Article
文章

UAV Path Planning and Obstacle Avoidance Based on Reinforcement Learning in 3D Environments

基于三维环境中强化学习的无人机路径规划与避障

Guan-Ting Tu and Jih-Gau Juang *
涂冠廷和蒋志高 *
Department of Communications, Navigation and Control Engineering, National Taiwan Ocean University, Keelung 20224, Taiwan
通讯处: 台湾基隆 20224，国立台湾海洋大学通讯与导航控制工程系
* Correspondence: jgjuang@mail.ntou.edu.tw
* 通讯作者:jgjuang@mail.ntou.edu.tw

Abstract: This study proposes using unmanned aerial vehicles (UAVs) to carry out tasks involving path planning and obstacle avoidance, and to explore how to improve work efficiency and ensure the flight safety of drones. One of the applications under consideration is aquaculture cage detection; the net-cages used in sea-farming are usually numerous and are scattered widely over the sea. It is necessary to save energy consumption so that the drones can complete all cage detections and return to their base on land. In recent years, the application of reinforcement learning has become more and more extensive. In this study, the proposed method is mainly based on the Q-learning algorithm to enable improvements to path planning, and we compare it with a well-known reinforcement learning state-action-reward-state-action (SARSA) algorithm. For the obstacle avoidance control procedure, the same reinforcement learning method is used for training in the AirSim virtual environment; the parameters are changed, and the training results are compared.

摘要: 本研究提出使用无人航空器 (UAVs) 执行涉及路径规划和避障的任务，并探讨如何提高无人机的工作效率及确保其飞行安全。考虑的应用之一是水产养殖笼检测；海上养殖使用的网笼通常数量众多且分布在海面上。有必要节约能耗，以便无人机能够完成所有笼检测并返回陆地基地。近年来，强化学习的应用越来越广泛。在本研究中，提出的方法主要基于 Q 学习算法来改进路径规划，并将其与著名的强化学习状态-动作-奖励-状态-动作 (SARSA) 算法进行比较。对于避障控制程序，使用相同的强化学习方法在 AirSim 虚拟环境中进行训练；改变参数，并比较训练结果。

Keywords: unmanned aerial vehicles; UAV; reinforcement learning; Q-learning; deep Q-learning; obstacle avoidance; path planning
关键词: 无人航空器；UAV；强化学习；Q 学习；深度 Q 学习；避障；路径规划

1. Introduction

1. 引言

check for updates
检查更新
Citation: Tu, G.-T.; Juang, J.-G. UAV Path Planning and Obstacle Avoidance Based on Reinforcement Learning in 3D Environments. Actuators 2023, 12, 57. https:// doi.org/10.3390/act12020057
引用:Tu, G.-T.; Juang, J.-G. 基于三维环境中强化学习的无人机路径规划与避障. Actuators 2023, 12, 57. https:// doi.org/10.3390/act12020057
Academic Editor: Ronald M. Barrett
学术编辑:Ronald M. Barrett
Received: 9 December 2022
收到日期:2022 年 12 月 9 日
Revised: 23 January 2023
修订日期:2023 年 1 月 23 日
Accepted: 23 January 2023
接受日期:2023 年 1 月 23 日

Published: 28 January 2023 4.0/).
发表日期:2023 年 1 月 28 日 4.0/).

With the development of integrated circuit technology, rapid growth in computing power has led to a surge in the application of drones. Advances in microcontrollers have created a new world for many innovations. In recent years, work efficiency and human resources shortages have been extensively discussed. The use of drones will greatly reduce the death and injury of pilots flying planes, which is another great advantage of drones, so that research into drones has become more widespread. With the development of drones, some problems have been addressed or have evolved, such as aquaculture, infrastructure inspection, environmental monitoring, and agricultural spraying. Therefore, the development of path planning and obstacle avoidance systems needs to keep pace with the times, so that drones can effectively perform and complete automation tasks in various environments.

随着集成电路技术的发展，计算能力的快速增长导致无人机应用的激增。微控制器的进步为许多创新创造了一个新世界。近年来，工作效率和人力资源短缺问题被广泛讨论。无人机的使用将大大减少飞行员驾驶飞机时的伤亡，这是无人机的另一个巨大优势，因此对无人机的研究变得更加普遍。随着无人机的发展，一些问题已经被解决或演变，比如水产养殖、基础设施检查、环境监测和农业喷洒。因此，路径规划和避障系统的发展需要与时俱进，以便无人机能够有效地在各种环境中执行和完成自动化任务。

