Path Planning for Unmanned Aerial Vehicles Based on Genetic Programming

YANG Xiaoyu[1,2], CAI Meng[3], LI Jianxun[1,2]

1. School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, 200240

2. Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai, 200240
E-mail: kimimccool@126.com, lijx@sjtu.edu.cn

3. Luoyang Institute of Electro-optical Equipment, AVIC, Luoyang 471009, China
E-mail: caimengluoyang@163.com

Abstract: Path planning system is one of the key component for the unmanned aerial vehicles (UAVs) and mobile robots in modern operational systems used in all sorts of circumstances. Generally, genetic algorithm (GA) plays a big role in dealing with optimization problems. However, compared to GA, genetic programming (GP) displays better modeling and optimizing ability in path planning problem. GP is capable of dealing with UAV and mobile robot path planning problems. GP improves performance by utilizing generalized hierarchical computer programs and optimizing evolutionarily. This paper presents an optimized GP method which applies to path planning problem. Several special designed function and symbol operators are proposed and appended to the binary tree structure, as well as the redesigned decoding system. With the combination of selection and reproduction operation, the optimized GP accomplishes the design of path planning. By using the optimized GP method, experiment results display better fitness paths against GA method.

Key Words: Path planning, Unmanned aerial vehicles, Genetic programming, Genetic algorithm, Optimized GP, Function and symbol operators

1 INTRODUCTION

With the continuous improvement of artificial intelligence and computer technology, the unmanned aerial vehicle (UAV) path planning has become the major research matter of task planning system[1][2]. Path planning system is mainly used in UAV and robot problems. Path planning for UAVs is a complex multiple-objective optimization issue considering the precise maneuvers under enemy threats and terrain obstacles which are hugely important[3].

Genetic algorithm (GA) is a technique of complex problem- solving and has a greater search space. In evolutionary calculation, GA has greater role. As an interesting domain of computer science, GA method is very robust in nature and is capable of optimizing complex results, which includes a great number of interacting components[4]. Generally speaking, GA enhances the performance by developing a population of individuals and using evolutionary operations. GA uses chromosome and its genes to denote the operators and evolves by mutation and crossover. Plenty of work have already completed by using GA to create a good route in the path planning problem[5]. However, the traditional GA has some weaknesses when facing the hierarchical problems as hierarchical structure is unpredictable. It becomes specifically serious when facing fitting problems. Moreover, many computer programs and mathematical problems cannot be presented by GA, especially the inequality constraints later in the article. Furthermore, because of the lack of dynamic, GA can merely describe fixed-length character, while in many cases dynamic length is required.

After a very careful but still partially incomplete search in IEEE Xplore Digital Library and CNKI Library, we come to the conclusion that genetic programming (GP) is a brand new method in UAV path planning problem. GP has the capability of dealing with UAV path planning and mobile robot planning problems, which are the two main domain of path planning research. In mobile robot problems, GP is applied to avoid obstacles, but robot planning problem has massive difference with UAV path planning problem. Hence, aiming at the weakness of GA, GP is proposed as a solution. GP is a typical form of the evolutionary algorithm (EA) that evolves computer programs or program-like executable structures[6]. GP can present the hierarchical formula and computer programs in a dynamic way, which brings wider application value. GP path planning strategy is an effective population-based tree structure meta-heuristic optimization method. The GP strategy has strong convergence and robustness and is convenient in designing operators in order to optimize paths. The advantage of GP strategy over GA lies in its convenience and ability of avoiding the navigating point used in GA, which brings better results. Particularly, an optimized GP is proposed to improve performance. Several function operators, symbol operators as well as genetic operators are created to accomplish it. Specially, the decoding system is redesigned for GP. Some elaborated function and symbol operators are

This work was jointly supported by National Natural Science Foundation(61175008); Project(6133190302); Shanghai Aerospace Science and Technology Innovation Fund(SAST201448); Aerospace Science and Technology Innovation Fund and Aeronautical Science Foundation of China(20140157001); 2015 Industry-university-research cooperation project of AVIC;

designed to control every step towards the objective. By copying and mutating, the algorithm breeds and creates changes in evolutionary way. The simulation results show that the optimized GP strategy is more effective than the traditional GA method by generating an optimized path with better fitness.

