Guidance and Navigation of Unmanned Aerial Vehicles with Applications to Obstacle Avoidance and Forest Fire Monitoring

By: Yintao Zhang
Supervisor: Professor Youmin Zhang

Department of Mechanical, Industrial and Aerospace Engineering

Presented in Partial Fulfillment of the Requirements for the Degree of
Doctorate of Philosophy (Ph.D.) in Mechanical Engineering at

Concordia University
Montreal, Quebec, Canada

May 2019

Abstract

As there has been growing interest worldwide in commercial, scientific and civil issues of unmanned aerial vehicles (UAVs), the demand for developing advanced guidance and navigation capabilities becomes more urgent. However, until now, the guidance and navigation of UAVs still needs human operations or at least semi-autonomous. Fully autonomous UAV navigation system is facing several challenges, such as sophisticated and hazardous environment. However, traditional path planning algorithms are weak at dealing with complicated environment especially for some small changes in the map. Reinforcement Learning (RL) is an approach to artificial intelligence that emphasizes learning by an agent from its interaction with the environment. The goal of the RL algorithm is to learn what actions the agent selects to take in a specific situations by learning a value function of situations or states. The learning algorithm is not conducted which actions to take, but it has to discover an optimal action in each state which yields the high rewards in a long-term objective. Which gives RL the ability to react to the local changes of the map as well as the global objective. In addition to path planning algorithm, path following is also necessary and crucial to accomplish the guidance and navigation mission since the environment disturbance cannot be ignored for small UAVs during the flight. Besides giving an overall picture of the historical, current, and future progress of RL path planning algorithms and path following methods, this research proposal in particular aims at outlining the research which is conducted to design and develop novel guidance and navigation strategies with applications to both individual and multiple UAVs. In order to ensure their efficient and reliable performance under sophisticated and hazardous environments, this proposed research is intended to provide advanced levels of path planning and path following methods of UAVs. The schemes and strategies expecting from this research proposal are also verified by a series of simulations.

Contents

1 Introduction and Background

Unmanned aerial vehicles (UAVs) in both military and civil fields have been given much attention over the past several decades [2, 5, 6, 25, 33]. The increasing demand has brought into focus on several challenges associated with multiple UAV operation. Reducing UAV dependence on human operation is a major concern. Currently, the execution of UAVs still requires full attention of human operators. In order to successfully carry out a complex mission, the UAVs not only need to share information and cooperate with each other to improve the overall group performance, but also need to react quickly to the highly dynamic and challenging environment [31].

Guidance and navigation systems are of critical importance for the overall performance of UAVs because they are concerned with the transient motion behavior associated with the achievement of motion control objectives. Guidance and navigation system could be divided into two separate but interrelated problems: 1) the path planning problem and 2) the path following problem.

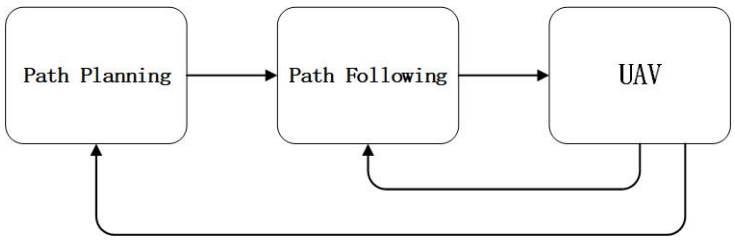


Figure 1: Block diagram of guidance and navigation systems

Path planning problem, as the name indicates, pertains to the procedure of determining which route to take. More specifically, the path planning problem of UAVs refers to determining a proper path from its position to a goal position in a particular environment without human intervention [8]. However, the environment may be imprecise, vast, dynamical, and partially non-structured [13]. In addition, the designed path have to satisfy the mission objective, such as collision-free, obstacle avoidance, search and cooperative requirement.

In addition to the path planning problem, it is of great importance to implement a guidance algorithm that will steer the UAV in a way such that it stays on the path, or lead it toward the path if the cross-track error (the shortest distance to the path) is nonzero. Therefore, for a UAV with a nonzero forward velocity, the guidance system in combination with the attitude controller should result in a stable overall system to ensure convergence to the desired trajectory. This constitutes the second problem that we are tackling in this proposal. It should be noted that trajectory tracking is not the same issue as path following, the main difference is that trajectory tracking requires the UAV to converge and follow a time-parameterized path. In [27], it is stated that the trajectory tracking problem is not appropriate for a small UAV because the fuel inefficiency and the object is too rigorous to achieve, which may lead the UAV to stall subject to wind.

The reminder of this proposal is organized as follows: Section 2 briefly introduces background, motivation, and literature review. Section 3 discusses the thesis scope and objectives. Section 4 outlines the applied research methodologies. Section 5 draws the timeline for appropriate completion of the proposed thesis. Section 6 presents the conclusion and potential future work. The research progress to date is illustrated in the appendix.

Table 1: Classification of RL path planning methods

RL Methods	Exploration Policy	Update Policy	Environment Model	Environment Status	References
CQL	Boltzmann ϵ-greedy	γ-policy state-chain	Grid-based Node-based	Static Dynamic	[15, 19, 20]
IQL	Boltzmann ϵ-greedy	Flag-policy	Grid-based	Static	[7, 16, 18]
GQL	Boltzmann ϵ-greedy Rank-based	Geo-policy γ-policy	Grid-based	Static Dynamic	[31]
Ant-Q	Pheromone Rank-based	HE-policy γ-policy	Grid-based Node-based	Static	[10, 12]
NN-QL	Boltzmann ϵ-greedy	γ-policy	Grid-based	Static Dynamic	[9, 28]

2 Literature Pertinent to the Proposal

In RL path planning methods, an agent performs an action on the environment and receives an immediate reward or penalty due to the action taken. The leaning algorithm adapts its parameter based on the status of the feedback (reward or penalty) signal from environment. Since the exact value of the futuristic reward is not known, it is guessed from the knowledge about the surrounding environment. The primary advantage of RL lies in its inherent power of automatic learning even in the presence of small changes in the environment.

As shown in Fig. 1, there exist extensive research on multi-agent navigation on RL that has been proposed.

There is a vast of literatures regarding the guidance task. Line-of-sight (LOS) guidance [11], the pure pursuit guidance [3] and the constant bearing guidance [3] are the most popular methods. Park [21] introduces a nonlinear guidance logic (NLGL) for fixed-wing UAV path following. The centrifugal acceleration is adopted as the lateral acceleration, and a fixed distance from the UAV to the desired path is used to adjust the rate of convergence. This guidance law is simple and intuitive to implement in most commercial autopilots. The vector fields approach is utilized in the straight line and loiter path following problem, and this study is extended to the curved path in [14]. In this approach, the cross tracking error is defined as the distance from the UAV to the closet point on the path, which may result in a singularity when the closet points exceeded one. To overcome this drawback, Rysdyk [26] proposes a guidance law that results in a constant line-of-sight orientation relative to the UAV, and the notion of convergence to within a "tube" around the desired path. The concept of adding integral action to the LOS guidance law is discussed in [4]. Borhaug [1] proposed a new integral LOS approach and considered both the kinematic and dynamic aspects of path-following when the task is to converge to a straight line.