In recent years, unmanned aerial vehicles (UAVs) have gradually become an important issue in many studies; these are divided into two categories, comprising fixed-wing aircraft and multi-axis rotorcraft. The wing of a fixed-wing aircraft does not need to rotate and the aircraft uses the airflow passing around the wing to provide lift. Fixed-wing aircraft can only fly forward and cannot fly in any other direction. In addition, fixed-wing aircraft are unable to make an immediate sharp turn or to fly backward. Furthermore, the runway for fixed-wing aircraft must be long enough for them to be able to take off as well as to land. Conversely, the rotorcraft, also known as the multirotor, has the capacity for vertical take-off and landing (VTOL); it can thus enjoy unlimited movement and freedom in the air. The multirotor can hover above the target location and can drop items of set weight within the load range for delivery to the designated location. The application of this study is to use unmanned multi-rotor aircraft to replace traditional human operations. The multirotor can drop sensors to collect environmental information from the cage net in the ocean for the purposes of aquaculture. In order to carry out this mission, making the UAV fly safely to its destination is the most important issue. The main purpose of this study is to plan appropriate paths and design an obstacle avoidance system to ensure that the UAV has a safe flight to its destination.

近年来，无人航空器 (UAVs) 逐渐成为许多研究的重要议题；它们分为两大类，包括固定翼飞机和多轴旋翼机。固定翼飞机的机翼无需旋转，飞机利用流过机翼的气流提供升力。固定翼飞机只能向前飞行，不能向其他方向飞行。此外，固定翼飞机无法立即进行急转弯或向后飞行。而且，固定翼飞机的跑道必须足够长，以便它们能够起飞和降落。相反，旋翼机，也称为多旋翼，具有垂直起降 (VTOL) 的能力；因此，它在空中拥有无限的移动和自由度。多旋翼可以在目标位置上空悬停，并且可以在载重范围内投放设定重量的物品到指定位置。本研究的应用是使用无人多旋翼飞机替代传统的手工操作。多旋翼可以投放传感器，从海洋中的笼网收集养殖用的环境信息。为了执行这一任务，确保 UAV 安全飞行至目的地是最重要的问题。本研究的主要目的是规划适当的路径并设计避障系统，以确保 UAV 能够安全飞行至目的地。

In past decades, the most frequently used path-planning algorithms include Dijkstra's algorithm, A Star (A*), and D*. Dijkstra's algorithm and A* plan the projected path from the starting position to the goal position. A* also considers the distance from the starting position to the goal [1]. D* plans the path by setting off from the goal to the starting position. It has the ability to adjust the projected path in the face of obstacles [2]. Although these algorithms can successfully plan paths, the drawback is that they are computationally expensive when the search space is large. Rapidly exploring random tree methods represents a less expensive approach because it can quickly explore a search space with an iterative function [3]. Still, these algorithms are time-consuming and may not be appropriate for UAVs. The main objective of path planning is to have less computational time needed for optimal path planning. The path generated by the path-planning algorithms should be optimal so that it consumes minimal energy, takes less time, and reduces the likelihood of collisions between UAVs and obstacles. Classical algorithms cannot truly reflect the actual needs of UAV path planning. In this paper, a UAV path-planning algorithm, based on reinforcement learning, was applied according to the simulated scenarios. The UAV can follow this planned path and perform visual real-time collision avoidance by the use of depth images and deep-learning neural networks.

在过去的几十年里，最常用的路径规划算法包括迪杰斯特拉算法、A 星 (A*) 算法和 D* 算法。迪杰斯特拉算法和 A* 算法从起始位置规划到目标位置的预期路径。A* 算法还考虑了从起始位置到目标位置的距离 [1]。D* 算法则从目标位置开始，反向规划到起始位置。它具有在面对障碍物时调整预期路径的

能力 [2]。尽管这些算法能够成功规划路径，但在搜索空间较大时，它们的计算成本较高。快速探索随机树方法代表了成本较低的途径，因为它可以用迭代函数快速探索搜索空间 [3]。然而，这些算法耗时较多，可能不适合无人机 (UAVs)。路径规划的主要目标是减少优化路径规划所需的计算时间。路径规划算法生成的路径应当是最优的，以便消耗最少的能量，花费最短的时间，并减少无人机与障碍物之间的碰撞可能性。经典算法无法真正反映无人机路径规划的实际需求。在本文中，根据模拟场景，应用了一种基于强化学习的无人机路径规划算法。无人机可以遵循这个规划路径，并通过使用深度图像和深度学习神经网络进行视觉实时避障。

