arccos \frac{(x_i - x_{i-1}, y_i - y_{i-1}) \cdot (x_{i+1} - x_i, y_{i+1} - y_i)}{\|x_i - x_{i-1}, y_i - y_{i-1}\| \|x_{i+1} - x_i, y_{i+1} - y_i\|}, \tag{7}$$

2) *Limited Slope (LS)*: Similar to the turning angle, the slope of the UAV should also be kept under the predefined threshold.

$$c_2 = 0 \quad \text{where} \quad c_2 = \sum_{i=2}^{N_w} LS_i \quad \text{with}$$
$$LS_i = \begin{cases} e^{\frac{\alpha - r_i}{\frac{\pi}{2} + \alpha}} & \textbf{if} \quad r_i < \alpha \\ e^{\frac{r_i - \beta}{\frac{\pi}{2} - \beta}} & \textbf{if} \quad r_i > \beta \\ 0 & \textbf{otherwise} \end{cases} \tag{8}$$

where α and β are the maximal diving and climbing angle, respectively. r_i is the slope at the i^{th} waypoint $\mathbf{W}_i$, and can be calculated as follow,

$$r_i = \arccos \frac{z_i - z_{i-1}}{\|x_i - x_{i-1}, y_i - y_{i-1}\|}, \tag{9}$$

3) *Limited Terrain (LT)*: A UAV is physically infeasible to go through the terrain. Hence, a path is restricted to be above the terrain.

$$c_3 = 0 \quad \text{where} \quad c_3 = \sum_{i=2}^{N_w} \sum_{j=1}^{N_d} LT_{ij} \quad \text{with}$$
$$LT_{ij} = \begin{cases} 1 & \textbf{if} \quad \mathbf{D}_{ij}^z \leq \text{map}(\mathbf{D}_{ij}^x, \mathbf{D}_{ij}^y) \\ 0 & \textbf{otherwise} \end{cases} \tag{10}$$

3) *Limited Map (LM)*: A UAV is not allowed to fly outside the investigated mission space, where unexpected dangers may highly likely conceal.

$$c_4 = 0 \quad \text{where} \quad c_4 = \sum_{i=2}^{N_w} \sum_{j=1}^{N_d} LM_{ij} \quad \text{with}$$
$$LM_{ij} = \begin{cases} 0 & \textbf{if} \quad \text{InRange}(\mathbf{D}_{ij}^x, \mathbf{D}_{ij}^y) \\ 1 & \textbf{otherwise} \end{cases} \tag{11}$$

$$\text{InRange}(x, y) = (l_x \leq x \leq h_x) \wedge (l_y \leq y \leq h_y), \tag{12}$$

where $[l_x, h_x]$ and $[l_y, h_y]$ indicate the bounds of x coordinate and y coordinate, respectively.

III. The Separately Evolving Waypoints Framework

We have recently proposed a novel framework, named Separately Evolving Waypoints (SEW) [3] to scale up the EA-based UAV path planners. The main idea of SEW is to decompose the original problem into a set of sub-problems, each aims to find an optimal waypoint that constructs the path in the end. In this case, the waypoints along a path will be evolved separately by EAs and high quality waypoints can be explicitly exploited to guide the further evolution.

Concretely, N_p candidate waypoints are first randomly initialized for each sub-problem, separately. At each iteration of the optimization, the sub-problems are sequentially evolved in an ascending order, i.e., from the start to the destination. For each sub-problem, N_p new waypoints are first generated in terms of a specific search operator based on the current N_p candidate waypoints. After that, these $2N_p$ waypoints are evaluated and N_p waypoints with the highest qualities are preserved for next iteration. Notice that, a planner is required to output an integrated path after the iterative search terminates. However, SEW does not explicitly evolve paths. Instead, SEW updates one path for output iteratively in terms of the evolved waypoints. Specifically, at each iteration, the i^{th} waypoint of the path will be updated with the best found waypoint in the i^{th} sub-problem just after it is evolved.

When planning a path for a UAV, several key factors should be considered for evaluation, such as maneuverability of the UAV, the spatial environment, safety and cost of the path. Commonly adopted criteria can be seen in Section II and detailed descriptions and technical justifications can be found in [1]–[3]. However, the existing objective functions and constraints cannot be adopted in SEW, since they are used to evaluate an integrated path. Fortunately, those functions are actually decomposable on waypoints. The reason is quite intuitive that the existing objectives and constraints restrict the flight in a manner of geometrical relationships, where most objects concerned are points, segments and angles. Those geometrical relationships are actually all decomposable on points, i.e., waypoints. On this basis, we suggest a rule geometrically decomposing the original evaluation functions in terms of the current waypoint associated with the other waypoints in the updated path if necessary. Based on this rule, the new evaluation functions for waypoints can be easily derived to evaluate the waypoints. To be specific, the derivations are obtained by removing the summation items on waypoints of the original functions, except for the $MPLR$ as it is for the whole path length. To evaluate the i^{th} waypoint, we first substitute it in the updated path as $[\mathbf{W}_1^*, \mathbf{W}_2^*, ..., \mathbf{W}_i, ..., \mathbf{W}_{N_w}^*]$. After that, the following decomposed functions are applied to $\mathbf{W}_i$ associated with the other waypoints in the path. The decomposed functions on the i^{th} waypoint $\mathbf{W}_i$ are briefly listed as follows.

