Automatica 50 (2014) 809–820

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Cooperative and Geometric Learning Algorithm (CGLA) for path planning of UAVs with limited information☆

Baochang Zhang [a,1], Wanquan Liu [b], Zhili Mao [a], Jianzhuang Liu [d], Linlin Shen [c,2]

[a] Science and Technology on Aircraft Control Laboratory, School of Automation Science and Electrical Engineering, BeiHang University, Beijing, 100191, China

[b] Department of Computing, Curtin University, Perth, WA 6102, United States

[c] Shenzhen Key Laboratory of Spatial Smart Sensing and Services, School of Computer Science & Software Engineering, Shenzhen University, China

[d] Media Laboratory, Huawei Technologies Co., Ltd., Shenzhen 518129, China

ARTICLE INFO

Article history:
Received 4 March 2013
Received in revised form
7 December 2013
Accepted 11 December 2013
Available online 15 February 2014

Keywords:
Path planning
UAV
Limited information

ABSTRACT

In this paper, we propose a new learning algorithm, named as the Cooperative and Geometric Learning Algorithm (CGLA), to solve problems of maneuverability, collision avoidance and information sharing in path planning for Unmanned Aerial Vehicles (UAVs). The contributions of CGLA are three folds: (1) CGLA is designed for path planning based on cooperation of multiple UAVs. Technically, CGLA exploits a new defined individual cost matrix, which leads to an efficient path planning algorithm for multiple UAVs. (2) The convergence of the proposed algorithm for calculating the cost matrix is proven theoretically, and the optimal path in terms of path length and risk measure from a starting point to a target point can be calculated in polynomial time. (3) In CGLA, the proposed individual weight matrix can be efficiently calculated and adaptively updated based on the geometric distance and risk information shared among UAVs. Finally, risk evaluation is introduced first time in this paper for UAV navigation and extensive computer simulation results validate the effectiveness and feasibility of CGLA for safe navigation of multiple UAVs.

1. Introduction

The Unmanned Aerial Vehicles (UAVs) have been widely applied in military fields and civil industries, and there are significantly growing research interests in this area over the past decade (Bortoff, 2008; Chandler & Pachter, 2002; Chandler & Rasmussen, 2000; Rabbath, Gagnon, & Lauzon, 2004; Zheng, Li, & Fan, 2005). The maneuverability, collision avoidance and information sharing in path planning for Unmanned Aerial Vehicles (UAVs) are among the most challengeable problems. The current progress has focused on these challenges from single UAV to multiple UAVs (Bellingham, Richards, & How, 2002; Flint & Fernandez, 2005; Kim, Gu, & Postlethwaite, 2006; Nikolos, Zografos, & Brintaki, 2007; Wu, Zheng, & Cai, 2011).

In previous works, Voronoi Graph Search (Bortoff, 2008) and Visibility Graph Search (Bellingham et al., 2002) are among the earliest path planning methods for both single UAV and multiple UAVs, but they have been proven to be effective only in simple environment. Both methods suffer from a fatal failure when the map information is partially available, especially when some obstacles are not detected. The A2D method reported in Kim et al. (2006) can be considered as a rule and inference based approach. When the environment becomes complicated, the method failed to find a good path to escape from dangerous zone. Evolutionary algorithms (Nikolos et al., 2007) have been used as a viable candidate to effectively solve path planning problems and can provide feasible solutions within a short time. Being different with a single UAV, the path planning for multiple UAVs mainly concentrates on the collaborative framework, information sharing (Bauso, Giarrè, & Pesenti, 2004; Beard & Stepanyan, 2003; Dogman, 2003; Flint & Fernandez, 2005; Flint, Polycarpou, & Fernandez-Gaucherand, 2002; Girard, Darbha, Pachter, & Chandler, 2007; Shanmugavel, Tsourdos, & White, 2010). Researchers in Bellingham et al. (2002) use the Mixed-integer Linear Programming to find a global path for multiple UAVs in order to avoid collision. However, the method fails to handle sudden changes in a local region. It is widely known

☆ This work was supported in part by the Natural Science Foundation of China, under Contracts 60903065, 61039003 and 61272052, in part by the Fundamental Research Funds for the Central Universities, and by the Program for New Century Excellent Talents in University of Ministry of Education of China. The material in this paper was partially presented at the ICUAS conference. This paper was recommended for publication in revised form by Editor Berç Rüstem.

E-mail addresses: zhang_baochang@hotmail.com (B. Zhang), w.liu@curtin.edu.au (W. Liu), zmaoaa@ust.hk (Z. Mao), jz.liu@sub.siat.ac.cn (J. Liu), llshen@szu.edu.cn (L. Shen).

[1] Tel.: +86 13621107142; fax: +86 13621107142.

[2] Tel.: +86 13641425339; fax: +86 13641425339.

0005-1098/$ – see front matter © 2014 Published by Elsevier Ltd.
http://dx.doi.org/10.1016/j.automatica.2013.12.035

that Dynamic Programming (DP) can be used to obtain an optimal path when full information and unlimited computation resource are available (Flint & Fernandez, 2005). In Flint et al. (2002), a dynamic programming method is presented to produce the near-optimal trajectories for multiple UAVs to cooperatively search for targets. However, path planning for multiple UAVs is generally a problem depending on individual behaviors, which is not directly suitable for global methods such as dynamic programming. A number of the improved versions of DP have since then been proposed. For example, a so-called stochastic dynamic programming is proposed for UAV path planning in Girard et al. (2007). The method is actually based on random parameters, specifically the number of flying times. The cost function is one of the key issues in DP and its variants. The distance between the center mass and the target is used in Bauso et al. (2004), and the distance and cooperation measures are exploited in Flint and Fernandez (2005) for UAV path planning. In Dogman (2003), the authors propose a path planning algorithm based on a map with probability of threats, which is built from a priori surveillance data. In Bauso et al. (2004) and Beard and Stepanyan (2003), the researchers develop a novel hybrid model, and design consensus protocols for the management of information. They further synthesize local predictive controllers through a distributed, scalable and suboptimal neuro-dynamic programming algorithm. Fuzzy reasoning, or approximate reasoning, has been an active topic in the fuzzy community since the inception of Zadeh's pioneering work (Zadeh, 1974). In Zhao, Zheng, Liu, Cai, and Lin (2011) and Zheng, Wu, Liu, and Cai (2011), fuzzy reasoning methods treat fuzzy reasoning as a process of optimization rather than logical inference in the UAV path planning. An explicit feedback mechanism and the 'virtual obstacle' method, the so-called feedback based CRI (FBCRI), are embedded into the optimal fuzzy reasoning method to solve the path planning for multiple UAVs (Zhao et al., 2011; Zheng et al., 2011). In FBCRI, the fuzzy rule base is updated during the reasoning process by incorporating newly generated rules into the original rule base (Zheng et al., 2011). Furthermore, by embedding some virtual sub-goals into the FBCRI based approach, a new cooperative path planning approach based on virtual sub-goals (CPVS) is proposed in Zhao et al. (2011). But the analytic robustness analysis of various fuzzy reasoning methods is often very complicated (Cai & Zhang, 2008; Zheng et al., 2011). Though there has been significant progress on path planning for multiple UAVs based on a given risk map and logical feedback controller, to the best of our knowledge, the path planning for multiple UAVs based on adaptive information sharing is not well studied in the literature.

In this paper, we address the path planning problem for both single and multiple UAVs in a dynamically changing environment from perspectives of information sharing and reinforcement learning (Even-Dar & Mansour, 2003; Watkins, 1989; Zhang, Mao, Liu, Liu, & Zheng, 2013). In path planning, Q-Learning (Watkins, 1989) (reinforcement learning) is of high performance for the case when the entire map is known to planners (Mao, Zhang, & Xu, 2012; Zhang, Mao, Liu, Liu, & Zheng, 2013). Geometric distance and information sharing are believed to be very important elements for path planning when partial area of the map is changing (Zhang, Mao, Liu, & Liu, 2013) based on a shared weight matrix and the reward matrix by all UAVs in a random process. However, such a valuable information is not used by the traditional reinforcement approach. In the theoretical aspect, we know that the convergence of Q-Learning is only studied in a probabilistic manner (Even-Dar & Mansour, 2003). Motivated by the above observation, we propose a new algorithm, called the Cooperative and Geometric Learning Algorithm (CGLA), to build a general path-planning model based on the criterion in terms of geometric distance and integral risk information. The theoretical convergence and complexity analysis about the proposed method are also well investigated in this paper.