3 Scope and Objectives of the Dissertation Research

This research aims to design and develop guidance and navigation systems with application to UAVs at both single agent and multiple agents levels. Particularly, the proposed research plan is organized around the following objectives:

1, Design and develop collision-free RL-based path planning methods for both single and multiple UAVs.

2, Design and develop threat avoidance RL-based path planning methods for both single and

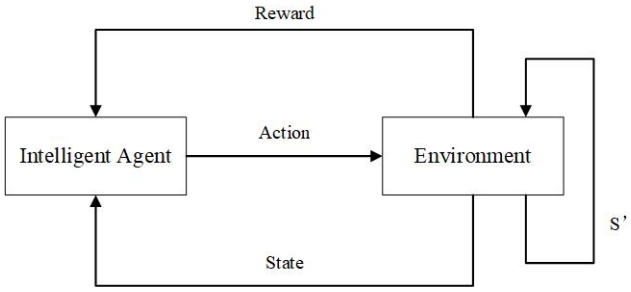


Figure 2: MDP model.

multiple UAVs.

3, Design and develop a real-time RL-based collision-free path planning method for both single and multiple UAVs in 3-D environment.

4, Design and develop forest monitoring path planning algorithm for both single and multiple UAVs.

5, Design and develop path following schemes for both Quadrotor and fixed wing UAVs.

The proposed research in brief is expected to synthesize advanced levels of guidance and navigation capabilities in UAVs, which in turn can guarantee the performance of UAVs in different environments at both individual and group levels. The schemes and strategies expected from this research proposal will be verified by a series of simulations and experiments on the Quardrotor and fixed-wing UAV benchmark.

4 Research Methodologies

4.1 Markov Decision Process Model

Solving the path planning problem with RL algorithms are usually based on a Markov Decision Process (MDP) model. As a result, a brief overview of the MDP model for path planning problem is given below.

A MDP is defined as a 4-tuple (S, A, P, R), where S is a finite set of states and A refers to all the actions of the agent. P stands for the transition probability and R presents a reward function. The expected reward R for a state-action pair (s, a) is defined as:

$$R(s,a) = \sum_{s' \in S} p(s,a,s') R(s,a,s') \tag{1}$$

where s is the current state, s' is the next state after the agent taking action a at state s.

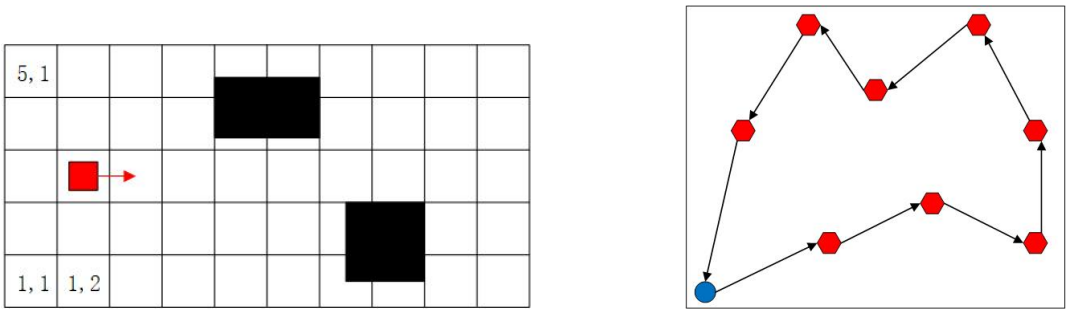
$Q(s, a)$ is the state-action pairs and their corresponding reward value. The exact Q values of all state-action pairs can be found by solving the linear system of the Bellman equations:

$$Q(s,a) = R(s,a) + \gamma \sum_{s' \in S} p(s,a,s') \sum_{a' \in A} Q(s',a') \tag{2}$$

For every MDP, there exists an optimal deterministic policy π^*, which maximizes the total expected reward from the initial state to the goal state. MDP is a discrete time stochastic control process. A MDP model is shown in Fig. 2. At each step, the process is in some state s, and the decision maker may choose any action a that is available in state s. The process responds

at the next time step by randomly moving into a new state s', and giving the decision maker a corresponding reward $R(s, a)$.

The probability that the process moves into its new state s' is influenced by the exploration policy. In other words, the next state s' is determined by the current state s and the chosen action a from the decision maker (exploration policy).



(a) Grid-based environment model (b) Node-based environment model

Figure 3: Environment model

As shown in Fig. 2, a path planning problem could be modeled as a MDP with the assist of four fundamental elements: environment model (shown in Fig. 3), state, action and reward function.

4.2 Classical Q-Learning Algorithm (CQL)

Classical Q-learning is one of the reinforcement learning in which the agent performs an action through state transition in the environment and receives a reward or penalty by executing the action to reach the goal state. The main aim of the agent is to learn the control strategy to select an action from the possible set action at the particular state to maximize the sum of the rewards from the specified start state to goal state through the state transition process.

Let $S_1, S_2, ..., S_n$ be the n possible states of an agent, and $a_1, a_2, ..., a_m$ be the set of m possible actions for each state. An agent selects an action from a set of possible n actions in each state and receives the specific reward that the agent acquires, which is known as immediate reward. Consider the immediate reward and total reward that the agent A acquires by executing an action a_j at state S_i is $r(S_i, a_j)$ and $Q(s_i, a_j)$ respectively. A policy is utilized to select the next state from current state, and this policy is used to to maximize the cumulative reward that the agent could be acquired during the transition process of states from the next state to the goal state. For example, let the agent be in state S_i and is expected to select the next best state. Then the Q-value at state S_i due to action of a_j is given below:

$$Q(S_i, a_j) = r(S_i, a_j) + \gamma MaxQ(\delta(S_i, a_j), a') \tag{3}$$

where $\delta(S_i, a_j)$ denotes the next state due to selection of action a_j at state S_i. Let the next state selected be S_k. Then $Q(\delta(S_i, a_j), a') = Q(S_k, a')$.

Pseudocode for CQL

1. Initialization
For each (S, a) pair, set $Q(S, a) = 0$;
2. Update

Repeat
Observe the current state S;
Select $a \in [a_1, a_2, ..., a_m]$ and execute it;
Observe the new state $S' \leftarrow \delta(S, a)$;
Update the Q-value $Q(S, a)$ by $Q(S, a) = r(S, a) + \gamma \underset{a'}{\text{Max}} Q(\delta(S, a), a')$;
Set the new state to current state $S \leftarrow S'$;
Until the end condition is satisfied.

4.3 Improved Q-learning Algorithm (IQL)

4.3.1 Improved Q-learning Properties

This section provides four interesting properties, based on which the IQL algorithm has been developed. The properties stress upon the following two issues

1) If the Q-value at any state S (S is known) needs no updating, the state is locked.

2) If S is a locked state, S' is a neighboring state of S, and any one of the four properties to be derived is applicable at the (S, S') pair, then we can compute the Q-value at sate S' once only using the property, and the S' is also locked.