The UAV can either be operated by the operator visually, or the operator can wear special glasses to fly in first-person view (FPV) mode. In addition, one can write code to use a microcontroller to operate the UAV. The traditional control-system technology uses a PID controller to give the flight attitude of UAVs good performance [4,5]. In addition, the adjustment of PID parameters, based on reinforcement learning, has been proposed by the authors of [6, 7] . In recent years, the use of drone applications in many fields has become more and more common. T. Lampersberger et al. proposed the application of a radar system and antenna in UAVs [8]. P. Udvardy et al. used drones to model a church in 3D and evaluated the results [9]. In [10], Z. Ma et al. discussed communication research using UAVs in an air-to-air environment. K. Feng et al. used UAVs for package delivery research [11]. Among these applications, UAV path planning and obstacle avoidance issues are worthy of study and discussion. In addition, research into the applications of reinforcement learning is also very popular. For example, Alphago is a well-known reinforcement learning application [12]. In other studies [13-16], research based on the Q-learning algorithm was applied to path planning. A. Anwar et al. used reinforcement learning methods to study obstacle avoidance control [17]. The final research results are amazing, and the content produced is excellent. In [18-21], many deep reinforcement learning methods for obstacle avoidance were proposed and the results were analyzed. With the help of a number of reinforcement learning experiences, this study proposes a comparison of path planning based on the SARSA algorithm and the Q-learning algorithm in reinforcement learning, along with the use of a deep Q-learning algorithm for obstacle avoidance simulation experiments.

无人机可以由操作员通过视觉操作，或者操作员可以佩戴特殊眼镜以第一人称视角 (FPV) 模式飞行。此外，可以编写代码使用微控制器来操作无人机。传统的控制系统技术使用 PID 控制器来保证无人机飞行姿态的良好性能 [4,5]。另外，基于强化学习调整 PID 参数的方法已经被本文作者提出 [6, 7] 。近年来，无人机在许多领域的应用变得越来越普遍。T. Lampersberger 等人提出了在无人机中应用雷达系统和天线 [8]。P. Udvardy 等人使用无人机对教堂进行 3D 建模并评估了结果 [9]。在 [10] 中，Z. Ma 等人讨论了在空对空环境中使用无人机进行通信研究。K. Feng 等人将无人机应用于包裹递送研究 [11]。在这些应用中，无人机的路径规划和避障问题值得研究和讨论。此外，对强化学习应用的研究也非常流行。例如，Alphago 是一个著名的强化学习应用 [12]。在其他研究 [13-16] 中，基于 Q 学习算法的研究被应用于路径规划。A. Anwar 等人使用强化学习方法研究避障控制 [17]。最终的研究成果令人惊叹，产生的内容非常优秀。在 [18-21] 中，提出了许多用于避障的深度强化学习方法并分析了结果。借助大量的强化学习经验，本研究提出了基于 SARSA 算法和 Q 学习算法的路径规划比较，以及使用深度 Q 学习算法进行避障模拟实验。

A brief introduction to path planning and obstacle avoidance, based on reinforcement learning, was presented in [22]. In this study, a detailed description and extension methods are presented. The proposed methods are divided into two parts. The first part concerns path planning. In path planning, we compare the Q-learning algorithm with the SARSA algorithm. The principles of these two algorithms are similar, but they still have their own characteristics. Therefore, we propose two reinforcement-learning methods for path-planning experiments. In the second part, obstacle avoidance control is utilized; because the input depth image data is more complicated, deep Q-learning is used for experiments, then the training results are observed in a simulated environment. Considering the future testing of reinforcement learning methods involving physical drones, we use ultrasonic sensor modules to assist in side-obstacle avoidance, so that the drone can maintain a safe distance from obstacles. This study applies reinforcement learning methods to train the UAV in a simulated environment, followed by testing and then analysis of the results.

在文献 [22] 中简要介绍了基于强化学习的路径规划和避障。在本研究中，详细描述了方法并提出了扩展方法。提出的方法分为两部分。第一部分关注路径规划。在路径规划中，我们比较了 Q 学习算法和 SARSA 算法。这两个算法的原理相似，但它们仍然具有各自的特点。因此，我们提出了两种用于路径规划实验的强化学习方法。在第二部分中，应用了避障控制；由于输入的深度图像数据更为复杂，因此使用深度 Q 学习进行实验，然后在模拟环境中观察训练结果。考虑到未来对涉及实体无人机的强化学习方法的测试，我们使用超声波传感器模块来辅助侧向避障，以便无人机能够与障碍物保持安全距离。本研究应用强化学习方法在模拟环境中训练无人机，随后进行测试并对结果进行分析。

2. System Description

2. 系统描述

In this study, a hexacopter was used to perform an assigned task, for instance, to collect sensor data for the purposes of aquaculture. This section details the main architecture, sensors, and computer hardware configuration used in deep-learning training. The computational hardware system includes an onboard computer and off-line training computer. The basic UAV system consists of a flight control board, power supply, sensors, motors, 3DR telemetry radio, and a remote controller. At the ground control station, the operator can send control signals to the flight control board via the remote controller and obtain flight status information from the Mission Planner software, which communicates with the 3DR telemetry radio device. For UAV systems, sensors such as accelerometers, barometers, and gyroscopes are used to monitor changes in the environment to maintain flight stability; then, the environmental information is sent back to the flight control board to adjust the flight attitude of the hexacopter. Figure 1 shows the UAV's system structure.