The concepts of the Individual Weight Matrix and cost matrix are proposed in this paper and they lead to an efficient path planning of both single UAV and multiple UAVs. Both matrices can be dynamically updated by taking both the distance information and integral risk into consideration. A single weight matrix or cost matrix is used by all UAVs (Zhang, Mao, Liu, & Liu, 2013), which will result into bad path planning results when no communication is available among UAVs, even can be of higher efficiency. This paper exploits an individual weight matrix and the cost matrix for a more reasonable path planning. In multiple UAVs, the "virtual obstacle" is proposed and embedded into the weight matrix calculation for prevention of possible collisions. Extensive computer simulation results show that the proposed approach performs very well in terms of the path length and the integral risk measure, which was validated to be a good criterion for path planning.

The rest of the paper is organized as follows: Section 2 describes the UAV threat environment modeling. Sections 3 and 4 present the main components of the CGLA algorithm. The extensive computer simulation results are given in Section 5, and Section 6 concludes the paper.

2. Modeling of UAV environment

UAVs often fly in low-attitude urban sky, or mountainous environment, which needs to be simulated to evaluate performances of different path planning methods. UAVs are vulnerable to attack from the ground, or other UAVs. It is necessary for UAVs to keep a certain distance from regions of high risk to ensure a safe flying. The distribution of the probabilistic risk to an obstacle on a map can be used to compute the risk at any location on the map. For example, the probabilistic risk of the area where UAVs could not fly over can be represented as a very large value. The measure of probabilistic risk exposure to a given obstacle can be seen as a continuous distribution function (Kim et al., 2006; Zhang, Mao, Liu, Liu, & Zheng, 2013) as shown in Fig. 1. Given an obstacle at position (x_i, y_i), the risk in a two-dimensional space f_d can be represented as $f_i(x, y)$ using a normal distribution:

$$f_i(x, y) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{\left(-\frac{d_i^2}{2\sigma_i}\right)}, \quad d_i = \sqrt{(x - x_i)^2 + (y - y_i)^2}. \tag{1}$$

When more than one obstacle exists on a map, the probabilistic risk at position (x, y) can be calculated as:

$$F(x, y) = 1 - \prod_{i=1}^{M} [1 - f_i(x, y)]. \tag{2}$$

3. The Individual Weight Matrix

The relationship between action and state (Watkins, 1989; Zhang, Mao, Liu, Liu, & Zheng, 2013) is a very important issue for the path planning problem. Firstly, in order to find the next point from the current point in a given map, we need to know the relationship or weight between any two positions or points. Secondly, further action for each step in path planning needs to be considered in terms of both the distance and the risk information for two neighborhood points in a given map. In the following, we will first discuss the state update method in the traditional Q-Learning method and then analyze its drawbacks.

3.1. Analysis of Q-Learning algorithm

Q-Learning is a model-free reinforcement learning technique (Watkins, 1989). It learns an action-value function that gives the expected utility of taking a given action in a given state. Q-Learning is able to compare the expected utility of all available actions without requiring a model of the environment. The problem formulation consists of an agent, a set of states S and a set of actions

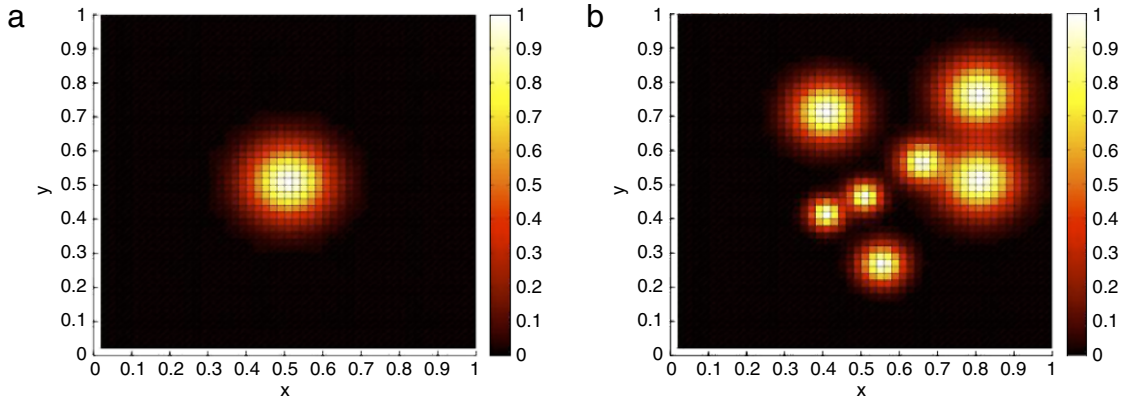


Fig. 1. (a) The probabilistic risk of exposure to an obstacle as normal distribution, (b) is the probabilistic risk of exposure to multiple obstacles.

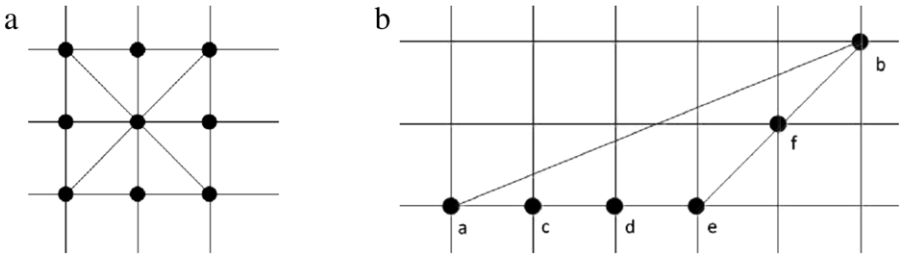


Fig. 2. Illustration of Geometric distance. (a) Eight directions in Q-Learning. (b) An example of the Geometric distance.

A associated with each state. By performing an action $a \in A$, the agent can move from one state to another and be rewarded (a real or natural number) by such a change of state. The goal of the agent is to maximize its total reward or cost by learning the optimal action for each state.

The algorithm therefore defines a function to calculate the quality of a state-action combination:

$$Q : S \times A \to R. \tag{3}$$

Before the learning process get started, Q returns an initial value chosen by the designer. Then, each time the agent is given a reward, i.e., the state has changed, new values are calculated for a combination of a state s from S, and action a from A. The core issue of the algorithm is that the new quality value is obtained by an incremental correction based on the new information as follows:

$$Q(s_{t+1}, a_{t+1}) \leftarrow Q(s_t, a_t)(1 - \alpha_t(s_t, a_t)) + \alpha_t(s_t, a_t)\left[R_{t+1} + \gamma * \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})\right] \tag{4}$$

where R_{t+1} is the reward observed after performing a_t in s_t, $\alpha_t(s_t, a_t)$ $(0 < \alpha \leq 1)$ is the learning rate. The discount factor γ is between $0 \leq \gamma < 1$, which determines the importance of future reward.

We now review the idea and shortcomings of the Q-Learning approach. The decision in Q-Learning depends only on the current state and the weight of the next states. As the next state is confined to one of the eight states of its neighborhood, the flight direction of a UAV is limited to the eight directions shown in Fig. 2(a), which substantially affects the feasibility of the UAV flight path and the algorithm complexity of finding the trajectory of UAVs (Zhang, Mao, Liu, Liu, & Zheng, 2013). As shown in Fig. 2(b), the geometric distance proposed in this paper can result in a more efficient and direct path from position a to b, while the corresponding one in Q-Learning has to pass c, d, e, f. Moreover, it is logically reasonable that the weight matrix should contain both geometric distance and object risk information in order to obtain the next state transition. In Q-Learning, the weight at a target point has to be very large to make the final path reasonable. In this paper, we calculate the integral distance between every two points on the map, and then design a cost matrix and individual weight matrix update mechanism to adapt the local environment changes.

3.2. Basic idea of the Individual Weight Matrix

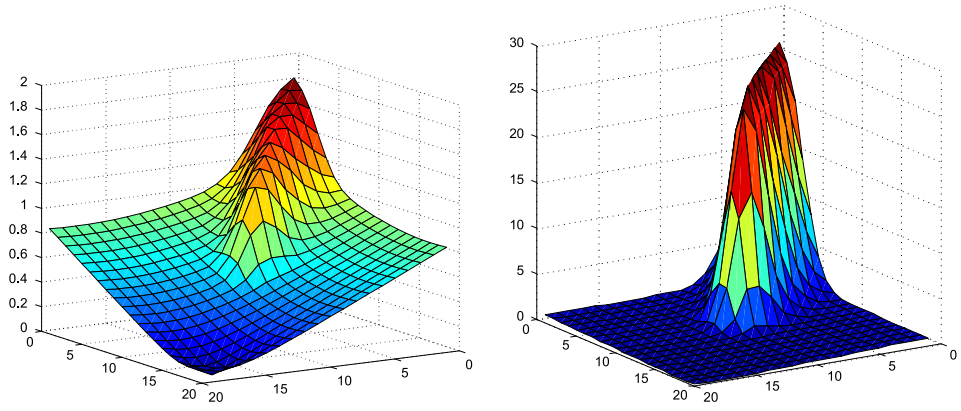
In this paper, we introduce a concept, the weight matrix, to describe the relationship between different points in the map. Different from Eq. (3) that describes the relationship between the state and action, the weight matrix is designed to measure any two points on the map in terms of risk and geometric distance. In Q-Learning, the action in each step is restricted to a set of actions determined by the state matrix. Generally the state and action relationship as in Eq. (3) cannot be too complicated, and the further action in each step is generally restricted to the eight directions for path planning of UAV.