Let S_p, S_n and S_G be the present, the next and the goal states, respectively. Let Q_p and Q_n be the Q-value at the present and next state S_p and S_n, respectively, for the best action. Let d_{xy} be the block distance between the states S_x and S_y. A Boolean variable Lock L_x is used to indicate that the Q_x value of a state is fixed permanently. The L_n is set to 1 ($L_n = 1$) if the Q-value of the state n is fixed and will not change further after the goal will be initialized to zero. Four interesting properties have been observed as indicated hereafter.

Property 1: If $L_n = 1$ and $d_{pG} > d_{nG}$, then $Q_p = \gamma \times Q_n$ and set $L_p = 1$.

Proof: Let the neighborhood state of S_p be $S \in S_a, S_b, S_c, S_n$, and the agent selects S_n as the next state as $d_{nG} < d_{xG}$ for $x \in a, b, c, n$.

Now

$$\begin{aligned} Q_p &= Q(S_p, a) \\ &= r(S_p, a) + \gamma \underset{a'}{\text{Max}} Q(\delta(S_p, a), a') \\ &= 0 + \gamma \underset{a'}{\text{Max}} Q(S_a|S_b|S_c|S_n, a') \\ &= \gamma \times Q(S_n, a')(\because d_{nG} \leq d_{xG}, \forall x) \\ &= \gamma \times Q_n \end{aligned} \tag{4}$$

Since $L_n = 1$ and $d_{pG} > d_{nG}$, $\therefore Q_p < Q_n$, and thus, $Q_p = \gamma Q_n$ for $0 < \gamma < 1$ is the largest possible value of Q_p, so Q_p should not be updated further. Therefore, $L_p = 1$ is set.

Property 2: If $L_p = 1$ and $d_{nG} < d_{pG}$, then $Q_n = Q_p/\gamma$ and set $L_n = 1$.

Proof: Since $d_{nG} < d_{pG}$, the agent will select the next state n from the current state p. Hence, by (4)

$$Q_p = \gamma \times Q_n \Rightarrow Q_n = \frac{Q_p}{\gamma} \tag{5}$$

Since $L_p = 1$ and $d_{nG} < d_{pG}$, $\therefore Q_p < Q_n$, and thus, $Q_n = Q_p/\gamma$ for $0 < \gamma < 1$ is the largest possible value of Q_n, so Q_n should not be updated further. Therefore, $L_n = 1$ is set.

Property 3: If $L_p = 1$ and $d_{nG} > d_{pG}$, then $Q_n = \gamma \times Q_p$ and set $L_n = 1$.

Proof: If the agent is moving from S_i to the goal G in one step, the immediate reward is R, for example. On the other hand, if the agent moves from S_i to any state other than the goal G, then the immediate reward is zero.

Now, suppose that the agent moves from S_p to the goal G in k. Now, as $L_p = 1$, k is the minimum number of state transitions to reach the goal from S_p. Therefore,

$$\begin{aligned} Q_p &= Q(S_p, a) \\ &= r(S_p, a) + \gamma \operatorname*{Max}_{a'} Q(\delta(S_p, a), a') \\ &= 0 + \gamma^k R \end{aligned} \tag{6}$$

Suppose that the agent moves from S_n to the goal G in $k+1$ transition steps. Here, $(k+1)$ is also the minimum number of steps to reach the goal G from S_n, failing which the agent would have selected some other state as the next state from the current state S_p

$$\begin{aligned} Q_n &= Q(S_n, a) \\ &= r(S_n, a) + \gamma \operatorname*{Max}_{a'} Q(\delta(S_p, a), a') \\ &= 0 + \gamma^{k+1} R \end{aligned} \tag{7}$$

Dividing (6) by (7),

$$\begin{aligned} \frac{Q_p}{Q_n} &= \frac{\gamma^k R}{\gamma^{k+1} R} = \frac{1}{\gamma} \\ \Rightarrow Q_n &= \gamma \times Q_p \end{aligned} \tag{8}$$

Since $d_{nG} > d_{pG}$, $Q_n < Q_p$. Therefore, $Q_\gamma \times Q_p$ has the largest possible value for $0 < \gamma < 1$. Furthermore, as $L_p = 1$ and S_n is the nearest state to S_p with respect to the given distance metric, therefore, L_n is set to 1.

Property 4: If $L_n = 1$ and $d_{nG} > d_{pG}$, then $Q_p = Q_n/\gamma$ and set $L_p = 1$.

Proof: Since $d_{nG} > d_{pG}$, Q_n can be evaluated from Q_p by 8. Thus,

$$Q_n = \gamma \times Q_p \Rightarrow Q_p = Q_n/\gamma \tag{9}$$

Since $L_n = 1$ and $d_{nG} > d_{pG}$, $\therefore Q_n < Q_p$, and thus, $Q_p = Q_n/\gamma$ for $0 < \gamma < 1$ is the largest possible value of Q_p, so Q_p should not be updated further. Therefore, $L_p = 1$ is set.

4.3.2 Improved Q-learning Algorithm

The CQL employs a Q-table to store the $Q(S, a)$ for $S = S_1, S_2, ..., S_n$ and $a = a_1, a_2, ..., a_m$. Thus, it requires an array of $(n \times m)$ size. However, in the IQL, it only requires to store the Q-value at a state S for the best action. Thus, for n states, it needs to store n Q-values. In addition, aside from the Q-storage, it requires n Boolean Lock variables, denoted by L_i for state $S_i, i = 1, 2, ..., n$, depicting the current status of the state. If the Lock variables at a state is 1, then the Q-value at that state need not be updated further.

Once the UAV reaches the goal, the first repeat-until loop exits, and the Q-table updating is initiated. The process of Q-table updating is continued until all the states are locked. The pseudocode of the IQL is given hereafter.

Pseudocode for IQL

1. Initialization

For all $S_i, i = 1, 2, ..., n$, except $S_i = S_G$, set $L_i = 0$;$Q_i = 0$;
L_G(for goal S_G)=1;
Q_G(for goal S_G)=100;
Assign γ in (0,1) and initial state is S_p;

2. Cycle

Repeat
Select a_i from $A = a_1, a_2, ..., a_m$ and execute it;
Until $S_P = S_G$;

3. Update Q-table

Repeat
a) Select a_i from $A = a_1, a_2, ..., a_m$;
b) Determine d_{nG} and d_{pG};
If $(d_{nG} < d_{pG})$
Then if $(L_n = 1)$
Then if$(L_p = 0)$
Then$Q_p = \gamma \times Q_n$;$L_p = 1$;
Else if $(L_p = 1)$
Then $Q_n = Q_p/\gamma$;$L_n = 1$;
Else if$(L_p = 1)$
Then if$(L_n = 0)$
Then $Q_n = \gamma \times Q_p$;$L_n = 1$;
Else if$(L_n = 1)$
Then $Q_p = Q_n/\gamma$;$L_p = 1$;
Until $L_i = 1$ for all i without obstacle;

4.4 Geometry Reinforcement Learning Algorithm

The Geometry Reinforcement Learning (GRL) algorithm exploits a specific weight matrix, which contains geometric information and is simple and efficient for path planning. In most of the RL path planning algorithm, the flight direction of a UAV is limited to eight directions, as the next state is confined to the eight neighboring states. So it affects the feasibility of the UAV flight path, and also the complexity of the trajectory . As shown in Fig. 4, it is more efficient to go by path 1: $a \to e$ (the GRL), rather than travel by path 2: $a \to b \to c \to d \to e$ (other RL methods).