$$o_1^i = \frac{\sum_{i=2}^{N_w} \|\mathbf{W}_i - \mathbf{W}_{i-1}\|}{\|\mathbf{W}_n - \mathbf{W}_1\|}, \tag{13}$$

$$o_2^i = \sum_{j=1}^{N_d} \sum_{k=1}^{M} RK_{ij}^k, \tag{14}$$

$$o_3^i = \sum_{j=1}^{N_d} \sum_{k=1}^{R} RRD_{ij}^k, \tag{15}$$

$$o_4^i = FA_i, \tag{16}$$

$$c_1^i = 0 \quad \text{where} \quad c_1^i = TA_i, \tag{17}$$

$$c_2^i = 0 \quad \text{where} \quad c_2^i = LS_i, \tag{18}$$

$$c_3^i = 0 \quad \text{where} \quad c_3^i = \sum_{j=1}^{N_d} LT_{ij}, \tag{19}$$

$$c_4^i = 0 \quad \text{where} \quad c_4^i = \sum_{j=1}^{N_d} LM_{ij}. \tag{20}$$

IV. The Proposed Planner

A. MMOAs can be helpful to SEW

Apart from the evaluation of candidate waypoints, evolving the waypoints is also a non-trivial problem. In [3], we simply chose a state-of-the-art global optimization approach, i.e., JADE [4]. In this section, we propose that MMOAs can solve the sub-problems more effectively.

As the number of obstacles increases, the infeasible areas will increase, where waypoints are more likely to be infeasible. Generally, the GOAs have the nature of convergence for refining a single solution, which is usually guided by the fitness information. As feasible areas are usually of much higher fitness values than the infeasible areas, the feasible areas in a sub-problem will quickly attract the whole candidate waypoints to converge. This may highly likely lead the whole population to be prematurely trapped in a local optimum. Even worse, if the population is failed to identify the feasible areas, there will be insufficient fitness information to guide the convergence. Therefore, the population will easily stagnate in the infeasible areas of some sub-problems. Consequently, the whole path will be infeasible.

On the contrary, MMOAs aim to search for multiple diverse optimal solutions in a single run [6]. For that purpose, the population of MMOAs is diversified to search different areas of the solution space simultaneously by some diversity maintenance techniques [5], [6]. On this basis, by performing diverse search, the candidate solutions will be pushed away from each other to avoid overlapping search. This can be beneficial to guide the evolution of the candidate waypoints when there is little fitness information. In this case, they are less likely to converge or stagnate in the infeasible areas.

There have been several MMOAs in the literature, e.g., niche based methods [9] and clustering based methods [10]. At each iteration, they first select a set of candidate solutions that are distant from each other in the solution space. After that, these solutions are utilized to generate new solutions. However, they all neglect the fact that a set of diverse (distant) solutions does not necessarily generate a diverse set of new solutions, as the latter also relies on the search operators used and the problem structure [7]. Recently, a novel MMOA named NCS has been proposed [8]. Distinct from the existing MMOAs that define the diversity as diversity among individuals, NCS defines the diversity as diversity among search behaviors. The search behaviors describe how new solutions will be produced based on the parent solutions and thus can capture the on-going interaction between search operators and solutions. NCS expects the search behaviors of different candidate solutions to show negative correlation [11]. In this way, NCS explicitly favors solutions that are more likely to move towards a region that is both of high quality and is unlikely to be searched by other solutions. Since the ultimate goal of employing MMOAs in SEW is to promote the waypoints in a sub-problem to be at high quality and diverse positions, NCS is more directly related to this goal and thus is expected to facilitate the search better.

B. Negatively Correlated Search

NCS first randomly initializes N_p Randomized Local Search (RLSs), each of which produces one new candidate solution in each iteration. These N_p RLSs parallel and iteratively update each candidate solution with a randomized search operator until a predefined halting condition is satisfied.