In this paper, the map is expressed into $N \times N$ lattice (i.e. $N = 20$), and the weight for the path between any two points on the map is computed by the Geometric distance and the integral risk measure:

$$A^i_{p_1,p_2} = d_{p_1,p_2} + K \times \max_j \int_C F^j(x, y)\, ds \tag{5}$$

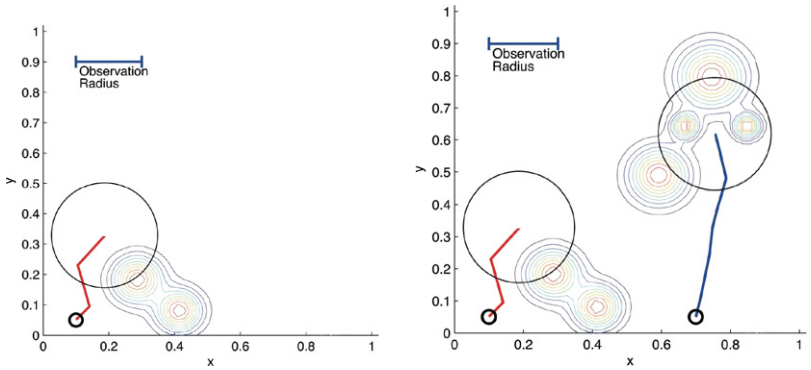
$A^i_{p_1,p_2}$ is first calculated by the threats information of the ith UAV, but further updated by the jth UAV of the maximum risk value, and C is the point set on the path from p_1 to p_2. d_{p_1,p_2} is the Geometric distance between p_1 and p_2, and K is the parameter related to the degree of threat, which affects the weight between two points. $F(x, y)$ is given in Eq. (2). By considering one threat as depicted in Fig. 1, we can visualize an example of threat at (0.5, 0.5) in a map. As shown in Fig. 3, we illustrate examples of the weight matrices calculated based on Eq. (5) from (0, 0) to other points when $K = 10$ and $K = 200$ with $F(x, y)$. Compared to Fig. 3(a), the values in Fig. 3(b) are much larger for each point in the map.

The advantages of the proposed Individual Weight Matrix are as follows:



(a) $K = 10$. (b) $K = 200$.

Fig. 3. Illustration of the weight values from (0, 0) to other points with different Ks.



(a) Illustration of observation radius. (b) Information sharing.

Fig. 4. Effect of information sharing for path planning.

(1) When the traditional Q-Learning approach has to rely on a set of complicated state-action rules to provide more choices for further action in the path planning, the proposed method is not restricted to the eight directions (Fig. 2).
(2) The Geometric distance and integral risk measure are embedded into the weight matrix, which contributes to a more reasonable path planning for UAVs.

4. Cooperative and geometric learning algorithm

CGLA is a general method designed for path planning of both single and multiple UAVs. For a single UAV, CGLA is executed when a new threatening object is detected and the weight matrix A is updated. For multiple UAVs, CGLA is scheduled to be performed when A is changed due to the detected risk and the dynamic information shared by other UAVs.

4.1. The idea of CGLA

In this section, we aim to minimize both the integral risk and the length (time) of the path. The risk matrix M and time or length matrix T on a path can be computed by:

$$M_i = \int_C F(x, y)\, ds \tag{6}$$

where C is the point set for a given path.

$$T_i = \int_C v ds \tag{7}$$

where v is the speed of a UAV. Assuming that the speed v is a constant, and the time cost is simply proportional to the length of the path. The optimal path C^* considering both the risk measure and path length can thus be found using the following objective:

$$C^* = \arg\min_C \sum_i (T_i + K \times M_i) \tag{8}$$

where C is any set containing a sequence of points from the beginning to the target on the map, T_i, M_i are the current length and risk matrix, and K is the parameter as mentioned in Eq. (5). The larger K is, the less risky the path found by the algorithm would be. To find the optimal path from the above weight matrix $T + K \times M$ from the current step, the dynamic programming method is usually used to search exhaustively for an optimal path. However, the DP approach cannot be used directly for efficient path planning of UAV when the map information or the weight matrix is dynamically changing. In this paper, we introduce a meaningful cost matrix to describe the dynamic environment in path planning and the computation of the proposed cost matrix is proved to be achieved in finite steps. Based on such computational cost, a path planning algorithm can be designed with a polynomial time complexity.

Based on the weight matrix, we also proposed an update mechanism to support information sharing among multiple UAVs. When working independently, each UAV can only know the information from the threatening objects located within its detection radius due to the limited view scope. However, if UAVs can share the information with each other, the view scope of UAVs can be greatly expanded. Fig. 4 gives an example of information sharing between two UAVs, where the center and radius of the circle show

the location and detection radius of UAV, respectively. As shown in Fig. 4(b), UAV on the left can know the information detected by the right one through cost matrix update, even it just took off from a starting point. The shared information can help the left UAV choose a more reasonable route in path planning. Multiple UAVs can cooperate together to complete a task. When new dangerous objects are visible to any UAV, the risk information on the map and the weight matrices are then updated accordingly. Moreover, if the distance between two UAVs is too small, each UAV has to be set as a virtual risk object for the other UAVs. These scenarios will be further addressed in the simulation part.

The weight matrix is a significant aspect of the proposed CGLA, and it will be dynamically changed in the following two cases:

(1) For single UAV, if any threatening object enters the circle area of the Observation Radius (OR), the threat information and the weight matrix will be updated according to Eqs. (1) (2) (5).
(2) For multiple UAVs, if we denote the safety distance between two UAVs, i.e. the minimum distance that two UAVs must maintain, as SD, we will check whether there is any UAV entering the circle area with high risk measured by the ratio of OR to SD. If any, the threat information in the map will be updated according to the position of the threatening UAV based on Eq. (1), and the weight matrix is then accordingly updated. The ratio γ of SD to OR is empirically set to be bigger than 0.1 in this paper.

4.2. Description of CGLA

The key idea of CGLA is how to calculate the cost matrix G defined below, which can be used to find the optimal path from a starting point to a target point in terms of distance and integral risk. Each element in the matrix G is defined to be the sum of cost from its position to a target point.

4.2.1. Calculating the cost matrix G for each UAV

(1) Assign G_{p_t} for a target point p_t with 0, and other points with $+\infty$ (a large value).
(2) Update the cost matrix G using the procedure as described below: First choose a point $p_r \mid r \in (1, N^2)$ on the map, and update the cost value G_{p_i} (for the sth UAV) based on values assigned to other points as follows:

$$G_{p_i}^{k+1} = \min\left\{G_{p_i}^{k}, A_{p_i,p_r}^{s} + G_{p_r}^{k}\right\}, \quad r, i \in \{1, 2, \ldots, N^2\} \tag{9}$$

where k is the number of iterative round.
(3) Repeat step (2) until the matrix G for any UAV converges based on a given A_{p_i,p_r} (actually A_{p_i,p_r}^{s}). We will provide a proof about the convergence of the above process in the next theorem.

For the cost matrix update formula in Eq. (9), we need to prove two convergence results for each $G_{p_r}^{k}$ when k is increasing. First we give the following fact from Eq. (9).

Lemma 1. *For any point p_r in the map, the sequence $\{G_{p_r}^{k}\}$ is decreasing with k.*

One can see from Eq. (5) that the weight value A_{p_i,p_r} is positive for any two points; therefore, the cost value is also positive for each point. According to Lemma 1, each sequence $\{G_{p_r}^{t}\}$ will converge as it is lower bounded as well as decreasing. Next, we will further prove that such convergence can be achieved in finite steps. For clarity of presentation, we first describe in detail the update steps in Eq. (9), and then give a theorem with formal proof.

For each point p_r, the initial value of $G_{p_r}^{0}$ will be zero if it is a target point and infinity (a large value in numeric) otherwise. In the next round, $G_{p_r}^{1}$ is determined only by a target point p_t as $G_{p_t}^{0} = 0$, and the values on all other points are very large. Therefore $G_{p_r}^{1} = A_{p_r p_t} + G_{p_t}^{1} = A_{p_r p_t}$. This concludes that in the first iteration, the cost matrix is, in fact, equal to its weight matrix.