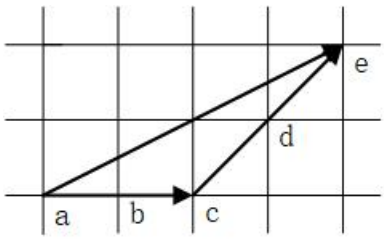


Figure 4: Illustration of path by GRL and other RL methods

The main idea of GRL is to add a weight term in the evaluation function. Specifically, the weight for the path between any two states in the environment is computed by the geometric

distance nad the integral risk measurement:

$$W_{s,s'} = \frac{l}{d_{s,s'} + d_{s',T}} + \frac{k}{R_{s,s'} + R_{s',T}} \tag{10}$$

where s, s', T are the current state, next state and target state, respectively. l and k are the parameters to adjust the weight between geometric distance and risk. $d_{s,s'}$ stands for the geometry distance (the path length) from s to s' and $R_{s,s'}$ represents the total risk measurement of the path from s to s', which is defined as follows.

$$R_{s,s'} = \int_{s}^{s'} F(x)ds \tag{11}$$

where $F(x)$ is the risk function, it can be different for various problems.

Let $S_1, S_2, ..., S_n$ be n possible stats in the environment, and $a_1, a_2, ..., a_m$ be the set of m possible actions for each state. It should be noted that in GRL, the actions at each state can be designed. For example, if the UAV could explore a 5×5 area, the total number of actions m is 24, while the UAV could explore a 7×7 area, the m is 48. An agent at state S_i selects an action a_j will receives a reward $G(S_i, a_j)$. The G-value at state S_i due to action a_j is given below:

$$G(S_i, a_j) = Q(S_i, a_j) + W(S_i, S_i') \tag{12}$$

where $Q(S_i, a_j)$ denotes the Q-value which is defined in (3) and $W(S_i, S_i')$ refers to weight which is described in (10). S_i' is the next state when the agent takes action a_j at state S_i.

Pseudocode for GRL

1. Initialization
Calculate the weight matrix W and set $Q(S, a) = 0$;
Update
Repeat
Observe the current state S;
Select $a \in [a_1, a_2, ..., a_n]$ with the maximum G-value $G(S, a)$ and execute it.
Observe the new state S' and update the Q-value by (3);
Update the G-value by (12);
Set the new state to the current state $S \leftarrow S'$;
Until the end condition is satisfied.

4.5 Ant-Q Algorithm

Ant-Q learning method is an extension of Ant System (AS) and reinforcement learning algorithm. Let $AQ(r, s)$ (Ant-Q-value), be a positive real value associated to the $edge(r, s)$. In other words, it is the Ant-Q-value from node r to node s. It is the Ant-Q counterpart of Q-learning Q-values, and is intended to indicate how useful it is to make move to node s when in node r. It should be noted that for a symmetric environment, the $AQ(r, s) = AQ(s, r)$. While in an asymmetric environment, the value of $AQ(r, s)$ can be different from $AQ(s, r)$.

Let $HE(r, s)$ be a heuristic value associated to $edge(r, s)$ which allows an heuristic evaluation of which moves are better. The definition of this heuristic value might be different in various problems. For example, in a search problem which considers the travel distance as the cost function. The $HE(r, s)$ could be the inverse of the distance.

Let k be an agent whose task is to make a tour aiming at the mission objective. In most search cases, the objective is to visit all the nodes and return to the starting one. Associated to k there is the list $J_k(r)$ of nodes still to be visited, where r is the current node.

An agent k situated in node r moves to node s using the following rule, called action choice rule or state transition rule:

$$s = \begin{cases} arg \underset{s \in J_k(r)}{\text{Max}} [AQ(r,s)]^\alpha \times [HE(r,s)]^\beta & \text{if } q \leq q_0 \\ S & \text{otherwise} \end{cases} \tag{13}$$

where α and β are parameters which weigh the relative importance of the learned AQ-values and the heuristic values, q is a value chosen randomly with uniform probability in $[0,1]$, q_0 $(0 \leq q_0 \leq 1)$ is a parameter such that the higher the q_0 is, the smaller the probability to make a random variable selected according to a probability distribution given by a function of the $AQ(r,u)$ and $HE(r,u)$, with $u \in J_k(r)$.

Pseudocode for Ant-Q

1. Initialization
For each pair (r,s), $AQ(r,s) = AQ_0$
For $k = 1$ to m
Let r_{k1} be the starting node for agent k
$J_k(r_{k1}) = [1, 2, ..., n] - r_{k1}$
$r_k = r_{k1}$
End-for
2. Cycle
For $i = 1$ to n
If $i \neq n$
For $k = 1$ to m
Choose the next node s_k according to formula 13
If $i \neq n-1$ Then $J_k(s_k) = J_k(r_k) - s_k$
If $i = n-1$ Then $J_k(s_k) = J_k(r_k) - s_k + r_{k1}$
$Tour_k(i) = (r_k, s_k)$
End-for
Else
For $k = 1$ to m
$s_k = r_{k1}$
$Tour_k(i) = (r_k, s_k)$
End-for
End-if
For $k = 1$ to m
$AQ(r_k, s_k) = (1-\alpha)AQ(r_k, s_k) + \alpha\gamma \underset{z \in J_k(s_k)}{\text{Max}} AQ(s_k, z)$
$r_k = s_k$
End-for
End-for
3. Update
For $k = 1$ to m
Compute L_k

End-for
For each $edge(r, s)$, compute the delayed reinforcement $\triangle AQ(r, s)$
Update AQ-values by applying formula **??**
4. End
If end condition is satisfied
The best solution is the shortest path of L_k
Else
Return to **Step 2**
End-if

4.6 LOS Path Following Control

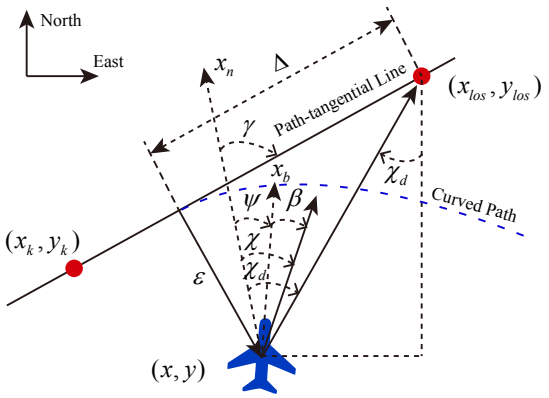


Figure 5: LOS guidance geometry of UAV

Fig. 5 shows the geometry of the LOS guidance problem and main variables are involved in it. A two-dimensional continuous path P is considered in this study, where the path is assumed to go through a set of successive waypoints (x_k, y_k) for $k = 1, ..., N$. For an arbitrary waypoint on the path, the path-tangential angle γ_P is calculated by

$$\gamma_P = atan2(y'_p(P), x'_p(P)) \tag{14}$$

where $x'_p(P) = \partial x_p / \partial P$ and $y'_p(P) = \partial y_p / \partial P$. γ_P is constant when the desired path is a straight line between two waypoints. According to (14), γ_P varies while the desired path is a parametrized curve. The function $atan2(y/x)$ is a generalization of the function $atan(y/x)$, which considers the signs of both x and y to determine the quadrant of the result and distinguish between diametrically opposite directions.