Suppose at the t^{th} iteration of NCS, N_p new solutions have been generated by applying some randomized search operators to N_p current solutions, respectively. Generally, the search bias of an RLS in the $(t+1)^{th}$ iteration can be described as the

probability distribution, from which a new solution will be sampled. Therefore, if the probability distribution corresponding to an RLS differs from those corresponding to the other RLSs, it is highly likely to generate new solutions in a region that is not covered by other RLSs. In [8], the Bhattacharyya Distance [12] is adopted to measure the difference or correlation between two probability distributions. As the waypoints are described in continuous domain, Eqs.(21) gives the Bhattacharyya distance for continuous probability distributions:

$$D_B(p_i, p_j) = -\ln\left(\int \sqrt{p_i(\mathbf{x})p_j(\mathbf{x})}\mathrm{d}\mathbf{x}\right) \tag{21}$$

where p_i and p_j denote the probability density functions of two distributions.

Let $\mathbf{x}_i$ denote the current solution obtained by the i^{th} RLS and $\mathbf{x}'_i$ denote the new solution generated based on $\mathbf{x}_i$. At each iteration, only one of the two solutions will be remained for the next iteration. The probability distribution corresponding to the i^{th} RLS in the next iteration depends on both the search operator and the solution remained. Therefore, by remaining either $\mathbf{x}_i$ or $\mathbf{x}'_i$, the i^{th} RLS actually chooses one of the corresponding distributions, denoted as p_i and p'_i. Ideally, the selected solution should be with high quality and should lead to a distribution that differs from those associated with the other RLSs. In NCS, the former is measured with the value of the fitness function, denoted as $f(\mathbf{x})$. The latter is defined as:

$$Corr(p_i) = \min_j\{D_B(p_i, p_j)|j \neq i\} \tag{22}$$

where p_j represents the distributions corresponding to other RLSs and a larger $Corr(p_i)$ is preferred.

Since $f(\mathbf{x})$ and $Corr(p)$ may be of different scales, they should first be normalized to make it easier to determining an appropriate trade-off for above cases. The normalization is conducted by requiring $f(\mathbf{x}_i) + f(\mathbf{x}'_i)$ and $Corr(p_i) + Corr(p'_i)$ equal to 1. With the normalization step, it is unnecessary to consider $f(\mathbf{x}_i)$ and $Corr(p_i)$ for deciding which solution to discard/remain because they now equal to $1 - f(\mathbf{x}'_i)$ and $1 - Corr(p'_i)$, respectively. On one hand, the smaller the $f(\mathbf{x}'_i)$, the better $\mathbf{x}'_i$ is from the viewpoint of solution quality (considering minimization cases). On the other hand, the larger the $Corr(p'_i)$, the less likely that the new solution produced by $\mathbf{x}'_i$ are similar to the solutions generated by other RLSs. As a result, NCS prefers the solution $\mathbf{x}'_i$ associated with small $f(\mathbf{x}'_i)$ and large $Corr(p'_i)$. On this basis, the following heuristic rule is adopted to aggregate the solution quality and correlation between RLSs into one selection criterion:

$$\begin{cases} \text{discard } \mathbf{x}_i, & \textbf{if} \quad \frac{f(\mathbf{x}'_i)}{Corr(p'_i)} < \lambda \\ \text{discard } \mathbf{x}'_i, & \textbf{otherwise} \end{cases} \tag{23}$$

where $\lambda \in (0, +\infty)$ is a parameter.

When instantiating NCS, the randomized search operator and the parameter λ should be specified. Generally, different search operators will lead to different search behaviors and different values of λ may lead to different decisions on which solution should be remained/discarded. Both of them can affect the search process and consequently the performance of NCS. In this work, we directly adopt the ones used in an instantiation of NCS, i.e., NCS-C [8].

In our case, the Gaussian mutation operator [13] is employed as the search operator for all RLSs of NCS-C. Given the current solution $\mathbf{x}_i$, the Gaussian mutation operator generates a new solution $\mathbf{x}'_i$ in the following way:

$$x'_{id} = x_{id} + \mathcal{N}(0, \sigma_i), \tag{24}$$

where x_{id} denotes the d^{th} variable of $\mathbf{x}_i$ and $\mathcal{N}(0, \sigma_i)$ denotes a Gaussian random variable with zero mean and standard deviation σ_i. All RLSs in NCS-C are initialized with the same value of σ_i. Then, each σ_i is adapted for every $epoch$ iterations according to the 1/5 successful rule suggested in [13], as given in Eq.(25):

$$\sigma_i = \begin{cases} \frac{\sigma_i}{r} & \textbf{if} \quad \frac{c}{epoch} > 0.2 \\ \sigma_i * r & \textbf{if} \quad \frac{c}{epoch} < 0.2 \\ \sigma_i & \textbf{if} \quad \frac{c}{epoch} = 0.2 \end{cases} \tag{25}$$

where r is a parameter that is suggested to be set beneath 1, and c is the times that a replacement happens (i.e., $\mathbf{x}'_i$ is preserved) during the past $epoch$ iterations.