Now we define a non-empty set of points excluding the target point as below:

$$\Delta_1 = \{p_i \mid G_{p_i}^{1} = \min\{G_{p_r}^{1}\},\ p_r \in \{1, 2, \ldots, N^2\} \setminus \{p_t\}\}. \tag{10}$$

One can see that Δ_1 is actually a set of points in which $G_{p_r}^{1}$ achieves its smallest value excluding the target value and of course it is not empty. Also we can observe from Eq. (9) that $G_{p_t}^{1} = 0$, which implies that the cost value at the target point is unchanged in this iteration. In order to update the cost matrix in the next round, we can choose one point $p_{r1} \in \Delta_1$, then compute $G_{p_r}^{2}$ for any point p_r using the update rule in Eq. (9). Now we can observe that $G_{p_t}^{2} = 0$ from $G_{p_t}^{1} = 0$. We also have $G_{p_{r1}}^{2} = G_{p_{r1}}^{1}$, which implies that the value $G_{p_{r1}}$ is unchanged in this round. Similarly, we define

$$\Delta_2 = \{p_i \mid G_{p_i}^{2} = \min\{G_{p_r}^{2}\}, \quad p_r \in \{1, 2, \ldots, N^2\} \setminus \{p_t\} \cup \{p_{r1}\}\}. \tag{11}$$

Now we can select one point $p_{r2} \in \Delta_2$, and there are two possible cases. One is that a target point p_t leads to a smaller $G_{p_{r2}}^{2}$, i.e., $G_{p_{r2}}^{2} = A_{p_{r2} p_t} + G_{p_t}^{1} = A_{p_{r2} p_t}$. The other case is that p_{r1} makes $G_{p_{r2}}^{2}$ smaller, i.e., $G_{p_{r2}}^{2} = A_{p_{r2} p_{r1}} + G_{p_{r1}}^{1} = A_{p_{r2} p_{r1}} + A_{p_{r1} p_t}$. This process can continue iteratively and we will have the following theorem for the convergence result with the following notations.

$$\Delta_m = \{p_i \mid G_{p_i}^{m} = \min\{G_{p_r}^{m}\}, p_r \in \{1, 2, \ldots, N^2\} \setminus \{p_t\} \cup \{p_{r1}\} \cup \{p_{r2}\} \cdots \cup \{p_{rm-1}\}\}. \tag{12}$$

Theorem 1. *Assume that the weight value A_{p_i,p_r} is consistently positive for any two points in the map and the initial values of $G_{p_r}^{0}$ are set in the step 1 with two points $\{p_s, p_t\}$. The update rule in Eq. (9) will generate a sequence $\{G_{p_r}^{k}\}$ for each point p_r with the following properties.*

(1) *Each sequence $\{G_{p_r}^{k}\}$ will converge in finite steps.*
(2) *$G_{p_t}^{k} = 0$ for all k and $G_{p_{rl}}^{k}$ will keep unchanged after finite steps for any other point.*
(3) *From the target point p_t, one can find the starting point p_s in Δ_i in finite steps.*

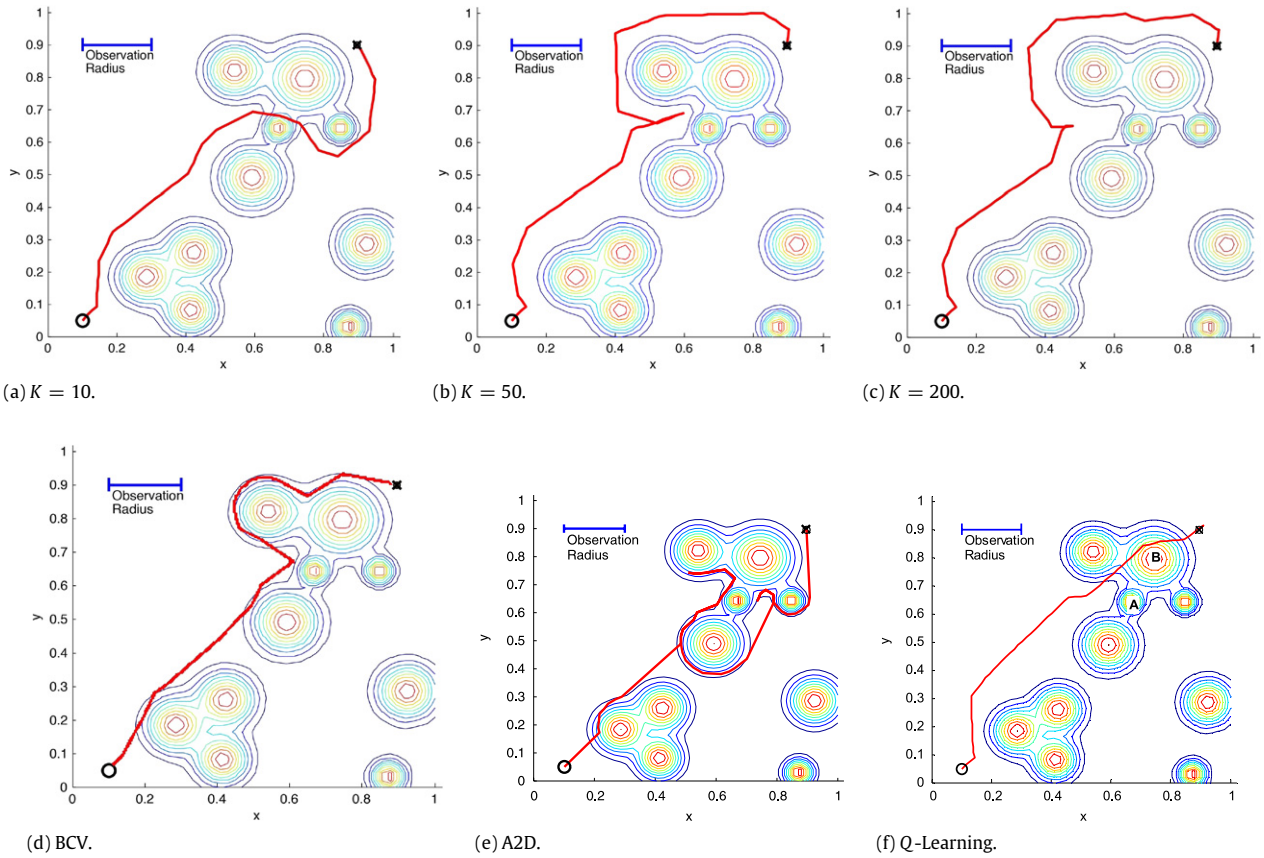
Proof. (1) The convergence is obvious from Eq. (9) as mentioned previously. In order to prove its convergence in finite steps, we only need to prove part (2). Recall that the possible elements in Δ_i are decreasing in each iteration and every point in the map will appear in Δ_i after some finite steps, so (3) is obvious from (2). So we will focus on the proof on part (2).

In fact, $G_{p_t}^{k} = 0$ is obvious from Eq. (9). Since $G_{p_{rk}}^{k}$ is achieved either by the target point p_t directly with A_{p_{rk},p_t} or by the previously selected points $\{p_{r1}, p_{r2}, \ldots, p_{rk-1}\}$, we can see that for each $p_{r_k} \in \Delta_k$, $G_{p_{rk}}^{k}$ is the minimum among all $G_{p_r}^{k}$. Therefore, we have the following from (9):

$$G_{p_{rk}}^{k+1} = \min_{r \in \{1,2,\ldots,N^2\}} \left\{G_{p_{rk}}^{k}, A_{p_{rk},p_r} + G_{p_r}^{k}\right\} = G_{p_{rk}}^{k}. \tag{13}$$

This implies that $G_{p_{rk}}^{l}$ will converge when $l \geq k$.

Given a weight matrix, the above searching procedure will give a path from a starting point to a target point with the minimal cost value G in finite steps. Path planning in dynamic environment will follow in the next section.



(a) $K = 10$. (b) $K = 50$. (c) $K = 200$.

(d) BCV. (e) A2D. (f) Q-Learning.

Fig. 5. An example of CGLA with different values of K and the results of the comparative methods.

4.2.2. Path planning algorithm in dynamic environment

In this subsection, we will propose a path planning algorithm CGLA for multiple UAVs in a dynamic environment based on the cost function defined in Eq. (9). As observed from last section, once A_{p_i,p_j} is given, one can find a path from p_s *to* p_t *for each UAV. Once each UAV moves one step towards the target point, the environment may have changed and we need to recalculate* A_{p_i,p_j}.

Path planning algorithm (CGLA)

Step 1. Given the two points $\{p_s, p_t\}$ with p_s being the starting point and p_t the target point for each UAV, we can calculate A_{p_i,p_r} for any two points.

Step 2. Compute the cost matrix as described in Section 4.2.1 starting from the target point. We check the following condition in *step* 2 for each iteration: if $p_s \in \Delta_k$ for some k, then stop.

