EasyChair Preprint

№ 5372

# Hybrid Obstacle Avoidance and Simulation of Carrier-Based Aircraft on the Deck of an Aircraft Carrier

Xue Junxiao, Kong Xiangyan, Dong Bowei, Guo Yibo and Xu Mingliang



April 25, 2021

# 航母甲板上舰载机的混合避障和仿真

薛均晓[1)], 孔祥燕[1)], 董博威[2)], 郭毅博[2)], 徐明亮[2)*]

[2)] (郑州大学软件学院 郑州 450002)
[1)] (郑州大学信息工程学院 郑州 450001)
(iexumingliang@zzu.edu.cn )

摘 要: 航母甲板上舰载机的混合避障是路径规划中一个重要的问题. 现有方法在处理动态障碍物时存在效果不稳定且决策空间离散的问题. 为了解决这些问题, 我们提出了一种结合模型预测控制(MPC)与深度确定性策略梯度(DDPG)的新算法. 首先, 用 MPC 算法预测动态障碍物的短期轨迹. 其次, 根据 MPC 预测的动态障碍物的短期轨迹, DDPG 为机器人提供在连续空间里学习和决策行为的能力. 最后, 人工势场的概念被引入来设置奖励函数, 以提高收敛速度和准确率. 我们使用 Unity 3D 在高动态复杂场景中进行仿真实验, 结果表明, 我们的方法与其他方法相比, 在精度上提高了 7%-30%. 且与 DQN(Deep Q Network)相比, 在路径长度和转弯角度上分别减少了 100 个单位和 400-450 度.

关键词: 路径规划; 避障; MPC; 人工势场
中图法分类号: TP391.41 **DOI:** 10.3724/SP.J.1089.202*.论文编号

# Hybrid obstacle avoidance and simulation of carrier-based aircraft on the deck of an aircraft carrier

Xue Junxiao[1)], Kong Xiangyan [1)*], Dong Bowei [2)], and Xu Mingliang [2)*]

[1)] ( *School of Software Engineering, Zhengzhou University, Zhengzhou* 450002)
[2)] ( *School of Information Technology, Zhengzhou University, Zhengzhou* 450001)

**Abstract:** The problem of mixed static and dynamic obstacle avoidance is essential for path planning in highly dynamic environment. Existing methods have the problem of unstable effects when dealing with dynamic obstacles, and many methods have discrete decision spaces. To address this problem, we propose a new algorithm combining Model Predictive Control (MPC) with Deep Deterministic Policy Gradient (DDPG). Firstly, we apply the MPC algorithm to predict the trajectory of dynamic obstacles. Secondly, the DDPG with continuous action space is designed to provide learning and autonomous decision-making capability for robots. Finally, we introduce the idea of the Artificial Potential Field to set the reward function to improve convergence speed and accuracy. We employ Unity 3D to perform simulation experiments in highly uncertain environment. The results show that our method

收稿日期: 20**-**-**; 修回日期: 20**-**-**. 基金项目: 国家自然科学基金(62036010、61822701); 河南省高校科技创新人才支持计划(18HASTIT020); 薛均晓(1982—), 男, 博士, 副教授 硕士生导师, CCF 专业会员, 主要研究方向为人工智能、虚拟现实; 孔祥燕(1996—), 女, 硕士生, 主要研究方向为强化学习、路径规划; 郭毅博(1987—), 男, 博士, 讲师, 硕士生导师, 主要研究方向为机器学习; 鲁爱国(1979—), 男, 本科, 研究员, 主要研究方向为人工智能; 李鉴(1969—), 男, 本科, 研究员, 主要研究方向为机器视觉; 万曦(1981—), 女, 本科, 高级工程师, 主要研究方向为机器学习; 徐明亮(1981—), 男, 博士, 教授, 博士生导师, 主要研究方向为计算机图形学、人工智能.

has made great improvement on accuracy by 7%-30% compared with the other methods, and on the length of the path and turning angle by reducing 100 units and 400-450 degrees compared with DQN (Deep Q Network), respectively.

**Key words:** path planning; obstacle avoidance; MPC; Artificial Potential Field

路径规划指在障碍物与目标物之间建立一条从起始位置到终点位置无碰撞、最优路径或次优路径[1][2]. 在航母甲板上, 障碍物数量多、密度大、种类复杂、随机性强, 环境狭窄、危险系数高, 靠人工决策来实现路径规划和避障会消耗大量的人力、物力, 规划效率低, 且一旦失误, 将带来不可估量的损失和伤害. 因此, 在这种复杂环境中, 自主路径规划和避障的能力成为一种迫切的需求.

甲板路径规划和一般的路径规划问题不同, 后者只强调规划时间和效率, 不过多考虑高密度的情况, 而前者须要将两者都重视起来. 由于舰面上同时存在多个机务保障点会出现多架舰载机同时出现调运的情况. 因此, 不仅要考虑舰载机与船体和静止的舰载机之间的碰撞, 还要考虑动态舰载机之间的碰撞. 在航母甲板上, 作业区域存在部分重合的情况, 且没有固定的作业通道. 因此增加了甲板上路径规划的难度.

本文的结构如下: 第一部分相关工作, 介绍了路径规划的相关算法. 包括传统算法, 智能仿生算法, 强化学习算法, 并引入了我们的算法及贡献点. 第二部分本文方法包括航母甲板的建模和算法框架. 我们分别从状态空间、动作空间、奖励函数对场景进行简单的建模. 我们将算法框架分为轨迹预测、动作选择、更新三大部分. 第三部分仿真分析与验证使用 Unity3D 来对航母甲板进行建模, 且将我们的算法与 DDPG, DQN 和 A2C 进行比较, 从准确率, 路径长度, 奖励值和路径平滑度验证我们方法的有效性. 第四部分给出了总结.

## 1 相关工作

路径规划算法主要分为传统算法、智能优化算法和基于强化学习的算法, 传统算法是最基本最成熟的算法, 这些算法原理简单易实现, 并得到广泛应用. 智能优化算法通过模拟自然界生物的行为规律实现优化的目的. 强化学习(RL)赋予智能体足够的智能不断与环境相互作用来获取未知环境的知识, 然后实时的做出决策。

### 1.1 传统路径规划算法

早期传统路径规划算法主要是基于图的路径规划[3] [4][5] [6] [7] [8]: 例如 C-space [3], Dijkstra[7], A*[8], 等, 但基于图的方法无法解决动态障碍的问题. 人工势场[9], 快速探索随机树[10]和 D * [11]可以用于解决此类问题. 但人工势场存在局部极小值的问题, 快速探索随机树无法应对连续高度动态的环境, 因此规划的路径通常不是最优的, 而且不平滑, D* 算法无法处理距离远的最短路径上发生的变化. 虽然传统路径规划有算法参数少、易实现等优点, 但此类算法无法处理高动态、高密集环境下的避障, 且实时性差.