According to Eq.(24), the probability distribution of each $\mathbf{x}_i$ follows a multivariate normal distribution. The expectation of the distribution is $\mathbf{x}_i$ and the covariance matrix $\mathbf{\Sigma_i}$ is $\sigma_i^2\mathbf{I}$, where $\mathbf{I}$ is the identity matrix of size D, i.e., the dimensionality of problem. For this case, the Bhattacharyya distance between two solutions $\mathbf{x}_i$ and $\mathbf{x}_j$ can be directly written as Eq.(26):

$$\begin{aligned} D_B(p_i, p_j) = \quad & \tfrac{1}{8}(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{\Sigma}^{-1}(\mathbf{x}_i - \mathbf{x}_j) \\ & + \tfrac{1}{2}\ln\left(\frac{\det \mathbf{\Sigma}}{\sqrt{\det \mathbf{\Sigma_i} \det \mathbf{\Sigma_j}}}\right), \end{aligned} \tag{26}$$

where $\mathbf{\Sigma} = \frac{\mathbf{\Sigma_i}+\mathbf{\Sigma_j}}{2}$.

The parameter λ is suggested to be time-variant to suit different stages of the search. Specifically, at the t^{th} iteration, λ is given as follow:

$$\lambda_t = \mathcal{N}(1, 0.1 - 0.1 * \frac{t}{T_{max}}), \tag{27}$$

where T_{max} is the user-defined total number of iterations for an execution of NCS-C.

By employing NCS-C to evolve the sub-problems of SEW, we propose a new planner, named SEW-NCS-C. The specification of SEW-NCS-C is described in the next subsection.

Algorithm 1 SEW-NCS-C(T_{max}, $\boldsymbol{\sigma}$, r, $epoch$, N_p, N_w)

Randomly generate N_p waypoints for N_w sub-problems as $\mathbf{W}_i^j, i = 1, 2, ..., N_w, j = 1, 2, ..., N_p$.
For $i = 1$ **to** N_w
  For $j = 1$ **to** N_p
    Compute $f(\mathbf{W}_i^j)$.
  Record the i^{th} waypoint of *BestPath*: $\mathbf{W}_i^* \leftarrow \underset{\mathbf{W}_i^j, j\in\{1,2,...,N_p\}}{\arg\min} \{f(\mathbf{W}_i^j)\}$
Set $t \leftarrow 0$
While $(t < T_{max})$ **do**
  Set $\lambda_t \leftarrow \mathcal{N}(1, 0.1 - 0.1 * \frac{t}{T_{max}})$.
  For $i = 1$ **to** N_w
    For $j = 1$ **to** N_p
      Generate a new waypoint $\mathbf{W}_i^{j'}$ by applying Gaussian mutation operator with σ_{ij} to $\mathbf{W}_i^j$.
      Compute $f(\mathbf{W}_i^j), f(\mathbf{W}_i^{j'}), Corr(p_i^j), Corr(p_i^{j'})$.
    For $j = 1$ **to** N_p
      If $f(\mathbf{W}_i^{j'}) < f(\mathbf{W}_i^*)$
        Update $\mathbf{W}_i^*$, with $\mathbf{W}_i^{j'}$.
      If $\frac{f(\mathbf{W}_i^{j'})}{Corr(p_i^{j'})} < \lambda_t$
        Update $\mathbf{W}_i^j$ with $\mathbf{W}_i^{j'}$.
  $t \leftarrow t + 1$
  If mod$(t, epoch)$ = 0
    For $i = 1$ **to** N_w
      For $j = 1$ **to** N_p
        Update σ_{ij} for the j^{th} RLS in the i^{th} sub-problem according to the 1/5 successful rule.
Output *BestPath*= $[\mathbf{W}_1^*, \mathbf{W}_2^*, ..., \mathbf{W}_{N_w}^*]$.

C. SEW-NCS-C

SEW-NCS-C works as follows: the SEW first decomposes the original path planning problem into a sequence of N_w sub-problems[1]. For the i^{th} sub-problem, N_p waypoints are first randomly initialized in the rotated Cartesian coordinate system [3], [14]. The external restriction associated with the adopted coordinate system is also imposed to bound the sub-problems. Then for each iteration of SEW-NCS-C, each sub-problem is evolved by an NCS-C. Specifically, for the i^{th} sub-problem, the j^{th} RLS generates a new waypoint $\mathbf{W}_i^{j'}$ based on the current waypoint $\mathbf{W}_i^j$ according to Eq.(24). The fitness function

[1]Strictly speaking, the first and last sub-problems do not need to be evolved as they correspond to the fixed start and destination.