This paper defines the multi-objective path optimization problem in mathematical way and optimizes the GP method by creating plenty of special operators. Based on GP, we present the exact solution to solve the path planning problem by using evolutionary strategies. Experiments using both GP and GA are accomplished and results are exhibited.

2 PROBLEM DEFINITION

2.1 Multi-objective optimization formulation

Multi-objective optimization model can convert to plenty of real life problems. Generally, multi-objective optimization problem is defined as below[7]:

$$\min V_i = f_i(x), i = 1, 2, \cdots, N \tag{1}$$

$$s.t \begin{cases} X = [x_1, x_2, \cdots, x_d], X \in R^d \\ g_j(X) \le 0, j = 1, 2, \cdots, J \\ h_k(X) = 0, k = 1, 2, \cdots, K \end{cases} \tag{2}$$

Where $f_i(x)$ represent the objective functions. X is the d-dimensional decision variable vector. In the constraint functions, $g_j(X)$ represent the inequality constraints, while $h_k(X)$ denote the equality constraints.

2.2 Paths optimization formulation

Path planning is a complex multi-objective optimized problem, the function is defined below:

$$\min f(x) = (f_1(x), f_2(x), f_3(x))^T \tag{3}$$

$$f_1(x) = \sum_{i=1}^{n} l_i^2, f_2(x) = \sum_{i=1}^{n} h_i^2, f_3(x) = \sum_{i=1}^{n} f_{TA_i} \tag{4}$$

Where $f_1(x) = \sum_{i=1}^{n} l_i^2$ represents the cost function of path length, while l_i denotes the path segment length. The minimization of the function leads to a smoother and shorter path. $f_2(x) = \sum_{i=1}^{n} h_i^2$ represents the cost function of the flight altitude, while h_i denotes the altitude of UAV, as minimum altitude should be ensured. $f_3(x) = \sum_{i=1}^{n} f_{TA_i}$ represents the evaluation value of all threats, while f_{TA_i} is calculated below[8]:

$$f_{TA_i}(x) = \begin{cases} \beta_j K_j / (R_j)^4, x \in Threat.j \\ 0, x \notin Threat.j \end{cases} \tag{5}$$

This function defines the threat evaluation of position x impacted by threat j. Inside the function, β_j, K_j are threat coefficients, R_j is the distance between x with j. $x \in Threat.j$ indicates that x is affected by threat j.

The constraints of the problem are shown below:

$$\begin{cases} l_{\max} - l_i \ge 0, i \in I \\ l_i - l_{\min} \ge 0, i \in I \\ \dfrac{a_i^T a_{i+1}}{\|a_i\| \cdot \|a_{i+1}\|} - \cos\psi \ge 0, i \in I \\ \dfrac{|z_i - z_{i-1}|}{|a_i|} - \tan\theta \le 0, i \in I \\ H_i - H_{\min} \ge 0, i \in I \end{cases} \tag{6}$$

Where the first two constraints denote the maximum path length $l_{\max}$ and minimum path length $l_{\min}$. The third and fourth constraints denote the maximum turning angle ψ and maximum pitch angle θ. The last constraint $H_{\min}$ shows the minimum flight altitude of UAV.

The upper objective functions and constraints constitute the path multi-objective optimization problem.

3 AN OPTIMIZED TREE-BASED GP

The concept of GP is developed by John. R. Koza[9] in 1989, while the tree-based structure of GP is presented by Cramer in 1985. GP has unique advantage in solving multi-objective optimization problem. This paper mainly utilizes GP to accomplish path planning. Meanwhile, as we describe the intrinsic nature and features of GP, plenty of these are improper. Hence, a new optimized GP is proposed in this section.

GP utilizes a set of feasible functions to describe problems. Moreover, these functions can be repeatedly denoted by N_f number of functions, i.e., $F = \{f_1, f_2, \cdots, f_{N_f}\}$, and N_{term} number of terminators, i.e., $T = \{a_1, a_2, \cdots, a_{N_{term}}\}$. Then for the functions $f_1, f_2, \cdots, f_{N_f}$, the independent variable number is supposed to be $z(f_1), z(f_2), \cdots, z(f_{N_f})$. The functions inside can be represented by arithmetic operator, standard mathematical function, Boolean operator, conditional expression and iterative function. Similarly, the terminators can be denoted by the input of system, recognizer, sensor, state variable and constant.