### 1.2 智能仿生算法

智能仿生算法中, 大部分是基于生物群体一些重要行为的特点的启发. 如蚁群算法[12]、粒子群算法[13-14]、遗传算法等. 蚁群算法一种基于蚂蚁觅食行为的智能仿生算法. 粒子群算法是一种模拟鸟群飞行觅食行为的智能仿生优化算法. 遗传算法是以生物进化论和遗传变异论为基础, 通过模仿生物进化机制达到全局搜索和优化的目的. 虽然智能仿生算法简单易懂, 鲁棒性强且具有较高的自组织能力. 但此类算法容易陷入局部最优解, 且收敛速度较慢.

### 1.3 强化学习算法

强化学习(RL)赋予智能体充足的智能不断与环境相互作用来获取未知环境的知识. Q-learning 可以被用来解决无人机的自主导航问题[15][16]. Sichkar[17]将 Q-learning[18]用于含有动静态障碍物

的路径规划中．虽然 Q-learning 适用于以上环境，但在复杂且高动态的环境中，由于维数诅咒，Q-learning 很难收敛．刘[19]利用与贪婪策略相结合的 Deep Q Network(DQN)[20]算法来支持大规模复杂环境中的智能体轨迹规划和优化．通过使用基于值的双 DQN(DDQN)[21]，曾[22]引入了一种用于无人机的高精度导航技术．这些算法只能处理离散动作(将环境建模为具有有限动作空间的网格世界)，当处理真实环境时，处理效率较低．

为解决以上问题，基于策略梯度[23]，本文提出了一种结合 MPC[24](模型预测控制)和 DDPG[25]的新算法．本文的主要贡献如下：

(1)提出了一种基于 DDPG 的避障方法，该方法使舰载机在训练过程中反复试验，以学习一种无冲突地到达目的地的策略．为了提高学习效率，利用人工势场(障碍和目标分别对舰载机施加排斥和吸引)的思想来设计奖励函数．

(2)考虑到舰载机的运动约束，根据运动学模型对舰载机进行建模．然后使用 MPC 算法预测舰载机的短期轨迹 u(t)．

(3)实验结果表明，与 DQN 相比，我们的算法与其他算法相比在精度上高了 7%-30%，在路径长度和转弯角度上分别减少 100 个单位和 400-450 度．

## 2 本文方法

### 2.1 场景建模

在甲板模型中，障碍物被分为静态障碍物和动态障碍物，停靠位置、起飞位置等被设置为作业点，作业时间随机生成．舰载机的任务是找到一条从起点到目标点尽最优或次优的无碰撞路径．

#### 2.1.1 状态空间

状态空间作为舰载机在决策之前获得的环境信息，用于帮助舰载机评估环境情况，实时做出决策动作．为了使舰载机更好地了解不断变化的环境，我们将状态分为静止环境状态和预测环境状态．状态值表示为 $s(t):\{s,u(t)\}$．

**2.1.1.1 静止环境状态**

静止环境状态描述当前环境中静止的障碍物和目标点对智能体舰载机的影响．静止环境状态 s 表示为：

$$s(t):\ \{(d^t_{e,x},d^t_{e,y}),(d^t_{o1,x},d^t_{o1,y})\cdots(d^t_{om,x},d^t_{om,y})\}$$

$$d^t_{e,x}=x_e-x^t \qquad d^t_{e,y}=y_e-y^t$$

$$d^t_{om,x}=x^t_{om}-x^t \qquad d^t_{om,y}=y^t_{om}-y^t$$

其中，$(d^t_{e,x},d^t_{e,y})$ 表示舰载机当前位置与目标点之间的坐标距离，而 $(d^t_{o1,x},d^t_{o1,y})\cdots(d^t_{om,x},d^t_{om,y})$ 表示舰载机当前位置与各个静态障碍物之间的坐标距离．$p_e:(x_e,y_e)$ 表示目标点的坐标，$p^t_{o_m}:(x^t_{om},y^t_{om})$ 表示第 m 个静态障碍物的坐标，$p^t:(x^t,y^t)$ 表示舰载机当前位置坐标．

**2.1.1.1 预测环境状态**

预测环境状态描述当前环境中动态障碍物预测位置和目标点对智能体舰载机的影响．预测环境状态 $u(t)$ 表示为：

$$\text{u(t):}\quad \{(d^t_{o'1,x},d^t_{o'1,y}),(d^t_{o'2,x},d^t_{o'2,y})\cdots(d^t_{o'n,x},d^t_{o'n,y})\}$$

$$d^t_{o'n,x}=x^{t+1}_{o'n}-x^t \qquad d^t_{o'n,y}=y^{t+1}_{o'n}-y^t$$

其中，$(d^t_{o'1,x},d^t_{o'1,y})\cdots(d^t_{o'n,x},d^t_{o'n,y})$ 表示动态障碍物的预测位置与舰载机当前位置之间的坐标距离．$p^t_{o'n}:(x^t_{o'n},y^t_{o'n})$ 表示第 n 个动态障碍物的坐标，$p^t:(x^t,y^t)$ 表示舰载机当前位置坐标．

#### 2.1.2 动作空间

舰载机的动作空间表示舰载机根据状态空间决定要执行的动作．我们将动作空间设置为：

$$A:(X,Y)\quad X=x*40\ Y=y*40 \qquad x,y\in(-1,1)$$

$X$ 和 $Y$ 分别表示舰载机在 $x$ 方向和 $y$ 方向上移动的距离．$x$，$y$ 作为 DDPG 中 actor 网络的输出．

#### 2.1.3 奖励函数

深度强化学习中的奖励函数(用于智能体学习的反馈信号)用于评估智能体采取的行为．奖励值设置的好坏决定了智能体最终是否能学到期望的技能，并直接影响算法的收敛速度和最终性能．其中最简单的方法是设置稀疏奖励，只有完成任务，智能体才能获得正回报．但是，此方法无法收集有用的经验数据以帮助智能体学习．因此，网络更新的收敛速度很慢，并且智能体可能无法学习最佳

策略. 在本文中, 我们将人工势场的思想(障碍物和目标分别对舰载机施加排斥和吸引)引入奖励函数的设计中. 设置四类奖励函数: (1)目标引力的奖励. (2)障碍斥力的奖励. (3)碰撞奖励. (4)到达目标的奖励. 奖励函数设置如下:

#### 2.1.3.1 目标引力的奖励

人工势场的引力场是指目标点对智能体产生引力所产生的场, 引力势场的大小与舰载机当前位置和目标点之间的距离成反比. 为了奖励值设计合理, 我们简化引力场的计算方法. 如下所示:

$$r_1=\begin{cases}L & D_{i,\mathrm{e}}^{t}\text{-}D_{i,\mathrm{e}}^{t+1}\geq L\\ D_{i,\mathrm{e}}^{t}\text{-}D_{i,\mathrm{e}}^{t+1} & l<D_{i,\mathrm{e}}^{t}\text{-}D_{i,\mathrm{e}}^{t+1}<L\\ l & D_{i,\mathrm{e}}^{t}\text{-}D_{i,\mathrm{e}}^{t+1}\leq l\end{cases}$$

$$r_1=\begin{cases}-l & D_{i,\mathrm{e}}^{t}\text{-}D_{i,\mathrm{e}}^{t+1}\geq -l\\ D_{i,\mathrm{e}}^{t}\text{-}D_{i,\mathrm{e}}^{t+1} & -\mathrm{L}<D_{i,\mathrm{e}}^{t}\text{-}D_{i,\mathrm{e}}^{t+1}<-l\\ -L & D_{i,\mathrm{e}}^{t}\text{-}D_{i,\mathrm{e}}^{t+1}\leq -L\end{cases}$$

$$\begin{cases}D_{i,e}^{t}=\left\{\left|p^{t},p_{e}^{t}\right|\right\}\\ D_{i,e}^{t+1}=\left\{\left|p^{t+1},p_{e}^{t+1}\right|\right\}\end{cases}$$

目标引力所产生的奖励值 $r$ 表示为 $D_{i,\mathrm{e}}^{t}$ 和 $D_{i,\mathrm{e}}^{t+1}$ 的距离差. $L$、$l$、$-l$、$-L$ 分别代表奖励值的上限和下限. 其中 $D_{i,e}^{t}$、$D_{i,e}^{t+1}$ 分别表示 t 时刻和 t+1 时刻舰载机与目标点之间的距离. $p^{t}$、$p^{t+1}$ 分别表示舰载机在 t 时刻和 t + 1 时刻的位置, $p_e$ 表示目标点的位置.

#### 2.1.3.2 障碍斥力的奖励

人工势场的斥力场是指目标点对智能体产生斥力所产生的场, 斥力场的大小与舰载机当前位置和障碍物之间的距离成反比. 类似于目标吸引的奖励, 我们简化斥力场的计算方法. 如下所示:

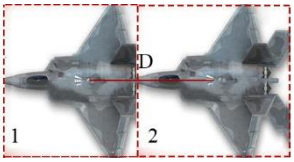



图 1 碰撞状态

$$r_2=\begin{cases}r_2+H & D_{i,j}^{t}\text{-}D_{i,oj}^{t+1}\geq H\\ r_2+D_{i,oj}^{t}\text{-}D_{i,oj}^{t+1} & h<D_{i,oj}^{t}\text{-}D_{i,oj}^{t+1}<H\\ r_2+h & D_{i,oj}^{t}\text{-}D_{i,oj}^{t+1}\leq h\end{cases}$$

$$r_2=\begin{cases}r_2+H & D_{i,j}^{t}\text{-}D_{i,o'k}^{t+1}\geq H\\ r_2+D_{i,o'k}^{t}\text{-}D_{i,ok}^{t+1} & h<D_{i,o'k}^{t}\text{-}D_{i,o'k}^{t+1}<H\\ r_2+h & D_{i,o'k}^{t}\text{-}D_{i,o'k}^{t+1}\leq h\end{cases}$$

$$r_2=\begin{cases}r_2-h & D_{i,oj}^{t}\text{-}D_{i,oj}^{t+1}\geq -h\\ r_2+D_{i,oj}^{t}\text{-}D_{i,oj}^{t+1} & -H<D_{i,oj}^{t}\text{-}D_{i,oj}^{t+1}<-h\\ r_2-H & D_{i,oj}^{t}\text{-}D_{i,oj}^{t+1}\leq -H\end{cases}$$

$$r_2=\begin{cases}r_2-h & D_{i,o'k}^{t}\text{-}D_{i,o'k}^{t+1}\geq -h\\ r_2+D_{i,o'k}^{t}\text{-}D_{i,o'k}^{t+1} & -H<D_{i,o'k}^{t}\text{-}D_{i,o'k}^{t+1}<-h\\ r_2-H & D_{i,o'k}^{t}\text{-}D_{i,o'k}^{t+1}\leq -H\end{cases}$$

$$\begin{cases}D_{i,oj}^{t}=\left\{\left|p^{t},p_{oj}^{t}\right|j=1,2\cdots m\right\}\\ D_{i,oj}^{t+1}=\left\{\left|p^{t+1},p_{oj}^{t+1}\right|j=1,2\cdots m\right\}\end{cases}$$

$$\begin{cases}D_{i,o'k}^{t}=\left\{\left|p^{t},p_{o'k}^{t}\right|k=1,2\cdots n\right\}\\ D_{i,o'k}^{t+1}=\left\{\left|p^{t+1},p_{o'k}^{t+1}\right|k=1,2\cdots n\right\}\end{cases}$$

障碍物斥力所产生的奖励值 $r$ 表示为 $D_{i,oj}^{t}$ 和 $D_{i,oj}^{t+1}$ 或 $D_{i,o'k}^{t}$ 和 $D_{i,o'k}^{t+1}$ 的距离差. $H$、$h$、$-h$、$-H$ 分别代表奖励值的上限和下限. 其中 $D_{i,oj}^{t}$、$D_{i,oj}^{t+1}$ 分别表示 t 时刻和 t+1 时刻舰载机与静态障碍物之间的距离. $D_{i,o'k}^{t}$、$D_{i,o'k}^{t+1}$ 分别表示 t 时刻和 t+1 时刻舰载机与动态障碍物之间的距离. $p^{t}$、$p^{t+1}$ 分别表示舰载机在 t 时刻和 t + 1 时刻的位置, $p_{oj}^{t}$、$p_{oj}^{t+1}$ 分别表示静态障碍物在 t 时刻和 t + 1 时刻的位置. $p_{o'k}^{t}$、$p_{o'k}^{t+1}$ 分别表示动态障碍物在 t 时刻和 t + 1 时刻的预测位置.

#### 2.1.3.3 碰撞奖励

如图 1 碰撞状态所示, 当 $\exists \mathrm{D}<d$ 时, $r_3=-50$. $\mathrm{D}=\left\{\left|p^{t},p_{oj}^{t}\right||j=1,2\cdots m,m+1\cdots m+n\right\}$; 其中, $d$ 表示舰载机间的安全距离; $p^{t}$ 表示舰载机 t 时刻的位置; $p_{oj}^{t}$ 任意一个障碍物 t 时刻的位置.