values $f(\mathbf{W}_i^j)$, $f(\mathbf{W}_i^{j'})$ are evaluated in terms of the decomposed objective functions and constraints. The correlations between $\mathbf{W}_i^j$, $\mathbf{W}_i^{j'}$ and other RLSs are calculated according to Eq.(22) and Eq.(26). After that, the better waypoints for the pairwise competitions are remained for the next iteration according to Eq.(23) and Eq.(27). To update the path for output, the best waypoint among the N_p remained waypoints is chosen. If the chosen waypoint is better than the i^{th} waypoint $\mathbf{W}_i^*$ of the path, just replace $\mathbf{W}_i^*$ with the chosen one. As the NCS explicitly prefers solutions that are more likely to produce diverse good solutions. The N_p RLSs will gradually locate the good waypoints that are diverse to each other. In this case, NCS-C always provides multiple diverse high quality options for updating the i^{th} waypoint of the path. Therefore, the path is less likely to be premature. After every $epoch$ iterations, the search step-size for each RLS in each sub-problem is updated based on Eq.(25). After SEW-NCS-C terminates, the path being iteratively updated will be output as the final solution. To summarize, the pseudo-code of SEW-NCS-C is given in Algorithm 1[2].

Lastly, it is worth noting that the fitness f in Eq.(23) is required to be a scalar. However, in our work, a waypoint $\mathbf{W}_i^j$ after evaluation will receive a vector of function values, i.e., 4 objectives values and 4 constraints penalties. To integrate the 8 evaluated values into one scalar, we simply use the weighted sum method here. In [3], the 8 criteria are assigned with 3-levels of priorities in terms of the human preferences, as shown in Table I. Naturally, we can set 3-levels of weights, denoted as $[w_1, w_2, w_3]$, for the 8 criteria. Specifically, the weight for the 4 constraints that must be satisfied, i.e., w_1, will be set to the largest, while for the third-level objectives, the weight w_3 will be set to the smallest. Consequently, the fitness value $f(\mathbf{W}_i^j)$ of a waypoint $\mathbf{W}_i^j$ can be calculated as follow:

$$f = w_1 \sum_{i=1}^{4} C_i + w_2 \sum_{i=1}^{2} O_i + w_3 \sum_{i=3}^{4} O_i \tag{28}$$

where C_i and $O_i, i = 1, 2, 3, 4$, are calculated as follows:

$$C_i = \begin{cases} 0 & \textbf{if} \quad c_i \leq PC_i \\ c_i & \textbf{otherwise} \end{cases} \tag{29}$$

$$O_i = \begin{cases} 0 & \textbf{if} \quad o_i \leq PO_i \\ o_i & \textbf{otherwise} \end{cases} \tag{30}$$

where c_i and $o_i, i = 1, 2, 3, 4$, are the constraints penalties and objective values of the current waypoint calculated in terms of the decomposed functions Eqs.(13)-(20). PC_i and $PO_i, i = 1, 2, 3, 4$ are the preferences for constraints and objectives, respectively.

V. Simulation Studies

To verify that MMOAs can evolve the sub-problems more effectively than GOAs, we conduct a set of simulated comparisons between SEW-NCS-C and the planner in [3], denoted as SEW-JADE. As SEW-JADE has shown significant superiorities to other existing EA-based planners, they will not be considered in the comparisons. In this section, the tested scenarios are first described. After that, the parameter settings and performance measures used are briefly introduced. At last, the simulation results are analysed.

A. Scenarios

The scenarios tested in this work are generated based on the ones used in [3]. Specifically, the terrain is represented as the landscape of a variant of the Foxhole Shekel optimization problem. formulated as Eq.(31):

$$h(\mathbf{x}) = \sum_{i=1}^{30} \frac{0.1}{\sum_{j=1}^{2}(x_j - a_{ij})^2 + b_i} \tag{31}$$

where parameters **a** and **b** can be modified to adjust the landscape. The mission space is limited within the space of $[0, 10] \times [0, 10] \times [0, 0.5]$. The obstacles are depicted as the range of hostile missiles. The number of obstacles varies by randomly setting the missiles on the ground in the range of $[1, 9] \times [0, 10]$. Specifically, we construct 3 scenarios in this simulation, where

[2]The matlab code is available on: http://home.ustc.edu.cn/~trevor/code_files/SEW-NCS-C.zip

TABLE I: Human Priorities and Preferences

Constraints				
Name	Minimal Turning Angle	Limited Slope	Limited Terrain	Limited Map
Denotation	c_1	c_2	c_3	c_4
Priority Level	1^{st}	1^{st}	1^{st}	1^{st}
Preference	0	0	0	0
Objectives				
Name	Minimal Path Length Ratio	Minimal Risk of Kill	Minimal Risk of Radar Detection	Minimal Flight Altitude
Denotation	o_1	o_2	o_3	o_4
Priority Level	2^{nd}	2^{nd}	3^{rd}	3^{rd}
Preference	1.5	0	30	0.5

the numbers of obstacles are set to 60, 120 and 180, respectively. For each missile, a coupled radar is set concentric with the missile. The radius of the range of missiles and radar are set to 0.33 and 0.67, respectively. Thus, given the minimal value of $h(\mathbf{x})$ in Eq.(31) is around 0.25 and the maximal flyable height is 0.5, the UAV cannot fly above the missiles and radars, but just passby them from the flank. The start and destination are set at $(0.5, 5, h([0.5, 5]))$ and $(9.5, 5, h([9.5, 5]))$, respectively. The human preferences of the objective functions and constraints are set the same as the ones used in [3], which are shown in Table I.