Originally, GP's goal is to discover the computer program that is able to indicate the nature of the problem, i.e., to find the computer program with the best fitness value. Thus, it's very reasonable and suitable to solve fitting problems by using GP. Nevertheless, the modeling problem of path planning is more realistic and traditional operators is incapable of describing the practical states or actions. Hence, we propose the special designed operators that distinct from these traditional operators.

3.1 Function operators

In path planning problems, the prior goal of the objective route is staying away from the scope of threats. The range of threats is defined as a circle with a given radius in two-dimension plane figure. Therefore, to judge whether the next move is in the range of threats, we denote the function operators in the function set as below:

$$F = \begin{Bmatrix} IF-FORWARD-AVAILABLE \\ IF-VERTICAL-AVAILABLE \\ IF-FLAT-AVAILABLE \end{Bmatrix} \quad (7)$$

By presenting these operators, The path planning system determines the direction of each step. First, The $IF-FORWARD-AVAILABLE$ operator calculates the angle between the current position and target position. This angle is the exact forward direction at current position. Then by assuming the UAV takes a step in the forward direction, GP calculates the supposed position that is called P_{next}. Afterwards, for each threat, GP calculates the distance between supposed position P_{next} and threat position, defining it as d. We compare d with the corresponding radius and if all the results are positive, the $IF-FORWARD-AVAILABLE$ operator is positive, i.e., forward direction is available. Inside the binary tree structure, as the branch component, the result of $IF-FORWARD-AVAILABLE$ determines the move direction. Second, $IF-VERTICAL-AVAILABLE$ operator operates by assuming the UAV takes a step in the vertical direction. GP calculates the supposed position P_{next} and compare the distance with every threat radius. Third operation $IF-FLAT-AVAILABLE$ is similar as it operates by assuming the UAV takes a step in the flat direction.

3.2 Symbol operators

The symbol operators can also be called the terminal operators since they represent the actual actions the UAV takes. Since we are heading towards the target point, the forward direction step is necessary and reasonable. We denote $Move-Forward$ operator to the symbol set to represent a step forward. Countering the scope of the threats, we propose two directions of movements: $Move-Vertical, Move-Flat$. These two operators help turn the path direction and avoid crushing into threats' range. Therefore, the symbol operator set is as below:

$$T = \begin{Bmatrix} Move-Forward \\ Move-Vertical \\ Move-Flat \end{Bmatrix} \quad (8)$$

Symbol operators are always the leaves of the binary tree. They are determined by their parent node, which is always a function operator.

The new operators has the advantage of adapting to the exact planning problem and generating the path point by point. We can also use the step to control the accuracy needed in path planning. Contrarily, these advantages are unrealizable by means of traditional operators. Thus, with the combination of function and symbol operators, various paths are created by the diverse structure and contents optimized GP brings in.

4 PATH PLANNING USING OPTIMIZED GP

Path planning is a multi-objective optimization problem. GA is utilized in multiple path planning situations, however, the results show that GA has strong dependency on the threats' positions, which leads to a high fitness value and thus a low efficiency path. GP follows the common mechanisms of the biological evolution: selection, reproduction, mutate and replacement, but specifically uses unique methods to accomplish path planning problem. The particular procedure of GP is described below.

4.1 Basic idea

The path planning problem is based on a specific terrain with plenty of threats. The attributes of threats are position, radius and weight. The UAV travels from the start position to the objective position without being included in the range of threats.

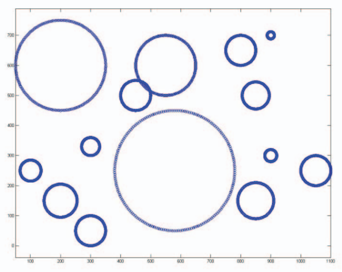

Fig 1. Terrain and threats range

GP is based on the pattern of computer manifestation, which has several distinct branches that demonstrated below. Linear GP utilizes a segment of senior programming language or machine language to make up commands. This segment of commands can be comprehended as a segment of string list. Graph-based GP uses graphs to store a set of programming operators. The storage method is manifold as both adjacent matrix and adjacent table can be adopted. Tree-based GP utilizes tree structure to store operators. The strength of tree structure is its simplicity and similarity with syntax tree. By maintaining the root node and its relationship with the child nodes, this structure can be easily expressed and understood. We choose binary tree as the GP structure due to its simplicity and efficiency.