While the UAV locates at (x, y), the cross-track error, which is defined as the orthogonal distance to the path tangential reference frame, can be computed by:

$$\varepsilon = -(x - x_p(P))sin(\gamma_P) + (y - y_p(P))cos(\gamma_P) \tag{15}$$

And the path normal line (which is perpendicular to the path and passes through the UAV) is obtained by:

$$y - y_p(P) = -(x - x_p(P))/tan(\gamma_P) \tag{16}$$

According to the definition in [29], the LOS is the line that starts at the reference point and passes through the objective of the guidance. On the other hand, in flight guidance applications, the LOS vector starts at the aircraft and passes through a point (x_{los}, y_{los}), which is located on the path-tangential line at a lookahead distance $\Delta > 0$ ahead of the direct projection of the aircraft's position (x, y) on to the path. In this paper, the guidance law is given by

$$\chi_d = \gamma_P + arctan(-\varepsilon/\Delta) \tag{17}$$

where χ_d is the desired course angle of the UAV

$$\chi_d = \psi_d + \beta \tag{18}$$

where β is the sideslip angle of the UAV. In other words, the LOS guidance ensures that the aircraft's velocity is directed toward the moving point (x_{los}, y_{los}) until the aircraft converges to the path.

By differentiating (15) with respect to time,

$$\begin{aligned} \dot{\varepsilon} =& usin(\psi - \gamma) + vcos(\psi - \gamma) \\ =& U(cos\beta sin(\psi - \gamma) + sin\beta cos(\psi - \gamma)) \\ =& Usin(\psi - \gamma + \beta) \end{aligned} \tag{19}$$

If we assume that the desired heading is perfectly tracked at all times and choose the desired heading angle as:

$$\psi_d = \gamma + arctan(\frac{-\varepsilon}{\Delta}) - \beta \tag{20}$$

The derivative of the cross-track error becomes:

$$\dot{\varepsilon} = U\frac{-\varepsilon}{\sqrt{\Delta^2 + \varepsilon^2}} \tag{21}$$

The lookahead distance Δ is further designed to be time-varying in this proposal, which is constructed as below:

$$\Delta(\varepsilon) = (\Delta_{max} - \Delta_{min})e^{-K\varepsilon^2} + \Delta_{min} \tag{22}$$

where Δ_{max} and Δ_{min} are the maximum and minimum allowed values for Δ, respectively.

In this proposal, a modified integral LOS algorithm based on the system kinematics is proposed. The integral LOS guidance system is:

$$\psi_d = \gamma - arctan(\frac{1}{\Delta}(\varepsilon + \kappa y_{int})) \tag{23}$$

$$\dot{y}_{int} = \frac{U\varepsilon}{\sqrt{\Delta^2 + (\varepsilon + \kappa y_{int})^2}} \tag{24}$$

The derivative of cross-track error could be rewritten as follows:

$$\begin{aligned} \dot{\varepsilon} =& Usin(\psi - \gamma + \beta) \\ =& Usin(-arctan(\frac{\varepsilon + \kappa y_{int}}{\Delta})) \end{aligned} \tag{25}$$

According to the following equation

$$sin(arctan(x)) = \frac{x}{\sqrt{x^2+1}} \tag{26}$$

We can obtain

$$\dot{\varepsilon} = -U\frac{\varepsilon + \kappa y_{int}}{\sqrt{\Delta^2 + (\varepsilon + \kappa y_{int})^2}} \tag{27}$$

The overall structure of the proposed method is shown in Fig. 6.

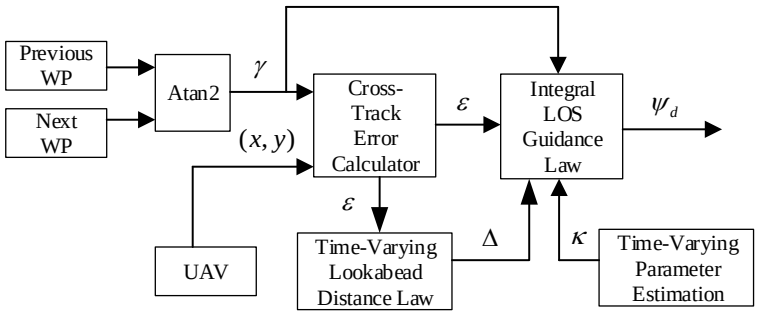


Figure 6: Structure of the guidance system

5 Timeline

The following Table 2 outlines the progress for accomplishing the objectives of the PhD study.

Table 2: Timelines for PhD study

Main Tasks	**2016-2017**			**2017-2018**			**2018-2019**			**2019-2020**		
	S	**F**	**W**	**S**	**F**	**W**	**S**	**F**	**W**	**S**	**F**	**W**
Course Study		>	>								-	-
Literature Review	>	>	>	>	>	>	>	>	>		-	-
Comprehensive Exam									>		-	-
CQL path planning				>	>	>	>	>	>	>	-	-
IQL path planning						>	>	>	>	>	-	-
GRL path planning							>	>	>	>	-	-
Ant-Q path planning								>	>	>	-	-
Path following control								>	>	>	-	-
Dynamic RL path planning								>	>	>	-	-
3-D RL path planning									>	>	-	-
Cooperative RL path planning										>	-	-
Experimental Setup and Validation									>	>	-	-
Publications				>	>	>	>	>	>	>	-	-
Thesis Writing									>	>	-	-

Note: **S**: summer; **F**: fall; **W**: winter semesters; RL: reinforcement learning; CQL: classical Q-learning; IQL: improved Q-learning; GRL: geometry reinforcement learning; Ant-Q: ant colony Q-learning; >: available; -: not available.

6 Anticipated Significance of the Work and Future Work

6.1 Conclusion

Although tremendous efforts on UAV guidance and navigation system have been dedicated to make UAVs more autonomous, there still exist significant challenges in the development of the system. Many key technical issues must be solved to bring the autonomy up to a state required for more sophisticated and hazardous applications. In the near future, the development of UAV path planning in the dynamic environment, as well as more accurate and reliable path following controller, remains an open issue, and there are numerous ongoing work on these topics.

The results of the research proposed here will be disseminated in the form of conferences, journals and technical reports. Some preliminary work of the proposed research in particular IQL and GRL path planning methods, as well as LOS path following control, at individual UAV level are completed, and partial results are also published. Up to now, all my previous research are summarized in one journal paper [24] and several conference papers [22, 23, 32].