B. Simulation Protocol

The simulated comparisons only involve two planners, i.e., SEW-NCS-C and SEW-JADE. Since the SEW is parameterless, only the parameters for the NCS-C and JADE need to be set. There is only one parameter for JADE, i.e., the population size N_p, and it is set to 10, which is suggested in [3]. For NCS-C, there are two parameters to be pre-defined. The population size is suggested as 10 in [8], while the parameter r in Eq.(25) is set to 0.8. Actually, the r in [8] was set to 0.99. Notice that, the T_{max} in this simulation is set to 100, the same to [3], while the T_{max} in [8] was 30000. We hence need to decrease the r to shrink the search step-size of RLSs in SEW-NCS-C faster to refine the solutions. Lastly, the weights for aggregating the 8 criteria are set to $w_1 = 1000, w_2 = 100$ and $w_3 = 0.1$ to address the differences among 3-levels priorities.

All the simulations are conducted on the Matlab 2014b software on a win-10 PC with i3-2130 CPU @ 3.40 GHz CPU and 4 GB RAM. To measure the results, the Statistical Front-Dominance Ranking Procedure (SFDRP) metric [3] is adopted to see whether the qualities of the output paths of SEW-NCS-C are statistically significantly better than those of SEW-JADE. Specifically, for two planners A and B, SFDRP first counts the numbers of the path of l^{th} run output by A is dominated by the paths output by planner B in 25 runs in terms of objectives and constraints in Section II, denoted as $(\diamondsuit_{A_l}^{B_{1:25}})$ and vice versa $(\diamondsuit_{B_l}^{A_{1:25}})$. Then, the non-parameteric Wilcoxon rank sum test is applied to the vectors $[\diamondsuit_{A_1}^{B_{1:25}} + 1, \diamondsuit_{A_2}^{B_{1:25}} + 1, ..., \diamondsuit_{A_{25}}^{B_{1:25}} + 1]$ and $[\diamondsuit_{B_1}^{A_{1:25}} + 1, \diamondsuit_{B_2}^{A_{1:25}} + 1, ..., \diamondsuit_{B_{25}}^{A_{1:25}} + 1]$ at 0.05 significance level. If this test finds a statistically significant difference, the median of each vector is used to infer which planner dominates the other one. Besides that, we also calculate the average convergence speed, average elapse time and the successful rate of each planner over 25 runs on each scenario. Briefly, the convergence speed of each run is defined as the iteration that the path has satisfied all constraints and optimized all objectives to below the preferences, denoted as CS. The average elapse time over 25 runs is denoted as $\widetilde{ET}$. The success rate, denoted as SR, counts the number of successful runs out of the total 25 runs. A successful run is defined as that the CS of that run is smaller than T_{max}. Then the average convergence speed over 25 runs is defined as the average of CS over those successful runs, denoted as $\widetilde{CS}$. The results are shown in Table II.

C. Results and Analyses

A general conclusion can be drawn from Table II that SEW-NCS-C outperforms SEW-JADE on all 3 scenarios. Specifically, all 3 SFDRP tests show that the qualities of paths obtained by SEW-NCS-C are statistically significantly better than those of SEW-JADE. The success rates of SEW-NCS-C become increasingly larger than those of SEW-JADE as the obstacles increase. Although the convergence speed of SEW-NCS-C is slightly slower than that of SEW-JADE due to the higher diversity maintained, it is quite acceptable (no more than half the time budget of one run). The runtime of both planners keeps almost the same as they use the same population size.