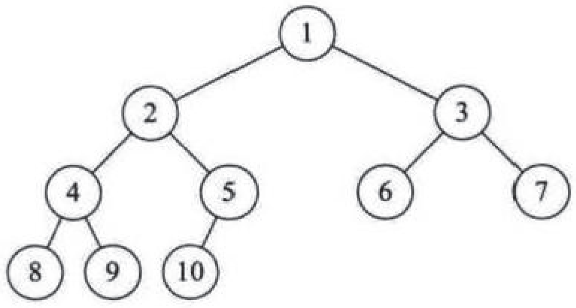

Fig 2. Binary tree

Hence, the main idea is using the binary tree structure to determine every exact step. Primarily GP initializes the basic tree that used in every step of the procedure. The elements of the tree are symbols and functions, i.e., symbols represent directions and functions denote conditions. Then the decoding section converts the tree into the direction of next step. The decoding section will keep calculating until the plane reaches the objective position. GP records every position of the path and calculates fitness value as the evaluation.

Every entity tree is decoded into a path to the objective. Selecting process is based on the tournament selection. In accordance with the fitness value, the elite entities will be selected and copied into next generation. Similar to the elite entities, some survivals are randomly chosen among the remaining entities. Moreover, GP imitates the evolutional heteromorphosis by adding crossovers and mutations to some entities. GP ensures the performance will improve by every generation and the final result is the best route.

4.2 Functions and symbols

Three functions are proposed in formula (7) in subsection 3.1 and three symbol operators are denoted in formula (8) in subsection 3.2, as described earlier.

4.3 Initialize population

Initial population consists of multiple entities. Maintaining a diverse population is a significant consideration in GP to keep the features in entities. The traditional methods of initializing population are diversified, Such as complete-grow method and incomplete-grow method[10]. As every branch reaches to the maximum depth, the complete-grow method brings complexity and diverse possibilities to the algorithm, while the shortcoming is the lack of diversity in structure. Contrarily, the branch depth of incomplete-grow method is random, which leads to the diversity of the tree structure and the algorithm, and also throws away the complexity to the population [11].
Aiming at the strength and weakness of the two methods above, we denote the mixture method to neutralize the disadvantages while combining the advantages. By defining a maximum depth, we append the entities evenly to every depth from 2 to maximum. The percentage of entities in each depth are given below:

$$n = 100/(\max imum - 1) \tag{9}$$

Half of the entities are initialized by complete-grow method and the other half by incomplete-grow method.
After setting the generation number, the first generation starts by randomly selecting each branch and leaf of the binary tree.

4.4 Decode and calculate fitness

Using the decoding section, the trees are converted into real path. The decode process iterates from root to leaf nodes. By judging whether the function node is correct, the iteration decides to move to the left child node or the right child node. The process stops until a leaf node occurs, as this node shows the move direction. The section ends when the path reaches end point.
Every path has fitness value as the evaluation. The fitness value is calculated below:

$$C = \sum_{i=1}^{n} (w_1 l_i^2 + w_2 h_i^2 + w_3 f_{TA_i}^2) \tag{10}$$

Where w_1, w_2, w_3 is the weight value.

4.5 Selection and breeding

Elitism in GP means always choosing the best entities during the optimization into next generation. For the traditional GP, plenty of methods can be adopted, such as fitness- proportion selection, greedy selection and differential selection. Among them, greedy selection is fit for large entity number situation, while the other three is more commonly used, especially fitness-proportion selection. This solution defines the possibility of selection as below:

$$P = \frac{f(s_i(t))}{\sum_{j=1}^{N} f(s_j(t))} \tag{11}$$

Here $f(s_i(t))$ denotes the fitness of $s_i(t)$ at generation t. This means the larger fitness value, the higher percentage would be.
However, better strategy can be adopted as those methods maintains uncertainty confronting the elite entity. We propose tournament selection[12][13] as optimization to the traditional GP. Tournament selection randomly samples k individuals with replacement from the current population of size N into a tournament of size k, then selects the one with best fitness as a parent from the tournament into the mating pool[14]. By setting k a little higher and manipulating the method several times, we can promote the best entities into next generation.
After the fitness value calculation, we make a quick sort in ascending order depending on the value. The first few elite entities are automatically copied into next generation as their performances are the best. Moreover, we randomly choose some entities as survivals into next generation. Other entities in next generation are derived from crossover and mutate results. The crossover operation randomly choose two entities: e_1, e_2. Each entity selects a node randomly from the binary tree. n_1, n_2 represent the two nodes and their child nodes, Similarly r_1, r_2 denote the remaining parts of the trees. Then the 4 split trees crossover into two new trees: $n_1 + r_2, n_2 + r_1$. Between the two new trees, we choose the shorter one to avoid reaching the length limit. The new tree is selected into next generation.
The mutate operation consists two operations: mutate and strong mutate. Their difference lies in the level of mutate, i.e., strong mutate tries at least twice while mutate merely tries once. By choosing one entity, we randomly select a node from the entity and replace the node with another one. Mutation brings diversity to the population and propel the evolutionary.
Both crossover and mutate operation have a percentage of arising. Commonly mutate chance is rare while crossover percentage is much bigger.

4.6 Termination

We set a generation limit to terminate the iteration. During the last generation, once again the quick sort ranks the performances and the first entity in the vector is the elite entity we seek for.
These biological mechanisms of evolution are the detailed building blocks for path planning using GP method. We present the pseudocode of GP below:

Table1. Pseudocode of GP

Genetic Programming:	
1	**Initialize:**

```
2   $N = 100, G = 50, P_m = 0.3, P_c = 0.9$
3   $Survivors = 15, Mutates = 50, Crossover = 35$
4    **Generate** $N$ trees randomly, $E = \phi$
5            Save trees in $E$
6      $K = 1$
7   **Begin:**
8     **While** $K < G$
9       Calculate **fitness** in $F[100]$
10      **Elite** and **survivors**: 15
11      **Mutation** with $P_m$
12      **Crossover** with $P_c$
13      **Selection** update $E$
14       $K += 1$
```

5 EXPERIMENTS AND RESULTS

In this section, we verify the feasibility and effectiveness of the proposed GP method which is composed of the new operators. Since GA is a good path planning method, we compare this two methods by utilizing them in different path planning scenarios. Different scenarios are filled with threats with multiple range. We apply three different scenarios and the paths results are shown below:

Scenario 1:

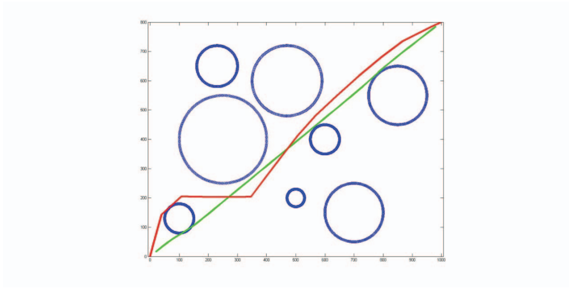

Fig 3. Path comparison 1 (Green: GP, Red: GA)

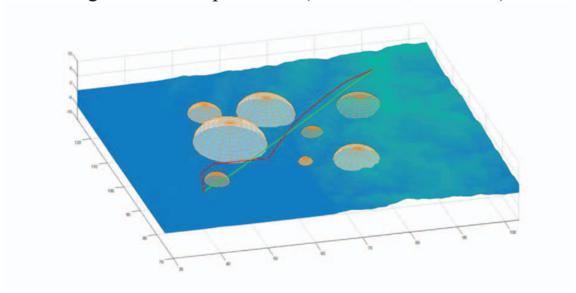

Fig 4. 3D Path comparison 1 (Green: GP, Red: GA)

The GP path fitness is 0.273772, while GA is 0.306801. GP is a lot better in terms of both fitness value and path shape, as path of GP tracks straight to the target. GA path circles around the first threat, which is not needed and makes the path much longer and complicated. In comparison, GP method takes a straight line towards target position and obviously much simpler than GA. Fitness value also proves this analysis.