#### 2.1.3.4 到达目标点奖励

当 $p^{t}=p_{e}^{t}$ 时, $r_4=200$. 其中 $p^{t}$ 表示舰载机当

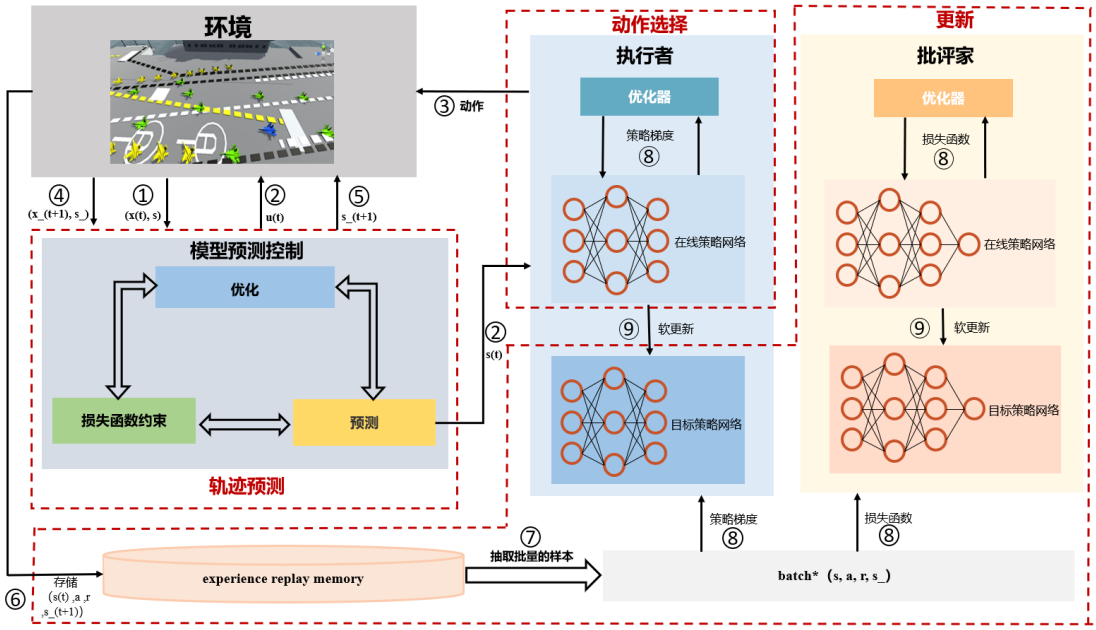



图 2　算法框架图

前位置，$p_e^t$表示目标位置.

#### 2.1.3.5　总奖励

如下所示，总奖励值被定义为:

$$R = \lambda_1 * r_1 + \lambda_2 * r_2 + \lambda_3 * r_3 + \lambda_4 * r_4$$

其中，$\lambda_1$、$\lambda_2$、$\lambda_3$、$\lambda_4$分别代表四种奖励值的权重.

### 2.2　算法框架图

#### 2.2.1　轨迹预测

轨迹预测任务旨在根据目标当前或者历史轨迹与环境信息，对该目标未来的行驶轨迹进行预测. 为了降低环境的动态性，我们使用模型预测控制算法(MPC)来预测动态障碍物未来的坐标，这种算法具有控制效果好，鲁棒性强的优点. 如图 1 算法框架所示，MPC 的预测模型将静止环境状态 s 和动态障碍物的历史轨迹

$x(t) = \{(p_1^t, p_1^{t+1} \cdots p_1^{t+N}), (p_2^t, p_2^{t+1} \cdots p_2^{t+N}) \cdots (p_n^t, p_n^{t+1} \cdots p_n^{t+N})\}$作为输入，通过 MPC 的预测模型计算出动态障碍物的预测轨迹

$y(t) = \{p_1^{t+N+1}, p_2^{t+N+1} \cdots p_n^{t+N+1}\}$. 通过动态障碍物的预测轨迹和当前舰载机的位置计算预测环境状态$u(t)$. 连接预测环境状态$u(t)$和静止环境状态 s 作为环境状态$s(t)$.

#### 2.2.2　动作选择

强化学习的决策的本质是动作选择. 如图 1 所示，$s(t)$为 DDPG 执行者在线策略网络的输入，输出舰载机要执行的动作$a:(x, y)$. 舰载机执行动作后，静止环境状态变为 $s\_$，动态障碍物的历史轨迹$x(t)$更新为$x(t+1)$. 同时获得奖励值 r. 然后，将动态障碍物的历史轨迹$x(t+1)$和静止环境状态 $s_$再次输入给 MPC 网络获得新的预测环境状态$u(t+1)$和新的环境状态$s_(t+1)$. 最后环境将$(s(t), a_t, r_t, s(t+1))$存储在经验池中.

#### 2.2.2　更新

DDPG 是基于 AC 框架对 DQN 算法的扩展. 它保留了 DQN 的双网络结构和经验重放. 但与 DQN 的硬更新不同，DDPG 采用软更新的方法来更新目标网络. 更新 Actor 和 Critic 网络时，从经验池中采样 N 个小批量样本，然后根据当前的目标 Q 值计算 Critic 网络的损失函数$L(\theta^Q)$. 同时，通过策略梯度的方法来更新 Actor 网络.

$$y_t = \begin{cases} R_t \\ R_t + \gamma Q'(\phi(S'_t), \pi_{\theta'}(\phi(S'_t)), \omega) \end{cases}$$

$$L(\theta^Q) = \frac{1}{N} \sum_t^N (Y_i - Q(s_i, a_i | \theta^Q))^2$$

$$\nabla_{\theta^\mu} J \approx N^{-1}\sum_{t}^{N}\nabla_{a_t} Q(s_t, a_t|\theta^Q)\nabla_{\theta^\mu}\mu(s_t|\theta^\mu)$$

最后，通过软更新来更新两个目标网络参数 $\theta^{\mu'}$ 和 $\theta^{Q'}$：

$$\begin{cases}\theta^{Q'} = \tau\theta^{Q} + (1-\tau)\theta^{Q'} \\ \theta^{\mu'} = \tau\theta^{\mu'} + (1-\tau)\theta^{\mu'}\end{cases}$$

其中，$\tau$ 是可配置的常数系数，用于调节软更新因子。

## 3 仿真分析与验证

本文算法基于 CUDA 9.0 深度学习框架和 Tensorflow 1.14.0 利用 Unity 3D 进行仿真. 运行环境为 Intel i7 870 0K 六核 3.7GHZ 主频, GPU 为 GeForce RTX 2070 SUPER.