Besides that, we also illustrate the behaviors of the paths obtained by both planners on the 180-obstacles scenario, where the infeasible areas are largest among 3 scenarios. To be fair, the median path (in terms of SFDRP tests) out of 25 runs obtained

TABLE II: The convergence speed, runtime, success rate and median of SFDRP of 2 planners on 3 scenarios

60-Obstacles				
Planner	$\widetilde{CS}$	$\widetilde{ET}(s)$	$SR(\%)$	SFDRP
SEW-JADE	14.71	89.40	84	4
SEW-NCS-C	21.68	89.79	88	5
120-Obstacles				
Planner	$\widetilde{CS}$	$\widetilde{ET}(s)$	$SR(\%)$	SFDRP
SEW-JADE	20.79	122.98	56	7
SEW-NCS-C	29.50	121.41	80	12
180-Obstacles				
Planner	$\widetilde{CS}$	$\widetilde{ET}(s)$	$SR(\%)$	SFDRP
SEW-JADE	15.00	190.57	4	7
SEW-NCS-C	44.37	187.95	76	25

by SEW-JADE and SEW-NCS-C on the 180-obstacles scenario are depicted in Fig. 1. In this figure, the ranges of missiles, i.e., obstacles, are represented as red filled circles and the paths are shown in black and blue curves. These two paths have very similar "heads" and "tails" outside the box area bounded by the two dash line. Between the two dash lines, it can be clearly seen that the path obtained by SEW-JADE is infeasible as it goes into the red circles. It can be inferred that the path obtained by SEW-JADE was "moving" to where the path obtained by SEW-NCS-C is. However, it stagnated due to some candidate waypoints got converged in the infeasible areas. Comparatively, the path obtained by SEW-NCS-C is feasible. This is due to the negatively correlated search behavior of NCS, which is helpful to resist the convergence and avoid premature.

Consequently, the simulation studies verify the proposed viewpoint that MMOAs can solve the sub-problems more effectively than GOAs.

VI. Conclusions and Discussions

This work investigated how to improve the performance of SEW against large number of obstacles. We discussed how the increasing number of obstacles would deteriorate the GOA and proposed that MMOAs could solve the sub-problems more effectively. To be specific, as the number of obstacles increases, GOAs will easily get prematurely converged or stagnated due to the lack of useful fitness information to guide the evolution. Contrarily, MMOAs try to search different areas of the solution space simultaneously for multiple diverse good solutions, by explicitly maintaining the diversity among the population. In this case, the explicit force of diversity maintenance is beneficial to avoid premature convergence or stagnation of the search.

To verify the proposed viewpoint, we employ a recently proposed MMOA, named Negatively Correlated Search (NCS), under SEW. A set of simulated comparisons have been performed between the new planner, i.e., SEW-NCS-C, and the original planner with a GOA, i.e., SEW-JADE. The simulation results have shown that SEW-NCS-C outperforms SEW-JADE on 3 scenarios with 60, 120 and 180 obstacles, respectively. An illustration on the 180-obstacles scenario has also been given for an intuitive study.

For future work, it would be interesting to extend the proposed idea to the constrained global optimization problems. Generally, as the infeasible areas (induced by the constraints) increases, there will be less fitness information provided to guide the evolution. On this basis, the GOAs may easily get prematurely converged or stagnated due to the lack of evolution motivation. On the contrary, if MMOAs are employed to evolve the sub-problems, the population may have higher exploration ability in each sub-problem to against premature, due to the external motivation of diversity maintenance.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61329302), the Program for New Century Excellent Talents in University (Grant No. NCET-12-0512) and EPSRC (Grant No. EP/K001523/1) and the Natural Science Key Research Project for Higher Education Institutions of Anhui Province (KJ2016A438). Xin Yao was also supported by a Royal Society Wolfson Research Merit Award.

References

[1] E. Besada-Portas, L. de la Torre, J. M. de la Cruz, and B. de Andrés-Toro, "Evolutionary trajectory planner for multiple uavs in realistic scenarios," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 619–634, 2010.

[2] C. Zheng, L. Li, F. Xu, F. Sun, and M. Ding, "Evolutionary route planner for unmanned air vehicles," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 609–620, 2005.

[3] P. Yang, K. Tang, J. A. Lozano, and X. Cao, "Path planning for single unmanned aerial vehicle by separately evolving waypoints," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1130–1146, 2015.