Scenario 2:

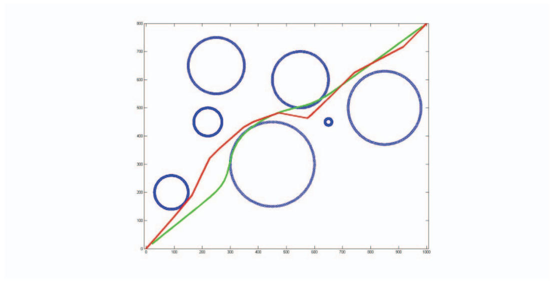

Fig 5. Path comparison 2 (Green: GP, Red: GA)

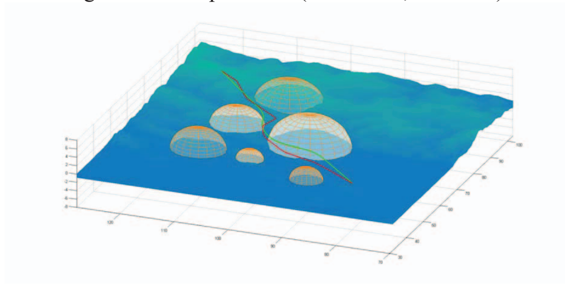

Fig 6. 3D Path comparison 2 (Green: GP, Red: GA)

The GP path fitness is 0.294781, while GA is 0.307859. GP is better in terms of fitness value and the path is also smoother than GA. GA path relies on the supporting nodes nearby the threats' scope, as GA path takes an unnecessary and dangerous movement towards the first threat. GP path takes a smooth turnaround and then heads straight to target point, which is convenient for the real manipulation of aircrafts. Fitness value also proves this analysis.

Scenario 3:

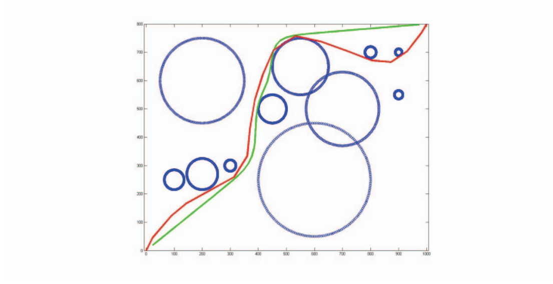

Fig 7. Path comparison 3 (Green: GP, Red: GA)

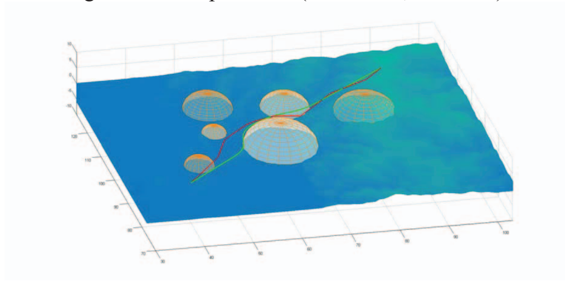

Fig 8. 3D Path comparison 3 (Green: GP, Red: GA)

The GP path fitness is 0.324808, while GA is 0.337479. GP is better in terms of fitness value. This figure shows that GP has little reliance on the threats' position while GA relies heavily on nodes surrounding the threats. Especially, at the end of the paths, GP heads straight towards target position, whereas GA takes an absolutely unnecessary route towards the underneath of the last two threats, leading to a much longer path and more dangerous situation considering the threat distance. Fitness value also proves this analysis.

Scenario 4:

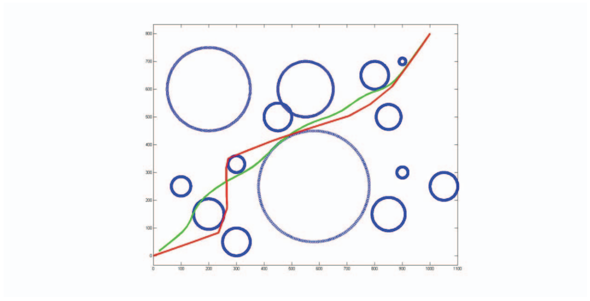

Fig 9. Path comparison 4 (Green: GP, Red: GA)

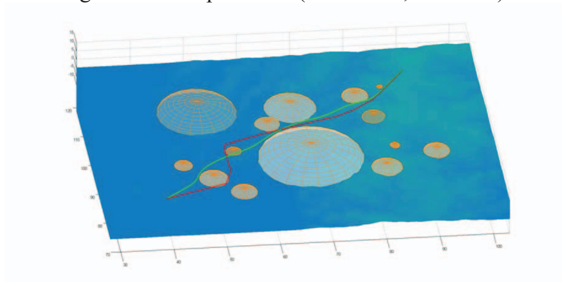

Fig 10. 3D Path comparison 4 (Green: GP, Red: GA)