Theorem 1 will guarantee that the above procedure will stop in finite steps. Therefore a path can be found with all the points $p^0, p^1, p^2, \ldots, p^M$ ($p^0 = p_s, p^M = p_t$).

Step 3. UAV will move to p^1, i.e., move one step to the target.

Step 4. Update all A_{p_i,p_r} according to the rules in Section 4. 1 to describe environmental changes after UVA's previous movement.

Step 5. With the updated A_{p_i,p_r} and p^1 as a new starting point, implement the above *step* 2 in the changed environment *excluding the previously selected point* p^0. Then we can find the next point the UAV will move to.

Step 6. Continue this process until the target point is reached.

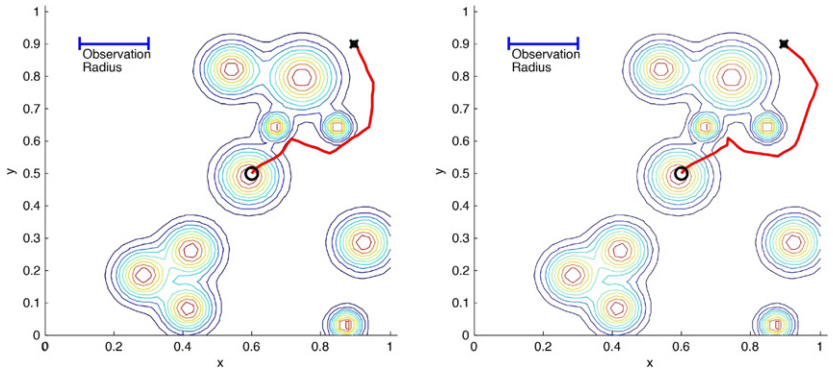
The above procedure uses the greedy algorithm to find each point towards a target point. As there are only finite points in a defined environment, the above algorithm will definitely converge to the target point as the embedded algorithm in step 2 always converges in finite steps. We have the following comments on the proposed algorithm.

Remark 1. The paths found in Section 4.2.2 will not be unique if Δ_k has more than one element before p_s is reached. The quality of different paths can be evaluated based on the proposed criterion in terms of path length and risk metric.

Remark 2. The proposed CGLA algorithm will converge in finite steps and we will give its complexity analysis in the next subsection. Here we should note that if we do not put a constraint of excluding the previously selected points in step 5, the path planning algorithm may theoretically have a chance to be stuck in a given region in the case that the previously selected points are selected as next point again in step 5. In fact such chance is very rare only with theoretical significance. Extensive simulations conducted in Section 5 show that this never happens due to possible random changes of the updated A_{p_i,p_r}. The two possible algorithms with or without such constraint always produce the same results in our simulations. This indicates that we can remove such constraint in step 5 in a practical implementation.

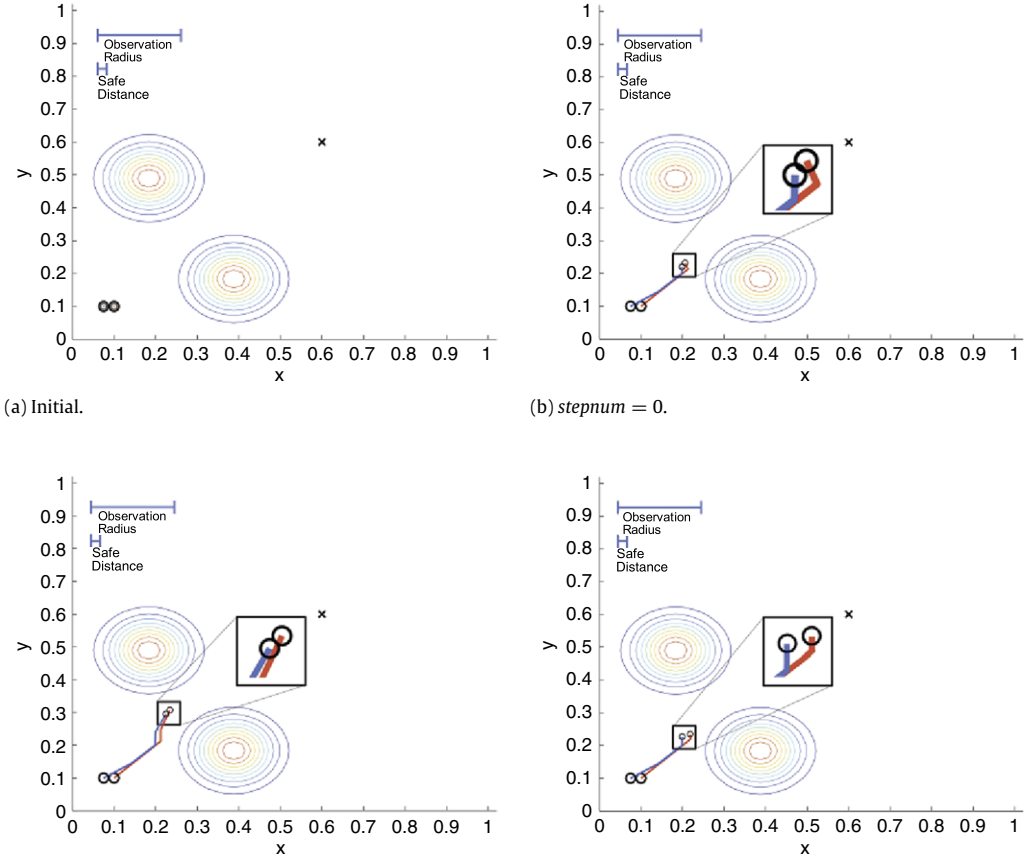
4.2.3. Complexity analysis

In this section, we will analyze the complexity of the path planning algorithm in Section 4.2.2. For clarity, here we use a variable T to represent the number of steps of the optimal path searching for any two points in a given map. T is a random number and here we use its average value $\overline{T}$ for complexity analysis. The whole complexity of the path planning algorithm in Section 4.2.2 can be decomposed as the following parts.



(a) $K = 10$. (b) $K = 200$.

Fig. 6. Illustration of CGLA path planning for a sudden threat.



(a) Initial. (b) *stepnum* = 0.

(c) *stepnum* = 5. (d) *stepnum* = 3.

Fig. 7. The evaluation of *stepnum* on path planning of UAV.

(1) Given N^2 points in a defined environment, the complexity of computing all A_{p_i,p_r} is N^4.

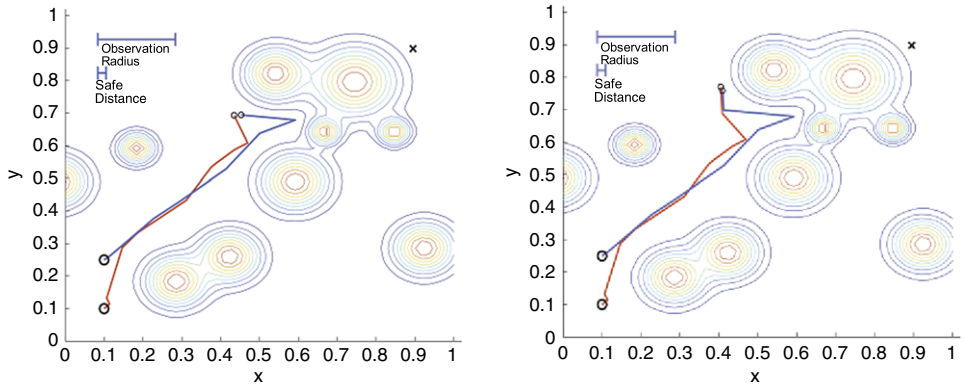
(2) The complexity of finding a shortest path in Section 4.2.1 is mainly coming from finding the minimal value G in each round from Δ_k. There is a sorting algorithm with complexity $O(N^2 \ln(N^2))$ (Sedgewick & Bentley, 2002). Therefore the total complexity for algorithm in 4.2.2 is $O(\overline{T}N^2 \ln(N^2))$.

(3) In step 5 of Section 4.2.2, the complexity of updating all A_{p_i,p_r} is N^4.

(4) Then each UAV moves one step from the starting point to the target point and starts from a new starting point. So the total complexity of the path planning algorithm after reaching a target point is

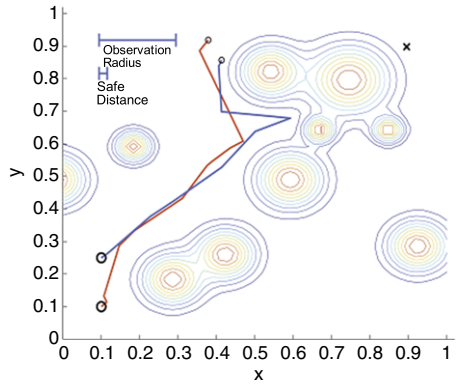
$$\overline{T}N^4 + O(\overline{T}^2 N^2 \ln(N^2)) = O(\overline{T}N^4 + \overline{T}^2 N^2 \ln(N^2)). \tag{14}$$

Remark 3. One can find from above analysis that the main complexity is from computing A_{p_i,p_r}. However, we should remember that in the process of updating A_{p_i,p_r} in step 5 of Section 4.2.2, if only the A_{p_i,p_r} in the region with its path passing through needs to be updated and its complexity should be much less in this case. Also the average value $\overline{T}$ is usually much smaller than N. Therefore, the complexity is roughly between $O(N^4)$ and $O(N^5)$. With this complexity, the proposed path planning algorithm CGLA is in fact efficient with current computer capacity.