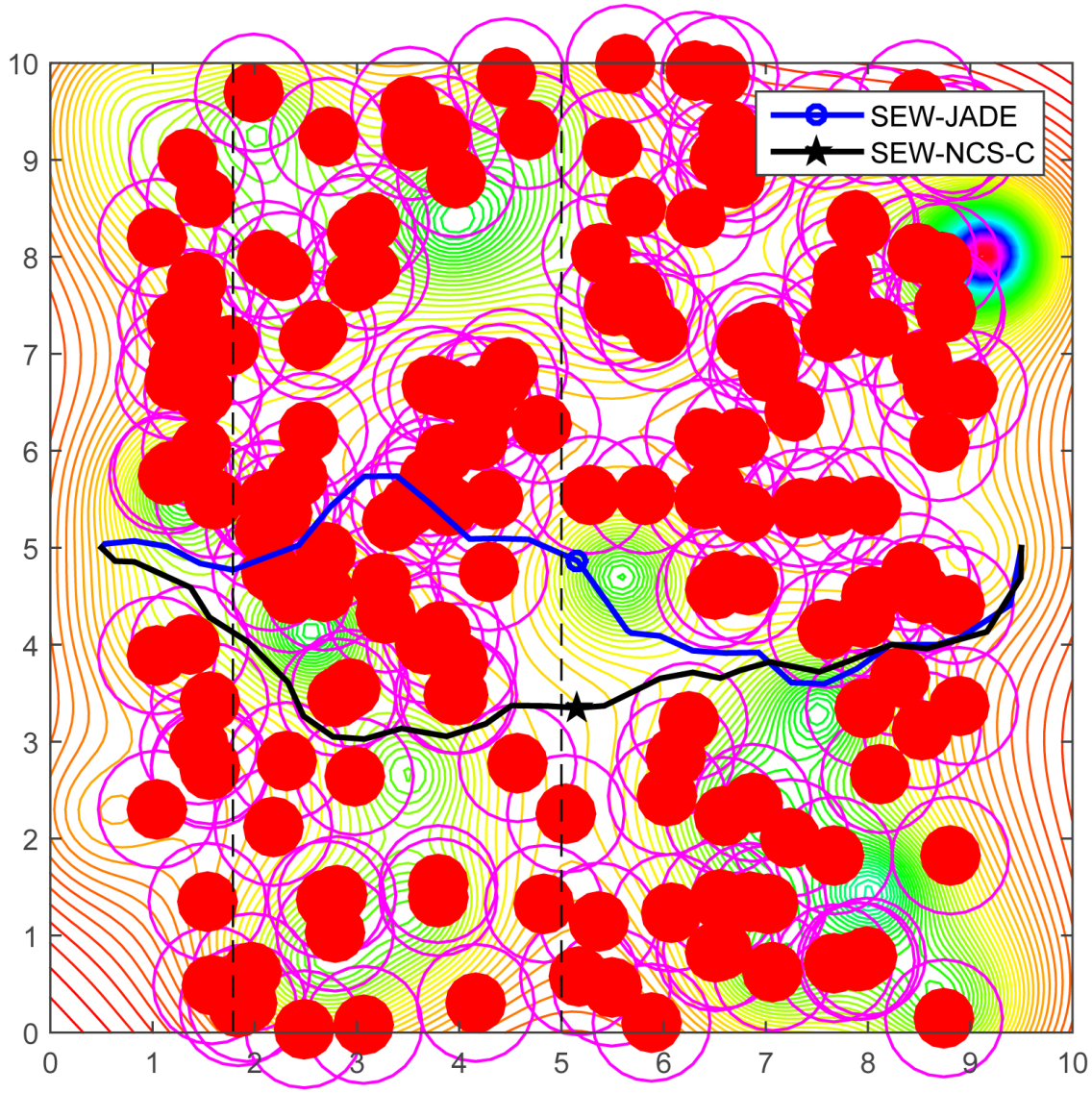

Fig. 1: The median path (in terms of SFDRP tests) out of 25 runs obtained by SEW-JADE and SEW-NCS-C on the 180-obstacle scenario.

[4] J. Zhang and A. C. Sanderson, "Jade: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[5] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: a survey," *ACM Computing Surveys*, vol. 45, no. 3, p. 35, 2013.

[6] S. Das, S. Maity, B.-Y. Qu, and P. N. Suganthan, "Real-parameter evolutionary multimodal optimizationa survey of the state-of-the-art," *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 71–88, 2011.

[7] P. Darwen and X. Yao, "Every niching method has its niche: Fitness sharing and implicit sharing compared," in *Parallel Problem Solving from NaturePPSN IV*. Springer, 1996, pp. 398–407.

[8] K. Tang, P. Yang, and X. Yao, "Negatively correlated search," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 542–550, March 2016.

[9] L. Li and K. Tang, "History-based topological speciation for multimodal optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 1, pp. 136–150, 2015.

[10] P. Yang, K. Tang, and X. Lu, "Improving estimation of distribution algorithm on multimodal problems by detecting promising areas," *IEEE Transactions on Cybernetics*, vol. 45, no. 8, pp. 1438–1449, Aug 2015.

[11] Y. Liu and X. Yao, "Ensemble learning via negative correlation," *Neural Networks*, vol. 12, no. 10, pp. 1399–1404, 1999.

[12] T. Kailath, "The divergence and bhattacharyya distance measures in signal selection," *IEEE Transactions on Communication Technology*, vol. 15, no. 1, pp. 52–60, February 1967.

[13] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies–a comprehensive introduction," *Natural computing*, vol. 1, no. 1, pp. 3–52, 2002.

[14] P. Yang, K. Tang, and J. Lozano, "Estimation of distribution algorithms based unmanned aerial vehicle path planner using a new coordinate system," in *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2014, pp. 1469–1476.