The GP path fitness is 0.299144, while GA is 0.313265. GP is better in terms of fitness value. This figure shows the difference direction selection between GP and GA. Obviously, the path of GP is much shorter and smoother than GA, especially considering the big turnaround of GA path. GP path swings between the first two threats and can be easily manipulated, whereas GA path takes a bad turn above the second threat instead of beneath it. In this scenario, GP shows better capability of delivering good path. Fitness value also proves this analysis.

Considering all four scenarios, the path created by optimized GP method is apparently better than GA as both figure and fitness value have compared.

6 CONCLUSIONS

As UAV path planning is the major research matter of path planning system, the main contributions of this paper are the optimization of genetic programming and a special -designed set of operators to accomplish the path planning for UAVs by using GP. The planner finds a feasible path effectively under the guide of the operators and quickly converts to a preferable route under the procedure of GP due to the evolutionary operators guided by the proposed optimization formulation.

Extensive simulation tests are operated under different circumstances and in various threat environments. The results show the effectiveness of the proposed strategy. For the future work of path planning using GP, we need to lay emphasis on the path planning on 3D space.

REFERENCES

[1] Murat Cakir, 2D Path Planning of UAVs with Genetic Algorithm in a Constrained Environment, 2015 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO), Istanbul, Turkey, May 27-29, 2015.

[2] Mayank Garg, Abhishek Kumar, P.B. Sujit, Terrain-based landing site selection and path planning for fixed-wing UAVs, 2015 International Conference on Unmanned Aircraft Systems (ICUAS), Denver, Colorado, USA, June 9-12. 2015.

[3] Hao Meng, Guizhou Xin, UAV Route Planning Based on the Genetic Simulated Annealing Algorithm, Proceedings of the 2010 IEEE International Conference of Mechatronics and Automation, August 4-7, 2010, Xi'an, China.

[4] Neeraj, Anil Kumar, Efficient Hierarchical Hybrids Parallel Genetic Algorithm for Shortest Path Routing, Confluence The Next Generation Information Technology Summit, 2014 5th International Conference, 25-26 Sept. 2014, ISBN: 978-1-4799-4237-4.

[5] DE Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company, Inc., Reading Mass, ISBN 0-201-15767-5, 1989.

[6] Angeline P., Kinnear K., Analysis and Implementation Issues in Genetic Programming, MIT Press, 1996, ISBN: 9780262290791.

[7] Karthik Sindhya, Kaisa Miettinen, Kalyanmoy Deb, A Hybrid Framework for Evolutionary Multi-objective Optimization, IEEE Transactions on Evolutionary Computation, vol. 17, no. 4, August 2013, pp. 495-510.

[8] Beard R W, McLain T W, Goodrich M A, et al, Coordinated target assignment and intercept for unmanned air vehicles, IEEE Transactions on Robotics and Automation, 2002, 18(6) :911-922.

[9] J. R. Koza, Genetic programming: on the programming of computers by natural selection, Cambridge, MA: MIT Press, 1992.

[10] S. M. Gustafson, An analysis of diversity in genetic programming, Ph.D. dissertation, School Comput. Sci., Univ. Nottingham, Nottingham, U.K., 2004.

[11] Edmund K. Burke, Steven Gustafson, Graham Kendall, Diversity in genetic programming: an analysis of measures and correlation with fitness, IEEE Transactions on Evolutionary Computation, vol. 8, no. 1, February 2004, pp. 47-62.

[12] T. Stewart, Extrema selection: Accelerated evolution on neutral networks, Proc. IEEE, Congr. Evol. Comput., May 2001, pp. 25–29.

[13] R. Poli, W. B. Langdon, Backward-chaining genetic programming, Proc. Genet. Evol. Comput. Conf., vol. 2. Jun. 2005, pp. 1777–1778.

[14] Huayang Xie, Mengjie Zhang, Parent Selection Pressure Auto-Tuning for Tournament Selection in Genetic Programming, IEEE Transactions on Evolutionary Computation, vol. 17, no. 1, February 2013, pp. 1-19