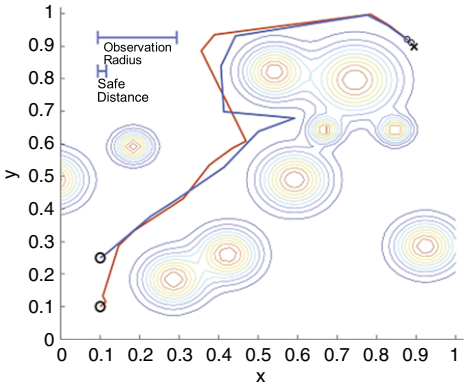


(a) Before collision.

(b) Without a virtual obstacle.



(c) With a virtual obstacle.



(d) The final result.

Fig. 8. Illustration of CGLA with a virtual obstacle model.

5. Computer simulation results

Extensive simulations are performed in this section. The same map was used for all competing methods like Behavior Coordination and Virtual (BCV) (Wu et al., 2011), Q-Learning (Zhang, Mao, Liu, & Liu, 2013), CPVS (Zhao et al., 2011), and FBCRI (Zheng et al., 2011) (with information sharing for multiple UAVs). The BCV method (Wu et al., 2011) is a real-time path planning approach based on coordination of the global and local behaviors. While most of the existing works only use path length as performance criterion, we include the integral risk for better measurement of path planning. The comparative figures and simulation results are cited from their original papers mentioned above and the paper (Zhang, Mao, Liu, Liu, & Zheng, 2013).

5.1. Comparison results for a single UAV

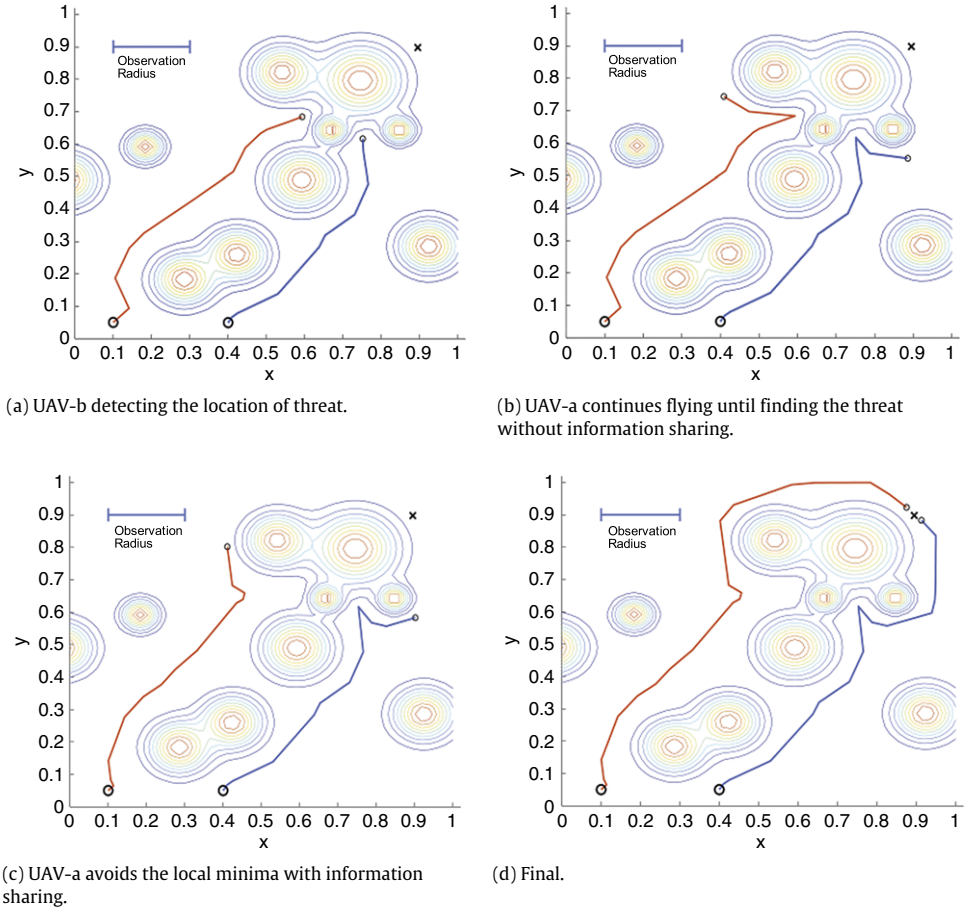
In this section, we evaluate the performance of CGLA for a single UAV. The threatening region is gradually detected when the distance between the UAV and the threaten object is less than the detection radius. The UAV is assumed to fly from starting point (0.1, 0.1) to the target (0.9, 0.9). The results of CGLA with different values of K are shown in Fig. 5. As shown in Fig. 5(a)(b), UAV is likely to choose a shorter-distance but higher-risk path when K is smaller. For example, the UAV passes through regions (0.5, 0.5) and (0.8, 0.8) to reach the target point when $K = 10$. When a large value is set for K, a UAV will fly away from the threat obstacles to reduce possible risks.

The comparison results with BCV are shown in Fig. 5(c)(d). Being different from CGLA, the BCV method suffers from local minima when only path length is considered in the objective function. Therefore, the BCV may produce a path with high risk as shown in Table 1. For example, if the UAV flies for a long time around a threat region, the cost on time should be much higher than that of passing through the high-threat areas directly to the target point. The CGLA learning is better than BCV as it considers both risk measure and path length together. It can be seen from Table 1 that in the case of $K = 10$, CGLA is superior in terms of the risk measure and path length. When the path length is similar, the risk of CGLA is only 1/4 that of BCV. As shown in Fig. 5(d), the path of BCV includes a long route close to the region of high risk. Both A2D and Q-Learning produce a path passing through high risk regions, which make their paths much more risky than that of CGLA. Specifically, Q-Learning method is much more risky than other methods as it is restricted to the eight directions in selecting the next action.

Table 1
Comparison of GCLA and other methods in the risk measure for single UAV.

Planning method	Risk measure	Path length
BCV	10.1658	256
A2D	24.3638	431
Q-Learning	27.7908	204.8
CGLA, $K = 10$	2.4318	243
CGLA, $K = 50$	0.6356	298.6
CGLA, $K = 200$	0.1387	273.2

In summary we propose a new performance measure including risk and path length for evaluation. The comparative results are shown in Fig. 5 and Table 1. While the path lengths in BCV, A2D and Q-Learning methods are re-scaled into the same unit, the value



(a) UAV-b detecting the location of threat.

(b) UAV-a continues flying until finding the threat without information sharing.

(c) UAV-a avoids the local minima with information sharing.

(d) Final.

Fig. 9. Illustration of CGLA with information sharing.

Table 2
Risk measure and path length for sudden threat testing.

Planning method	Risk measure	Path length
CGLA, $K = 10$	1.9896	105
CGLA, $K = 100$	1.5393	110

of risk measure is calculated as its original value. As we only have the source code of Q-Learning, we only compared time costs with this algorithm here. For example, CGLA spends 7.636752 s and Q-Learning takes 26.4662 s for path planning with the map given in Fig. 7 on a PC with 2.9 GHz 2 CPU. The proposed CGLA is much more efficient than Q-Learning.

On the other hand, as a pilotless vehicle, UAV should respond immediately to any sudden changes and avoid unexpected threats in a route to its destination in real time. To be an effective path planning method, CGLA has to provide an escape path, when dangerous objects are detected. In fact, CGLA is simple, reliable and robust for any possible sudden threat as this method can adaptively adjust to the updated weight matrix. As shown in Fig. 6, when a threatening object is suddenly detected in the point $(0.6, 0.5)$ where UAVs have to escape to the target point $(0.9, 0.9)$, the weight matrix could be updated to include the new risky objects. With the updated weight matrix, CGLA can successfully choose a path with small risk to avoid the dangerous object. The simulation results are shown in Fig. 6(a), (b) with different Ks. From Table 2, we can see that the UAV can flee from the threatening area easily when a larger value of K is given, which leads to a path with smaller risk and longer distance. From this simulation result, we can conclude that CGLA can provide an efficient path planning to flee from unexpected threats.