### 3.1 场景设置

在仿真系统中设计了 2 个不同的场景, 如图 2 和图 3 所示, 在甲板上, 蓝色舰载机需要避开动静态障碍物(绿色和黄色舰载机分别代表动态和静态障碍物), 找到一条从起点到目标点(白板的位置)的无碰撞路径. 如图 2 所示, 在场景 1 中, 我们设置了 5 个动态障碍物和 15 个静态障碍物, 如图 3 所示, 为了验证我们的算法在高动态环境中的有效性, 在场景 2 中, 我们设置了 9 个动态障碍物和 15 个静态障碍物.

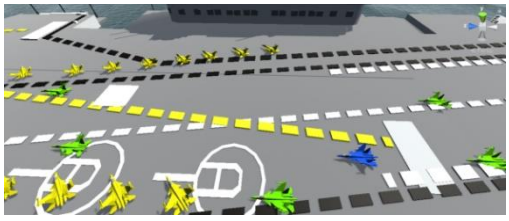


图 3 场景一

图 4 场景二

### 3.2 障碍物设置

障碍物分为动态障碍物和静态障碍物. 静态障碍物指舰载机在寻径的过程中, 一直静止在初始位置上的障碍物. 动态障碍物指在舰载机寻径的过程中会发生运动的障碍物(正在或即将做任务的舰载机), 其状态在临时静态和临时动态之间转换.当障碍物从甲板上一个作业点向另一个作业点运动时其状态为临时动态, 障碍物进行作业时其状态为临时静态. 开始时, 所有动态障碍物的状态默认为临时动态, 从预先设定好的初始位置 $(p_x^o, p_y^o)$, 以初始速度 $(v_x^o, v_y^o)$ 开始运动, 当障碍物运动到作业位置 $(p_x^t, p_y^t)$ 时, 其状态更改为临时静止, 在此位置上进行随机短时间作业, 作业完成后, 其状态变为临时动态且随机生成新的速度 $(v_x, v_y)$.

### 3.3 仿真分析

我们分别描述两个场景中智能体和动态障碍物的轨迹. 如图 3、4 所示, 智能体和动态障碍物分别由蓝色和绿色舰载机表示且其轨迹分别为蓝色和绿色虚线. 黄色舰载机在整个寻径过程中静止不动. 舰载机和动态障碍物最终的落脚点用标有数字的红色矩形表示. 如图 5 场景 1 的轨迹图所示, 当舰载机根据障碍物预测轨迹判断出它可能会和障碍物 3 发生碰撞时, 舰载机上下移动来躲避障碍物, 随后直接向目标点移动. 如图 6 场景 2 轨迹图所示, 当障碍物发现 2 和 5 出现在舰载机周围时, 舰载机向左下方移动以避免碰撞两个动态障碍物, 并确保舰载机靠近目标.

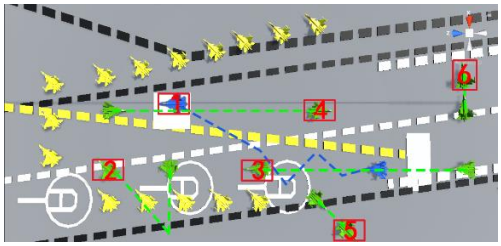



图 5 场景 1 轨迹图

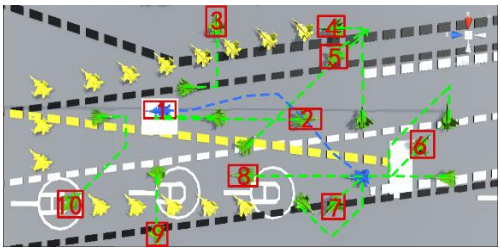



图 6 场景 2 轨迹图

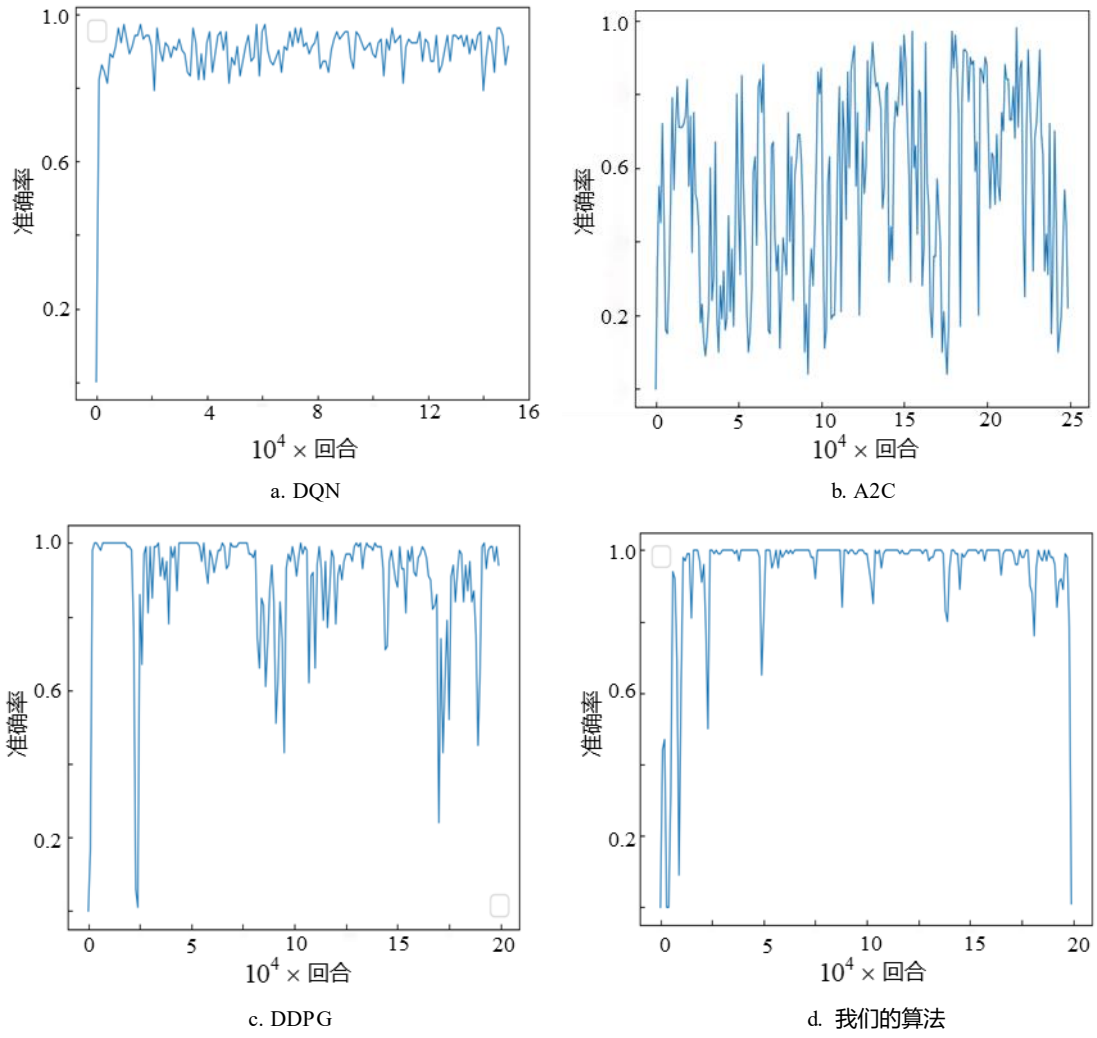



图 7　场景一准确率

### 3.4　评价指标

为了保证本文算法在大量动态障碍物中仍能保持较高的准确率，我们通过增加动态障碍物来设置多个实验场景，从准确率、路径长度、平均奖励值、平均转弯角度这 4 个方面和 DDPG，DQN 和 A2C 算法进行比较.