6.2 Future Work

1. *Dynamic environment*: The design of real-time RL path planning methods in a dynamic environment brings several challenge issues. For example, the path planning algorithm needs to learn the environment and make decision at the same time, which requires a extreme fast convergence speed. Moreover, how to update Q-table without reaching the target point is also an interesting and challenging issue. Therefore, designing a path planning algorithm in dynamic environment will be investigated in my next stage research;

2. *3-D path planning*: The biggest challenge of 3-D path planning using RL methods is the dimension of states. Unlike path planning problem in 2-D environment, the states in 3-D environment is growing exponentially which leads to a enormous Q-table. How to deal with huge possibilities and data needs more efforts of researchers. Thus, the 3-D path planning algorithm will be further studied;

3. *Cooperative path planning*: The design of multiple UAVs path planning algorithm is totally different compared to the algorithm for a single one. Comprehensively considering the team coordination and individual requirement, how to cooperatively update the Q-table is still an open problem. The effort to solve this problem will also be investigated.

4. *Reliable path following controller*: Although LOS control strategy has a good performance, we still need a better control strategy while the path is sharply curved. Disturbance observer (DO) based controller has a fast and accurate response dealing with external disturbance. So a DO-based path following control strategy will be studied and compared to LOS control in the future work.

5. *Experimental validation*: In order to verify the effectiveness of the proposed methodologies in the practice, a Quadrotor platform is to be developed and several indoor or outdoor experiments will also be carried out.

References

[1] E. Borhaug, A. Pavlov, and K. Y. Pettersen. Integral LOS control for path following of underactuated marine surface vessels in the presence of constant ocean currents. In *47th IEEE Conference on Decision and Control*, pages 4984–4991, 2008.

[2] S. A. Bortoff. Path planning for UAVs. In *American Control Conference*, volume 1, pages 364–368, Chicago, USA, 2000.

[3] M. Breivik. Topics in guided motion control of marine vehicles. 2010.

[4] M. Breivik and T. I. Fossen. Guidance laws for autonomous underwater vehicles. In *Underwater vehicles*. 2009.

[5] P. Chandler, S. Rasmussen, and M. Pachter. UAV cooperative path planning. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 4370, Denver, USA, 2000.

[6] P. R. Chandler, M. Pachter, D. Swaroop, J. M. Fowler, J. K Howlett, S. Rasmussen, C. Schumacher, and K. Nygard. Complexity in UAV cooperative control. In *American Control Conference*, volume 3, pages 1831–1836, Anchorage, USA, 2002.

[7] P. K. Das, H. S. Behera, and B. K. Panigrahi. Intelligent-based multi-robot path planning inspired by improved classical Q-learning and improved particle swarm optimization with perturbed velocity. *Engineering Science and Technology*, 19(1):651–669, 2016.

[8] G. Dudek and M. Jenkin. *Computational principles of mobile robotics*. Cambridge university press, 2010.

[9] M. Duguleana and G. Mogan. Neural networks based reinforcement learning for mobile robots obstacle avoidance. *Expert Systems with Applications*, 62:104–115, 2016.

[10] L. Fei and G. Zeng. *Study of genetic algorithm with reinforcement learning to solve the TSP*, volume 36. 2009.

[11] T. I. Fossen, M. Breivik, and R. Skjetne. Line-of-sight path following of underactuated marine craft. In *6th IFAC Conference on Manoeuvring and Control of Marine Craft*, pages 211–216, Girona, Spain, 2003.

[12] Luca M. Gambardella and M. Dorigo. Ant-Q: A reinforcement learning approach to the traveling salesman problem. *Machine Learning Proceedings*, 170(3):252–260, 1995.

[13] M. Gerke and H. Hoyer. Planning of optimal paths for autonomous agents moving in inhomogeneous environments. In *8th International Conference on Advanced Robotics*, pages 347–352, Monterey, USA, 1997.

[14] S. Griffiths. Vector field approach for curved path following for miniature aerial vehicles. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 6467, 2006.

[15] M. A. K. Jaradat, M. Al-Rousan, and L. Quadan. Reinforcement based mobile robot navigation in dynamic environment. *Robotics and Computer-Integrated Manufacturing*, 27(1):135–149, 2011.

[16] A. Konar, I. G. Chakraborty, S. J. Singh, L. C Jain, and A. K Nagar. A deterministic improved Q-learning for path planning of a mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 43(5):1141–1153, 2013.

[17] A. M. Lekkas and T. I. Fossen. Integral LOS path following for curved paths based on a monotone cubic hermite spline parametrization. *IEEE Transactions on Control Systems Technology*, 22(6):2287–2301, 2014.

[18] S. D. Li, X. Xu, and L. Zuo. Dynamic path planning of a mobile robot with improved Q-learning algorithm. In *International conference on information and automation*, pages 409–414, Lijiang, China, 2015.

[19] X. Ma, X. G. Ya, L. D. Sun, and Y. Li. State-chain sequential feedback reinforcement learning for path planning of autonomous mobile robots. *Journal of Zhejiang University Science*, 14(3):167–178, 2013.

[20] Z. Ni, H. B. He, J. Y. Wen, and X. Xu. Goal representation heuristic dynamic programming on maze navigation. *IEEE transactions on neural networks and learning systems*, 24(12):2038–2050, 2013.

[21] S. Park, J. Deyst, and J. How. A new nonlinear guidance logic for trajectory tracking. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 4900, 2004.

[22] Y. H. Qu, Y. T. Zhang, and Y. M. Zhang. Optimal flight path planning for uavs in 3-D threat environment. In *International Conference on Unmanned Aircraft Systems*, pages 149–155, Orlando, USA, 2014.

[23] Y. H. Qu, Y. T. Zhang, and Y. M. Zhang. A UAV solution of regional surveillance based on pheromones and artificial potential field theory. In *International Conference on Unmanned Aircraft Systems*, pages 380–385, Denver, USA, 2015.

[24] Y. H. Qu, Y. T. Zhang, and Y. M. Zhang. A global path planning algorithm for fixed-wing UAVs. *Journal of Intelligent & Robotic Systems*, pages 1–17, 2018.

[25] C. A. Rabbath, E. Gagnon, and M. Lauzon. On the cooperative control of multiple unmanned aerial vehicles. *IEEE Canadian Review*, 46(1):15–19, 2004.

[26] R. Rysdyk. Unmanned aerial vehicle path following for target observation in wind. *Journal of Guidance, Control, and Dynamics*, 29(5):1092–1100, 2006.

[27] P. B. Sujit, S. Saripalli, and J. B. Sousa. Unmanned aerial vehicle path following: A survey and analysis of algorithms for fixed-wing unmanned aerial vehicless. *IEEE Control Systems Magazine*, 34(1):42–59, 2014.

[28] H. R. Xiao, L. Liao, and F. Y. Zhou. Mobile robot path planning based on Q-ANN. In *International Conference on Automation and Logistics*, pages 2650–2654, Jinan, China, 2007.

[29] R. Yanushevsky. *Guidance of unmanned aerial vehicles*. CRC press, 2011.