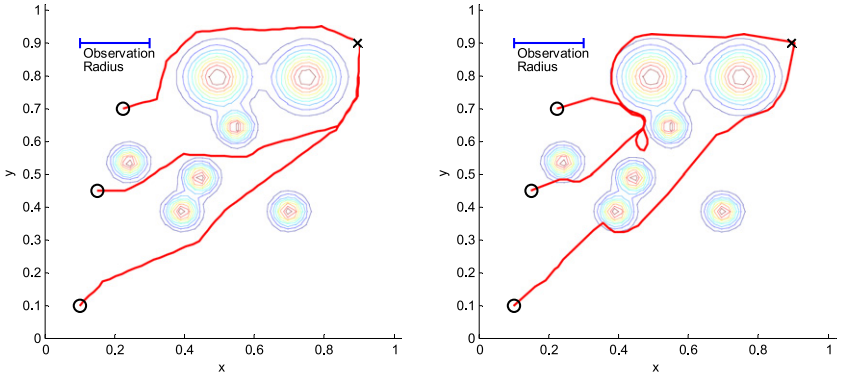
5.2. Path planning for multiple UAVs

5.2.1. Virtual obstacle setting

In order to avoid collision among a group of UAVs, we need to handle its nearby UAVs. A UAV can be treated as a threatening object when it flies into the detection radius of other UAVs. By considering movements of all UAVs, we can set the threat area in front of each UAV in its flight direction. The SD of two UAVs must be maintained. Here we introduce an idea of virtual obstacle for each UAV to keep a safe distance from each other and avoid collisions. The essence is that one UAV is considered as a threatening object or an obstacle for others in the process of path planning. Two situations are considered here. Firstly, if a UAV is detected in the observation radius and it is far away from the safe distance, then the UAV is considered as the traditional threatening objects as shown in Eq. (1). Secondly, when the distances of neighboring UAVs are less than a preset threshold SD, they are set as the virtual obstacles. The distance between the UAV and the virtual obstacle is denoted by $stepnum * 0.4$ unit, where $stepnum$ is a parameter, and 0.4 is empirically determined from the extensive computer simulation results.

Fig. 7 gives an example of two close UAVs with distance being smaller than SD. The effect of different $stepnum$ on path planning was firstly tested. If two circles linking to lines of different colors are separated, the collision of the two UAVs is avoided. Otherwise, the path for one UAV is colliding with the other. We can see from Fig. 7(a) and (c) that UAVs will collide with each other when $stepnum$ is set to 0 or 5. When the value of $stepnum$ is 3, UAVs can avoid collision and safely reach the target.

To further evaluate the performance of the virtual obstacle setting method, we conduct more simulations as shown in Fig. 8(a).



(a) $K = 10$. (b) FBCRI.

Fig. 10. Path planning results of multiple UAVs with $K = 10$ and the result of FBCRI.

Table 3
The evaluation of CGLA with information sharing.

Planning methods	Risk measure	Path length
Without information sharing	0.9825	281.6
With information sharing	0.5383	240

In this simulation, UAV a (*the red UAV*) starts from (0.1, 0.1), while UAV b (*the blue UAV*) starts from (0.1, 0.25), and the target for both UAVs is the point (0.9, 0.9). Fig. 8(b) shows the case when virtual threat is not used. Both UAVs fly at the same speed and cause a collision near location (0.5, 0.7). If the virtual obstacle is set for each UAV in this situation, UAV a adaptively change the path to avoid the collision (see Fig. 8(c)).

5.2.2. Collaborative path planning with information sharing

(a) The evaluation of information sharing for path planning.

Different UAVs can share the risk information for the collaborative path planning. As a result, UAVs may have a more broad view scope, and local minima can be effectively avoided. The information sharing can be easily done in CGLA. In the process, we first change the environment by Eqs. (1) and (2), and the risk information is updated by the new risky objects detected by all UAVs. We further update the weight matrix and cost matrix for each UAV by the information from other UAVs according to Eqs. (5) and (9) respectively. The distance information in Eq. (5) remains unchanged. The environment information shared by all UAVs can contribute to expand the view scope of each UAV.

Fig. 9 shows the simulation results of two UAVs. When information was not shared among the UAVs, two threatening objects at locations (0.6, 0.7) and (0.7, 0.8) detected by one UAV are unknown to the other. Two UAVs still fly towards the area of high risk, and resulted in local minima. It is an obvious waste of operation time and energy, and also greatly increases the risk of the journey.

The quantitative comparison of the information sharing effects for multi-UAV path planning is shown in Table 3. By using the information from other UAVs, CGLA has a better performance as the view scope is greatly expanded and produces a much better path planning in terms of risk and path length. Compared to Q-Learning, the proposed method also achieves a much better performance.

(b) Qualitative evaluation of the proposed method to other methods.

To further qualitatively evaluate our method on collaborative path planning for multiple UAVs, we compare in this section our method with a multi-UAV collaborative path planning method, namely the feedback based CRI (FBCRI). FBCRI also uses the technology of virtual targets to avoid non-convergence for performance improvement. When $K = 10$, the simulation results calculated by

Table 4
Comparison of CGLA and other methods in risk measure.

Planning methods	UAV1	UAV2	UAV3
FBCRI	3.5326	4.5282	4.5576
CGLA, $K = 10$	0.2096	0.8282	0.4095

Table 5
Comparison of CGLA to other methods in path length.

Planning methods	UAV1	UAV2	UAV3
FBCRI	195	244	211
CPVS	188	234	199
CGLA, $K = 10$	186	157	125

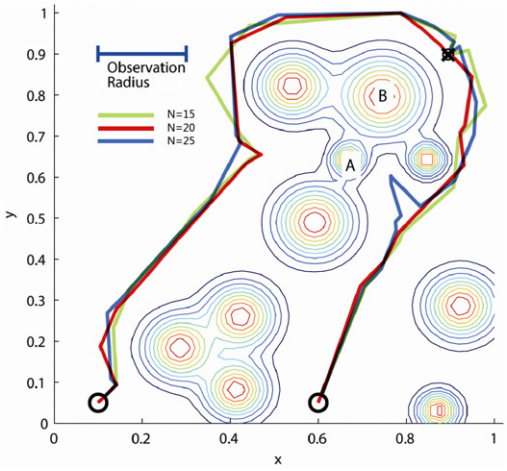


Fig. 11. The illustration of paths with different Ns for CGLA.

CGLA for UAV 1, 2, 3 flying from (0.1, 0.1), (0.15, 0.45) and (0.225, 0.7) to (0.9, 0.9) are shown in Fig. 10(a). While FBCRI needs the manually selected rules for next action, the results shown in Fig. 10(b) suggest that FBCRI is not flexible in adaptively avoiding various threatening objects.

The risk measure and path length for the two methods are shown in Tables 4 and 5. It can be seen from the tables that CGLA achieves a much better performance than FBCRI. Specifically, when the path lengths of CGLA and FBCRI are similar, the risk measure for CGLA is much smaller. Compared to CPVS, CGLA also achieves a much better performance as shown in Table 5.

5.2.3. Evaluation of the proposed method on different choice of Ns and different number of obstacles

Theoretically, with a large size of map N, the path planning usually takes longer as there will be more points in the space and the

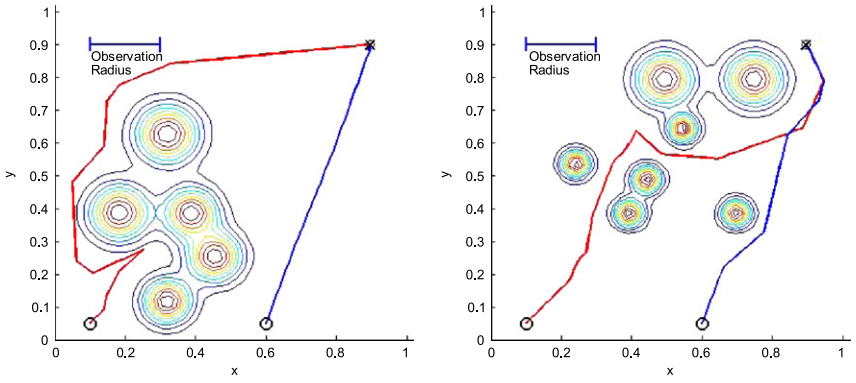


Fig. 12. The illustration of CGLA on new scenarios.

complexity will also increase. Here we do some simulations with different choice of N. The results for $N = 15, 20, 25$ with $K = 40$ are shown in Fig. 11, which shows that the method is robust to the variation of N as the path is nearly the same.

Finally we implement CGLA with different number of obstacles for two UAVs. We set 5 and 7 risky objects in this case, which are different from the former map. As shown in Fig. 12, our method can still work well on the new maps, which further validates its effectiveness in different cases.