(1)准确率指舰载机不碰到任何障碍物的情况下从起点到达目的地的概率，准确率越高算法越有效，具体计算如公式(1)所示，当舰载机无碰撞到达终点时，$f$ 为 1，否则为 0:

$$T_r = \frac{\sum_{i=1}^{100} f}{100} \tag{1}$$

(2)路径长度是直接反应算法质量的重要评价指标，为了避免算法偶然性带来的误差，我们随机抽取 100 组数据计算其平均路径长度.

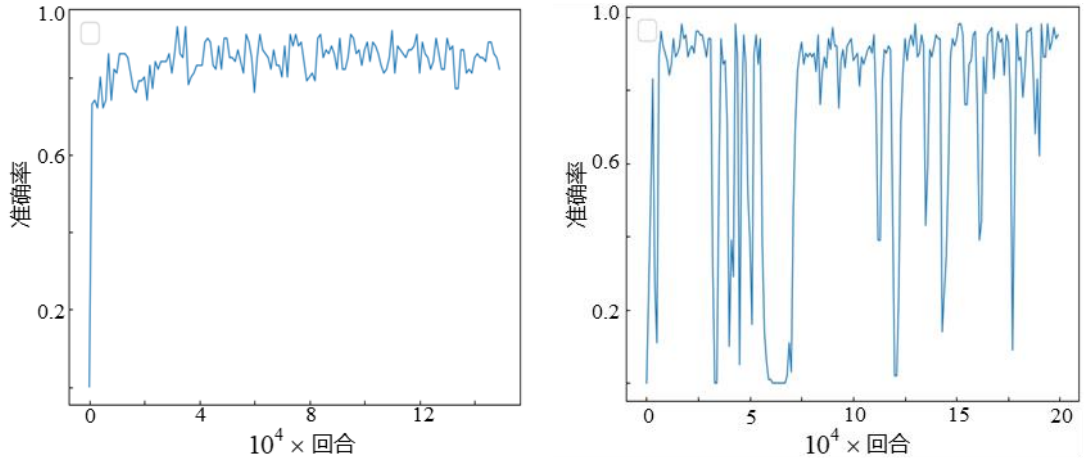
(3)励值 $r_t$ 是一个标量反馈信号，其衡量智能体在时刻 $t$ 所执行动作 $a$ 的表现. 每 100 步我们计算一次平均奖励值，平均奖励值越高表示舰载机选择的路线越短越合理.

(4)转弯角度是指舰载机在一次无碰撞轨迹中转过的角度之和. 和路径长度一样，随机抽取 100 组无碰撞轨迹计算平均转弯角度.

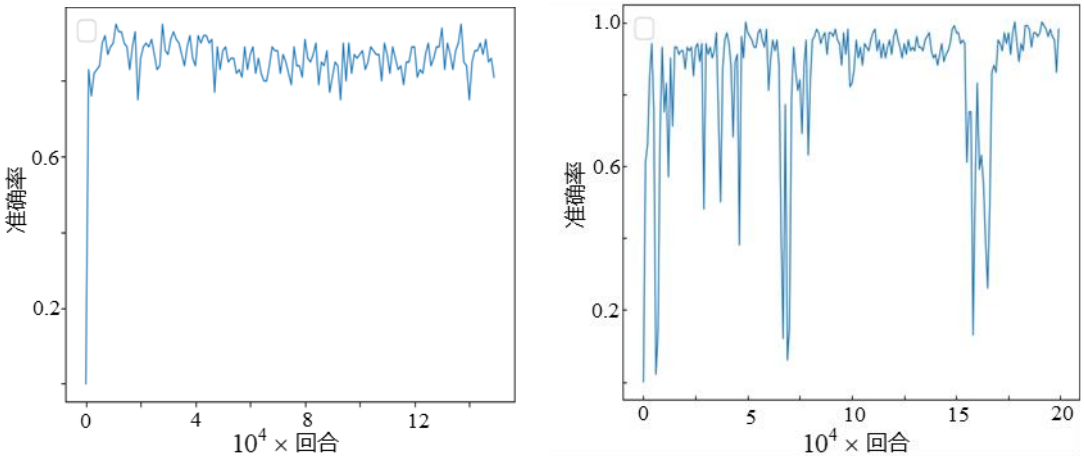
### 3.5　仿真结果

#### 3.5.1　准确率

首先我们比较了两个场景中不同算法的准确率. 如表 1 所示，随机抽样 100 组准确率计算平均值，在场景 1 中，我们算法的平均准确率为 98%，DDPG 和 DQN 算法分别为 92%、90%. 而 A2C 算




a. DQN b. A2C




c. DDPG d. 我们的算法

图 8 场景二准确率

法的准确率最低只有 67%. 与 DQN、A2C 和 DDPG 相比，我们的算法的准确率分别增加了 8%、31%和 6%. 如图 7 所示，DDPG 和我们的算法的准确率都可以达到 100%，但我们的算法达到 100%的次数最多且最稳定. 而 A2C 和 DQN 算法的准确率较低，且 A2C 算法的稳定性最差. 如表 1 所示，在场景 2 中，四种算法的准确率均有所下降，但我们的算法的准确率仍然最高，且可以达到 91%.

表 1 准确率

| | 场景一 | 场景二 |
|---|---|---|
| DQN | 90% | 84% |
| A2C | 67% | 64% |
| DDPG | 92% | 82% |
| 我们的算法 | 98% | 91% |

3.5.2 平均奖励值

为了比较四种算法的学习效果，我们比较了四种算法的平均奖励值. 如图 9 场景 1 中四种算法的平均奖励值所示，由于 A2C 算法难收敛，其平均奖励值只能稳定在 100 左右，而其他算法都能达到 150. 如图 9、10 所示，相比于我们的算法，DDPG 的平均奖励值更不稳定，即 DDPG 在没有预测的情况下学习效果较差. DQN 和我们的算法的平均奖励值都稳定在 150 左右，而我们的算法具有更高的稳定性. 对比各个算法，我们的算法的平均奖励值最高且最稳定，因此具有最佳的学习效果.