[30] X. Yu, Z. X. Liu, and Y. M. Zhang. Fault-tolerant flight control design with finite-time adaptation under actuator stuck failures. *IEEE Transactions on Control Systems Technology*, 25(4):1431–1440, 2017.

[31] B. C. Zhang, Z. L. Mao, W. Q. Liu, and J. Z. Liu. Geometric reinforcement learning for path planning of UAVs. *Journal of Intelligent & Robotic Systems*, 77(2):391–409, 2015.

[32] Y. T. Zhang, Y. M. Zhang, Z. X. Liu, Z. Q. Yu, and Y. H. Qu. Line-of-sight path following control on UAV with sideslip estimation and compensation. In *Chinese Control Conference*, pages 4711–4716, Wuhan, China, 2018.

[33] C. W. Zheng, L. Li, F. J. Xu, F. C. Sun, and M. Y. Ding. Evolutionary route planner for unmanned air vehicles. *IEEE Transactions on robotics*, 21(4):609–620, 2005.

7 Appendix: Progress to Date

With respect to the presented timeline in section 5, this section provides a brief review of the progress to date of the PhD thesis work. The aforementioned CQL, IQL, GRL and Ant-Q algorithms are validated in different environment as well as the LOS path following control. Firstly, the grid-based path planning problems are investigated using CQL, IQL and GRL, After that, Ant-Q algorithm is utilized to solve a node-based path planning problem. Finally, the LOS guidance law is validated.

7.1 Grid-based Path Planning Problem

There are two main directions in the grid-based path planning problem. One is to design a collision-free path in a multiple obstacle environment and the other is threat avoidance problem, which considers not only the path length but also the risk UAVs may take during execution.

7.1.1 Collision-free

In order to test the proposed RL path planning algorithm, a multi-obstacle environment is designed in Fig. 7. The objective is to design a shortest collision-free path from start location to target point.

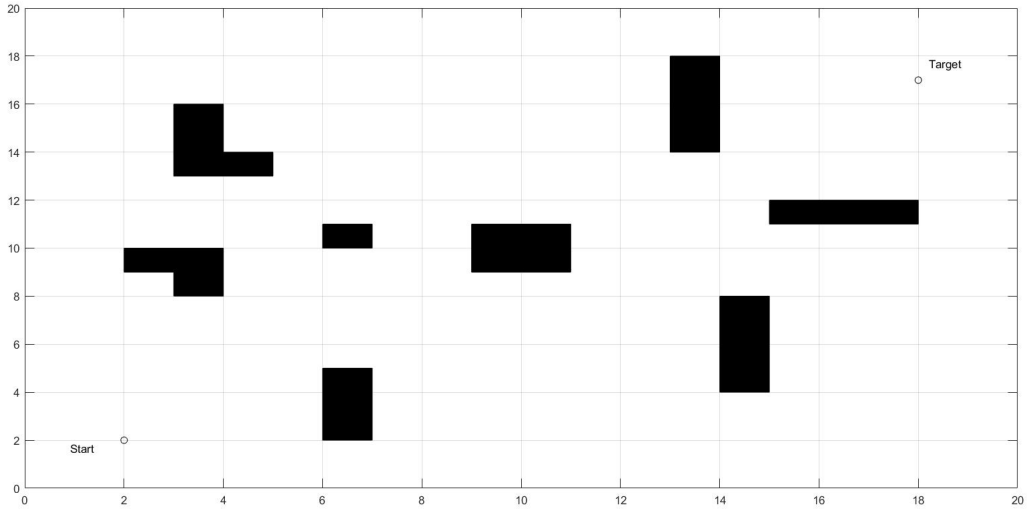
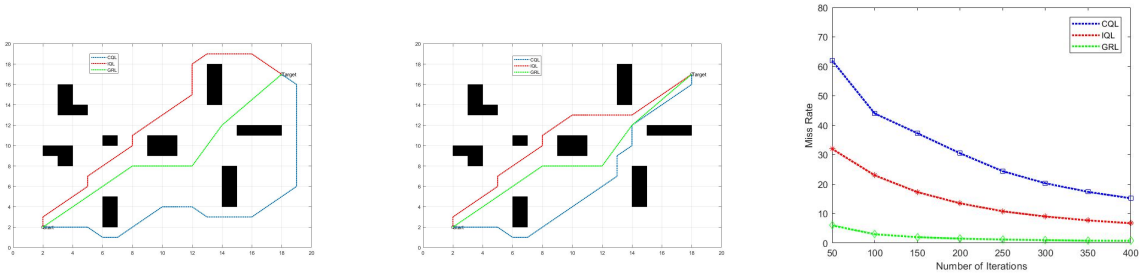


Figure 7: A grid-based environment

Fig. 8 shows the planned path using different RL path planning algorithms. The left figure represents the best solution of 3 RL algorithms within 100 iteration. While the right figure shows the best paths of 200 iterations. It is obviously that the GRL converges fast and has the shortest path. And the IQL also has a good convergence speed and a relatively good path while the CQL is trapped in the local optimal at bottom of the map.

As shown in the Fig. 8 below, the CQL has a pretty high miss rate, which means the attempt does not reach the target point, that's the reason why it has a poor convergence speed. The GRL almost has zero miss rate because of the geometry information added in the evaluation function, which leads the path towards target.

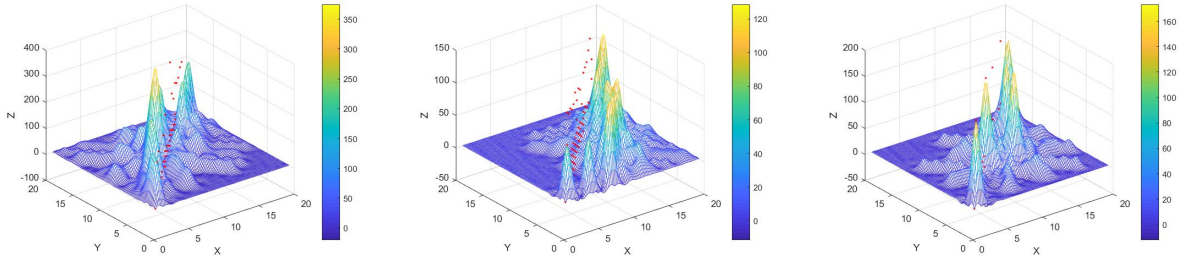
Fig. 9 shows the heatmap of the states, which refers to how many times the state has been explored during the learning process. The IQL has the ability to lock a state and needs no more exploration at state. As a result, it needs less exploration compared to CQL. Due to the geometry information of



(a) Best solution at iteration 100 (b) Best solution at iteration 200 (c) The misstate of CQL, IQL and GRL

Figure 8: Path and missrate comparison of CQL, IQL and GRL

GRL, the GRL explore less states while learning. These are the reasons why IQL and GRL have a fast convergence speed.



(a) Heatmap of CQL (b) Heatmap of IQL (c) Heatmap of GRL

Figure 9: Heatmap of CQL, IQL and GRL

7.1.2 Threat avoidance

The threat avoidance is another important directions in path planning problem. As shown in Fig. 10, a multiple obstacle environment is designed to test the proposed algorithms.