6. Conclusions

In this paper, we propose a new Cooperative and Geometric Learning method to solve the path planning problem for multiple UAVs. Compared with other existing methods, CGLA leads to a very simple Path Planning solution. The parameters K and N well studied in CGLA can balance between the safety and economy of the paths. K can be adjusted to be suitable for different kinds of tasks of UAVs, which are designed to find the path with more flexibility. CGLA is also designed for multi-UAV collaborative planning. The collisions of different UAVs can be avoided using the virtual obstacle strategy with specific pre-defined parameter (*stepnum* together with a risk factor k). Meanwhile, we also verify that information sharing based on the individual weight matrix and the cost matrix among UAVs is very effective for navigation. By using the scheme of information sharing, the view scope of each UAV can be expanded, and a shorter path with smaller risk can be achieved. The computational complexity of the CGLA is in order of $O(N^4)$ or $O(N^5)$, which increases polynomially with number of N. All our simulations are conducted using a PC with 2.9 GHz 2 CPU and all of them completed efficiently.

Our future work will focus on how to set K values and updating A_{p_i,p_r} *adaptively* for more efficient path planning. Another future work is to use the coarse and fine grid to design fast path planning algorithms in the proposed framework. We will also try to apply the algorithm to a real UAV to test its effectiveness in practical applications.

References

Bauso, D., Giarrè, L., & Pesenti, R. (2004). Multiple UAV cooperative path planning via neuro-dynamic programming. In *Proc. of IEEE conference on decision and control* (pp. 1087–1092).

Beard, R. W., & Stepanyan, V. (2003). Information consensus in distributed multiple vehicle coordinated control. In *IEEE conference on decision and control: vol. 2* (pp. 29–2034).

Bellingham, John, Richards, Arthur, & How, Jonathan P. (2002). Receding horizon control of autonomous aerial vehicles. In *IEEE proceedings of the American control conference* (pp. 3741–3746).

Bortoff, Scott A. (2008). Path planning for UAVs. In *IEEE proceedings of the American control conference* (pp. 36–364).

Cai, K. Y., & Zhang, L. (2008). Fuzzy reasoning as a control problem. *IEEE Transactions on Fuzzy Systems*, *16*(3), 600–614.

Chandler, P.R., & Pachter, Meir (2002). Complexity in UAV cooperative control, In *The proceedings of the American conference* (pp. 1831–1836).

Chandler, P.R., & Rasmussen, S. (2000). UAV Cooperative path planning. In *AIAA Guidance, navigation, and control conference and exhibition* (pp. 1–11).

Dogman, A. (2003). Probabilistic approach in path planning for UAVs. In *Proc. of IEEE international symposium on intelligent* (pp. 608–613).

Even-Dar, Eyal, & Mansour, Yishay (2003). Learning rates for Q-learning. *Journal of Machine Learning Research*, *5*, 1–25.

Flint, M., & Fernandez, E. (2005). Approximate dynamic programming methods for cooperative UAV search. In *Proc. of IFAC world congress: vol. A4* (pp. 59–64).

Flint, M., Polycarpou, M., & Fernandez-Gaucherand, E. (2002). Cooperative control for multiple autonomous UAV's searching for targets. In *IEEE conference on decision and control* (pp. 2823–2828).

Girard, A., Darbha, S., Pachter, M., & Chandler, P. (2007). Stochastic dynamic programming for uncertainty handling in UAV operations. In *Proc. of the American control conference* (pp. 1079–1084).

Kim, Yoonsoo, Gu, Da-Wei, & Postlethwaite, Ian (2006). Real-time path planning with limited information for autonomous unmanned air vehicles. *Automatica*, *44*, 696–712.

Mao, Z., Zhang, B., & Xu, B. (2012). Path planning of UAV based on Q-learning. http://www.paper.edu.cn/releasepaper/content/201203-807.

Nikolos, Ioannis K., Zografos, Eleftherios S., & Brintaki, Athina N. (2007). *Studies in computational intelligence: vol. 70. UAV path planning using evolutionary algorithms* (pp. 77–111).

Rabbath, C. A., Gagnon, E., & Lauzon, M. (2004). On the cooperative control of multiple unmanned aerial vehicles. *IEEE Canadian Review*, 15–19.

Sedgewick, Robert, & Bentley, Jon (2002). *Quicksort is optimal*. Knuthfest: Stanford University.

Shanmugavel, Madhavan, Tsourdos, Antonios, & White, Brian A. (2010). Collision avoidance and path planning of multiple UAVs using flyable paths in 3D. In *International conference on methods and models in automation and robotics*, MMAR (pp. 218–222).

Watkins, Christopher J.C.H. (1989). Learning from delayed rewards, Ph.D. Thesis King's College, Cambridge.

Wu, Shanjie, Zheng, Zheng, & Cai, Kai-yuan (2011). Real-time path planning for unmanned aerial vehicles using behavior coordination and virtual goal. *Control Theory & Applications*, *28*(1), 1–4.

Zadeh, L. A. (1974). The concept of a linguistic variable and its applications to approximate reasoning. *Information Science*, *8*, 199–249.

Zhang, B., Mao, Z., Liu, W., & Liu, J. (2013). Geometrical reinforcement learning for path planning of UAV. *Journal of Intelligent & Robotic Systems*, http://dx.doi.org/10.1007/s10846-013-9901-z.

Zhang, B., Mao, Z., Liu, W., Liu, J., & Zheng, Z. (2013). Cooperative and geometric learning for path planning of UAVs. In *ICUAS*.

Zhao, Limeng, Zheng, Zheng, Liu, Wei, Cai, Kai-Yuan, & Lin, Shumin (2011). Real-time path planning for multi-UAVs with share of threats information. In *IEEE conference on industrial electronics and applications* (pp. 1359–1364).

Zheng, Changwen, Li, Lei, & Fan, Jingxu (2005). Evolutionary route planner for unmanned air vehicles. *IEEE Transactions on Robotics*, *21*(4), 609–620.

Zheng, Zheng, Wu, Shanjie, Liu, Wei, & Cai, Kai-Yuan (2011). A feedback based CRI approach to fuzzy reasoning. *Applied Soft Computing*, *11*(1), 1241–1255.

Baochang Zhang received the B.S., M.S., and Ph.D. degrees in Computer Science from the Harbin Institute of Technology, Harbin, China, in 1999, 2001, and 2006, respectively. From 2006 to 2008, he was a research fellow with the Chinese University of Hong Kong, Hong Kong, and with Griffith University, Brisbane, Australia. Currently, he is an associate professor with the Science and Technology on Aircraft Control Laboratory, School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. He was supported by the Program for New Century Excellent Talents in University of Ministry of Education of China. His current research interests include pattern recognition, machine learning, face recognition, and wavelets.

Wanquan Liu received the B.Sc. degree in Applied Mathematics from Qufu Normal University, PR China, in 1985, the M.Sc. degree in Control Theory and Operation Research from Chinese Academy of Science in 1988, and the Ph.D. degree in Electrical Engineering from Shanghai Jiaotong University, in 1993. He once held the ARC Fellowship, U2000 Fellowship and JSPS Fellowship and attracted research funds from different resources over 2 million dollars. He is currently an Associate Professor in the Department of Computing at Curtin University and is in editorial board for seven international journals. His current research interests include large-scale pattern recognition, signal processing, machine learning, and control systems.

Zhili Mao received the B.S., degree in Automation Science from Beihang University, Beijing, China in 2012. From 2013 he was a postgraduate student getting his M.Phil. degree in Fok Ying Tung. Graduate School, Hong Kong University of Science and Technology, *Clear Water Bay, New Territories, Hong Kong.*

Jianzhuang Liu received the Ph.D. degree in Computer Vision from The Chinese University of Hong Kong, Hong Kong, in 1997. From 1998 to 2000, he was a research fellow with Nanyang Technological University, Singapore. From 2000 to 2012, he was a postdoctoral fellow, then an assistant professor, and then an adjunct associate professor with The Chinese University of Hong Kong. He joined Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, as a professor, in 2011. He is currently a chief scientist with Huawei Technologies Co. Ltd., Shenzhen, China. He has published more than 100 papers, most of which are in prestigious journals and conferences in computer science. His research interests include computer vision, image processing, machine learning, multimedia, and graphics.

Linlin Shen received Ph.D. degree from University of Nottingham, UK in 2005. From 2005 to 2006, he was a research fellow with University of Nottingham, working on MRI brain image processing. He has been with Shenzhen University, China since 2006 and is currently a professor at the School of Computer Science and Software Engineering. His research interests include computer vision, image processing, and pattern recognition. He received the Most Cited Paper award from the journal of Image and Vision Computing in 2010 and his team was the winner of International Competition on Cells Classification by Fluorescent Image Analysis organized by ICIP 2013.