3.5.2 平均路径长度和平均转弯角度

在航母甲板上中，找到一条平滑的最优或次优的路径不仅可以减少资源消耗，缩短规划时间而且还可以提高任务调度的效率. 因此我们比较

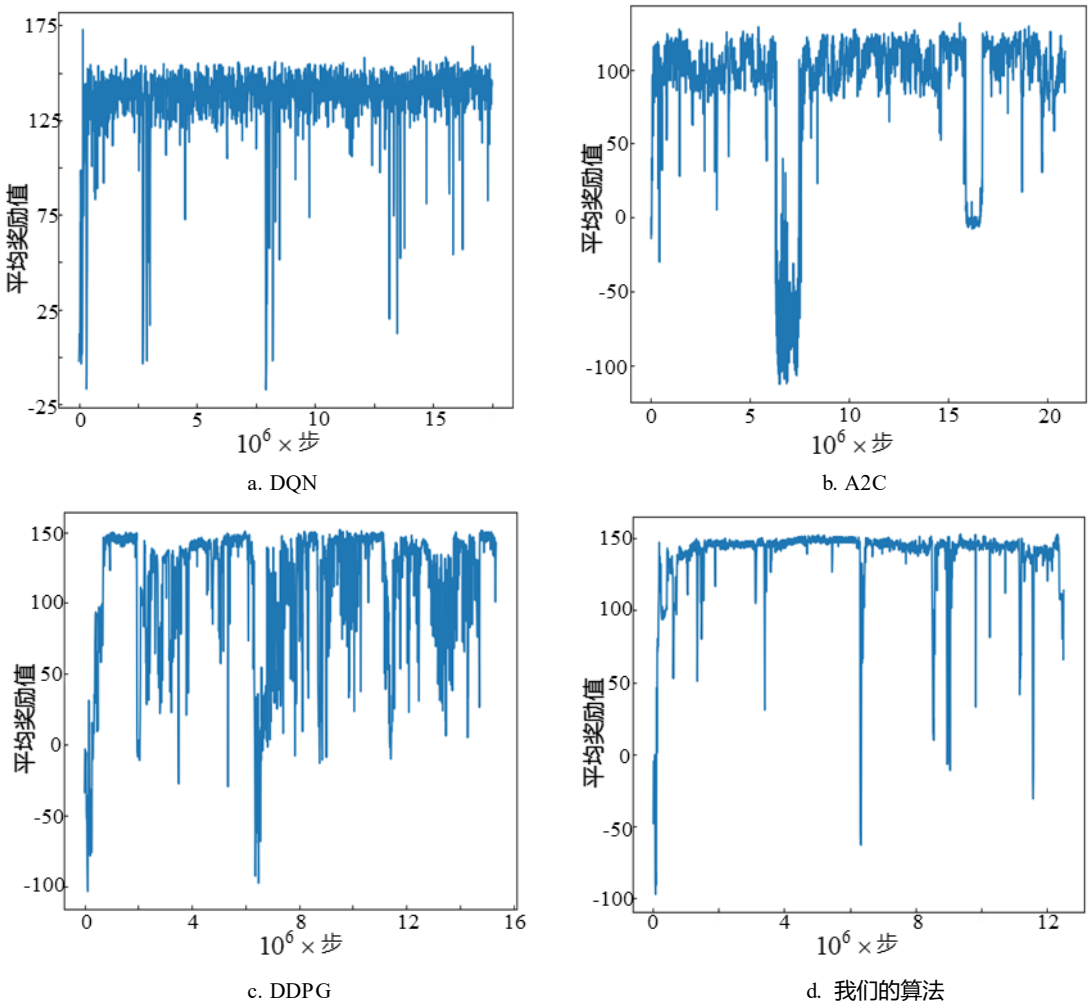



图 9　场景一平均奖励值

各个算法的平均路径长度和平均转弯角度. 如表 2 平均路经长度和表 3 平均转弯角度所示, 场景 1 和场景 2 中四种算法的平均路径长度和平均转弯角度差距不大. 对比四种算法, A2C、DDPG 和我们的算法规划的路径长度和平均转弯角度基本相似. 由于 DQN 只能处理离散动作, DQN 所规划的路径的平均路径长度最长且平均转弯角度最大.

表 2　平均路径长度

| | 场景一 | 场景二 |
|---|---|---|
| DQN | 438.22 | 465.45 |
| A2C | 352.60 | 324.16 |
| DDPG | 328.97 | 320.61 |
| 我们的算法 | 328.95 | 315.68 |

表 3　平均转弯角度

| | 场景一 | 场景二 |
|---|---|---|
| DQN | 586.73 | 522.62 |
| A2C | 142.03 | 158.00 |
| DDPG | 139.96 | 134.28 |
| 我们的算法 | 139.80 | 128.41 |

## 4　结　语

本文设计了一种基于 DDPG 和 MPC 的路径规划新算法, 该算法结合了深度强化学习的感知和决策能力以及 MPC 的预测能力, 应用于高度不确定的场景. MPC 实现了动态障碍物的轨迹预测, 大大降低了环境的不确定性, 有效地解决了传统算法在动态环境中面临的收敛速度慢, 泛化能力差等问题. DDPG 适用于解决连续状态空间的问题, 该问题更符合真实场景. 为了验证算法的效率, 我们选择 Unity3D 来对甲板上舰载机的路径规划进