A. Comparison between IQL and GRL

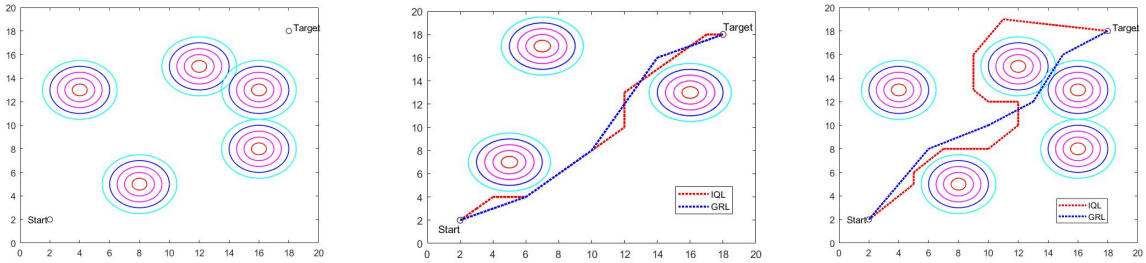
Unlike the collision-free problem, the threat avoidance problem needs to consider both path length and path risk. Due to its complicated calculation, only the IQL and GRL will be tested.

As shown in Fig. 10, if the environment has few threat source and they are evenly distributed in the map. The IQL and GRL both has a good performance.

As shown in Fig. 10 if the number of threat source increases, the IQL is easily to be trapped in local optimal while the GRL still has a good performance.

B. Trade-off between travel length and threat

Equation 10 gives the designer a flexible way to trade-off the path length and risk. As shown in Fig. 10, if the path length is considered to be the major cost, the GRL could travel through the high risk area. But the IQL is weak at this.



(a) Threat environment (b) Best solution of IQL and GRL (c) Best solution of IQL and GRL

Figure 10: Threat environment and solution comparison

7.2 Node-based Path Planning Problem

7.3 LOS Path Following Control

In the simulation, the fixed wing UAV is assumed to maintain straight level flight. As a result, only the lateral motion is necessary. The linearized model is given by [30]

A PID controller is applied in order to accomplish the yaw control of the inner loop. It gives the control signal $u = [\delta_a \ \delta_r]^T$ based on ψ_d and ψ.

In order to validate the proposed path following approach, several simulations have been carried out.

Case 1: The goal of the first case is to demonstrate the effectiveness of the proposed LOS path following method.

Case 2: This case is to validate the effectiveness of the proposed integral LOS path following method with a fixed compensation parameter κ.

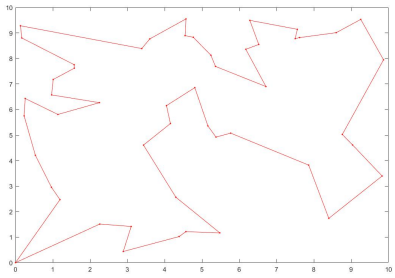
To generate a smooth and practical path connecting desired waypoints, these waypoints are are interpolated using the cubic Hermite spline algorithm [17].

The UAV starts at the position $(0, 0)$ with 0 deg initial heading angle, traveling with a constant speed of $30m/s$. Definition of the parameters are $\Delta_{max} = 60$, $\Delta_{min} = 10$, K for the integral LOS guidance equals to 1, and the fixed compensation parameter of **Case 2** $\kappa = 1$.

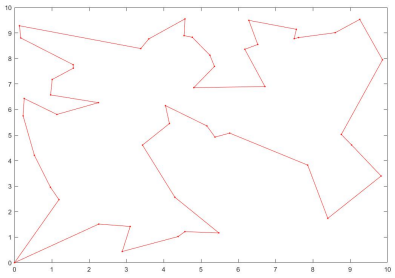
The trajectories of **Case 1** and **Case 2**, as well as the desired one, are shown in Fig. 12.

And Fig. 12 displays the cross-track error in two cases.

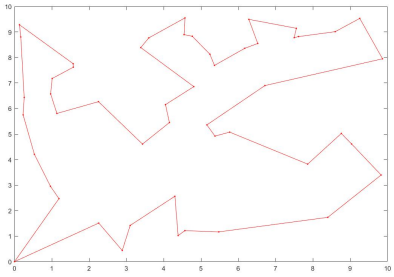
It is obviously to tell **Case 2** has less cross-track error and better performance than **Case 1**.



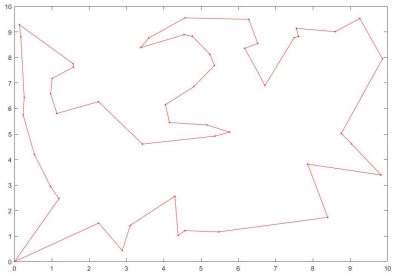
(a) Ant-Q Distance 59.8145 Iteration 645



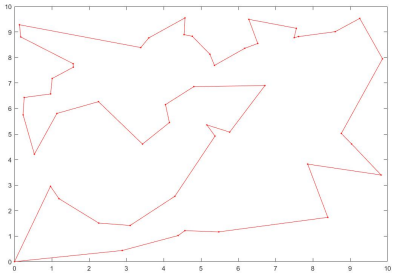
(b) GA Distance 59.9435 Iterations 9153



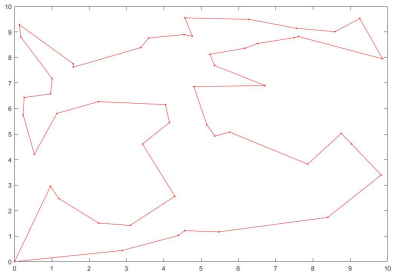
(c) Ant-Q Distance 60.7182 Iterations 418



(d) GA Distance 60.8783 Iterations 6196

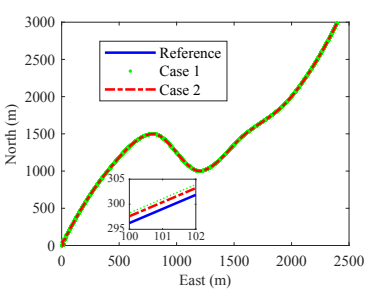


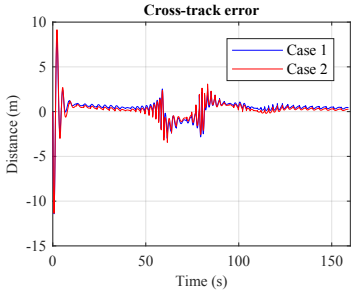
(e) Ant-Q Distance 62.7263 Iterations 236

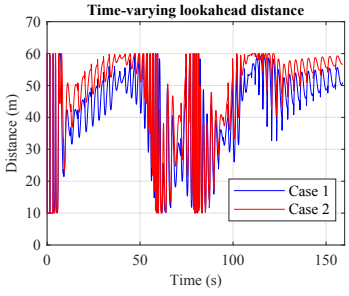


(f) GA Distance 63.4603 Iterations 2230

Figure 11: Simulation results comparison between Ant-Q and GA







(a) Performance comparison of path following (b) Cross-track error comparison (c) Lookahead distance comparison

Figure 12: Performance comparison