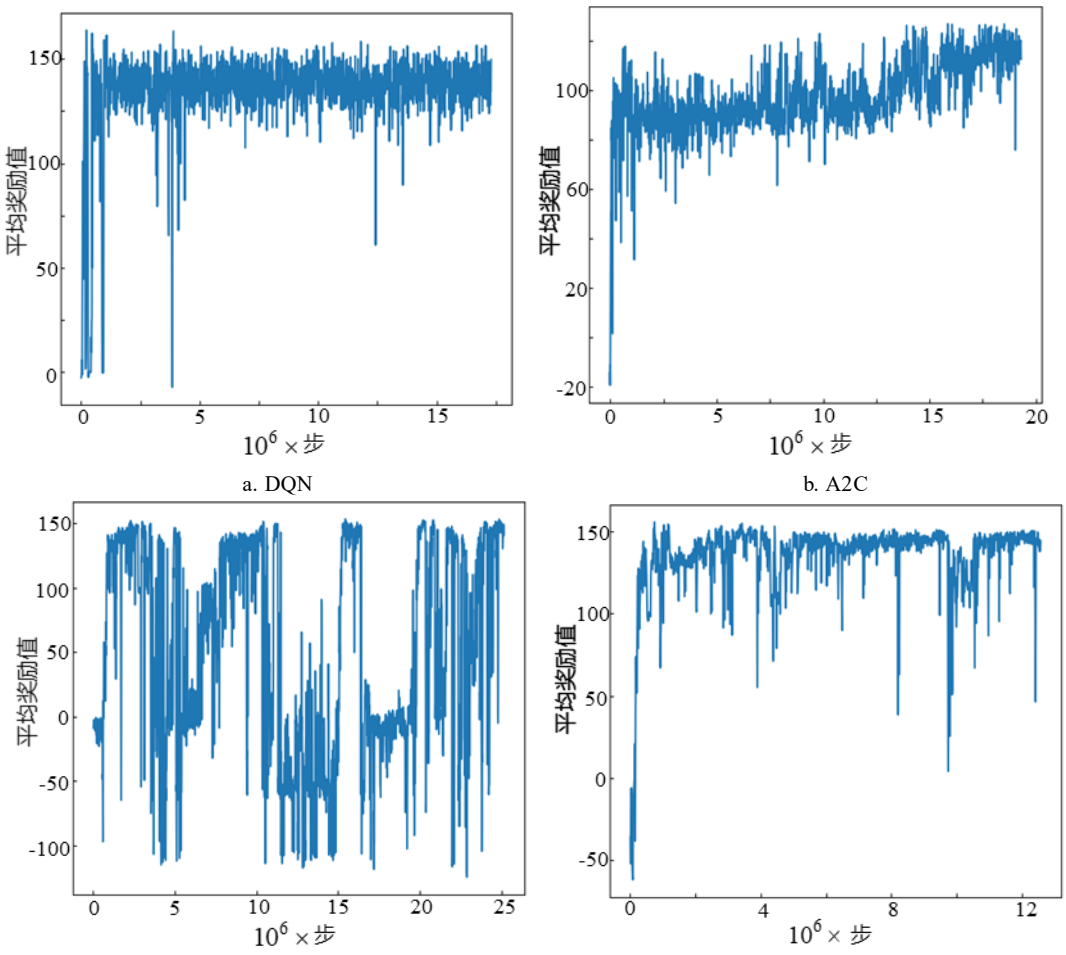



图 10 场景二平均奖励值

仿真，并两个场景中智能体和动态障碍物的轨迹进行了分析. 其结果表明，与其他算法相比，我们的算法在精度上提高了 7%-30%；与 DQN 相比，我们的算法在路径长度和转弯角度方面分别减少了 100 个单位和 400-450 度.

## 参考文献(References):


[1] Wang, D. Indoor mobile-robot path planning based on an improved a* algorithm. Journal of Tsinghua University Science and Technology 2012;52(8):1085–1089.

[2] Li, L, Ye, T, Tan, M, Chen, Xj. Present state and future development of mobile robot technology research. Robot 2002;24(5):475–480.

[3] Lozano-Pérez, T, Wesley, MA. An algorithm for planning collision-free paths among polyhedral obstacles. Communications of the ACM 1979;22(10):560–570.

[4] Barraquand, J, Latombe, JC. Robot motion planning: A distributed representation approach. The International Journal of Robotics Research 1991;10(6):628–649.

[5] Hoffmann, G, Waslander, S, Tomlin, C. Quadrotor helicopter trajectory tracking control. In: AIAA guidance, navigation and control conference and exhibit. 2008, p. 7410.

[6] Bohlin, R, Kavraki, LE. Path planning using lazy prm. In: Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065); vol. 1. IEEE; 2000, p. 521–528.

[7] DIJKSTRA E W.A note on two problems in connexion with graphs[J].Numerische Mathematik, 1959, 1 (1):2569-271.

[8] HART P E, NILSSON N J, RAPHAEL B.AFormal basis for the heuristic determination of minimum cost paths[J].AcmSigart Bulletin, 1972, 4 (37):28-29.

[9] Hwang, YK, Ahuja, N, et al. A potential field approach to path planning. IEEE Transactions on Robotics and Automation 1992;8(1):23–32.

[10] Bruce, J, Veloso, MM. Real-time randomized path planning for robot navigation. In: Robot Soccer World Cup. Springer; 2002, p. 288–295.

[11] Stentz, A. Optimal and efficient path planning for partially known environments. In: Intelligent unmanned ground vehicles. Springer; 1997, p. 203–220.

[12] Dorigo M, Gambardella L M. Ant colony system: a cooperative learning approach to the traveling salesman problem[J]. IEEE Trans on Evolutionary Computation, 1997, 1(1):53-66.

[13] Kennedy J. Particle Swarm Optimization. Boston: Springer,

2017: 967-972.

[14] Eberhart R, Kennedy J. A new optimizer using particle swarm theory // Proceeding of the 6th International Symposium on Micromachine and Human Science. Pkcataway, 1995: 39-43.

[15] Yan, C, Xiang, X. A path planning algorithm for uav based on improved q-learning. In: 2018 2nd International Conference on Robotics and Automation Sciences (ICRAS). IEEE; 2018, p. 1–5.

[16] Bouhamed, O, Ghazzai, H, Besbes, H, Massoud, Y. Q-learning based routing scheduling for a multi-task autonomous agent. In: 2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS). IEEE; 2019, p. 634–637.

[17] Sichkar, VN. Reinforcement learning algorithms in global path planning for mobile robot. In: 2019 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM). IEEE; 2019, p. 1–5.

[18] Watkins, CJ, Dayan, P. Q-learning. Machine learning 1992;(3-4):279–292.

[19] Liu, Q, Shi, L, Sun, L, Li, J, Ding, M, Shu, F. Path planning for uav-mounted mobile edge computing with deep reinforcement learning. IEEE Transactions on Vehicular Technology 2020;69(5):5723–5728.

[20] Mnih, V, Kavukcuoglu, K, Silver, D, Graves, A, Antonoglou, I, Wierstra, D, et al. Playing atari with deep reinforcement learning. arXiv preprint arXiv:13125602 2013;.

[21] Sewak, M. Deep q network (dqn), double dqn, and dueling dqn. In: Deep Reinforcement Learning. Springer; 2019, p. 95–108.

[22] Zeng, Y, Xu, X, Jin, S, Zhang, R. Simultaneous navigation and radio mapping for cellular-connected uav with deep reinforcement learning. arXiv preprint arXiv:200307574 2020;

[23] Peters, J, Schaal, S. Policy gradient methods for robotics. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE; 2006, p. 2219–2225.

[24] Camacho, EF, Alba, CB. Model predictive control. Springer science & business media; 2013.

[25] Lillicrap, TP, Hunt, JJ, Pritzel, A, Heess, N, Erez, T, Tassa, Y, et al. Continuous control with deep reinforcement learning. arXiv preprint arXiv:150902971 2015;.