在本研究中，使用六旋翼无人机执行指定任务，例如，为了水产养殖目的收集传感器数据。本节详细介绍了用于深度学习训练的主要架构、传感器和计算机硬件配置。计算硬件系统包括机载计算机和离线训练计算机。基本的无人机系统由飞行控制板、电源、传感器、电机、3DR 遥测无线电和遥控器组成。在地面控制站，操作员可以通过遥控器向飞行控制板发送控制信号，并通过与 3DR 遥测无线电设备通信的 Mission Planner 软件获取飞行状态信息。对于无人机系统，加速度计、气压计和陀螺仪等传感器用于监测环境变化以保持飞行稳定性；然后，将环境信息发送回飞行控制板，以调整六旋翼飞行器的飞行姿态。图 1 显示了无人机的系统结构。

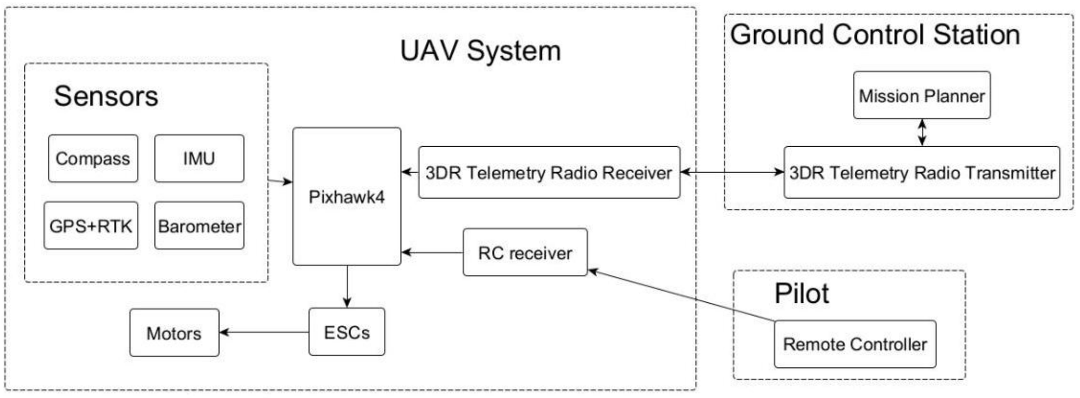


Figure 1. UAV system structure.

图 1。无人机系统结构。

The flight control board of the hexacopter is the Pixhawk 4, which is part of an independent open-source project. It has a more advanced 32-bit ARM Cortex ® M7 processor, 16 PWM/servo outputs, and others used to establish the flight status. The necessary sensors include a three-axis gyroscope, barometer, magnetometer, and accelerometer. The attitude of the hexacopter can be improved by the two sets of accelerometers, while other sensors can help the hexacopter to locate its position. In order to prevent positioning errors, the Pixhawk 4 can also have the compass and GPS module installed externally, away from the motor, electronic speed controller, and power supply lines.

六旋翼飞行器的飞行控制板是 Pixhawk 4，它是独立的开源项目的一部分。它具有更先进的 32 位 ARM Cortex ® M7 处理器、16 个 PWM/伺服输出等，用于建立飞行状态。必要的传感器包括三轴陀螺仪、气压计、磁力计和加速度计。通过两组加速度计可以提高六旋翼飞行器的姿态，而其他传感器可以帮助飞行器定位。为了防止定位错误，Pixhawk 4 还可以在外部安装指南针和 GPS 模块，远离电机、电子速度控制器和电源线。

The ground control station (GCS) software and the flight control board work together, and the appropriate firmware version can be selected and installed. The operator can easily obtain the flight information via the Mission Planner software, which helps the operator to understand the current rotorcraft status and record the flight data during the mission at the same time. This allows us to acquire information if the hexacopter had crashed or encountered other emergency situations. The NEO-M8N module is highly compatible; therefore, it is widely used in UAV positioning systems. GNSS can concurrently receive GPS, Galileo, GLONASS, and Beidou satellite signals, accessing them all at the same time. The safety switch of the drone hardware is also on the same module. In a favorable area without obscuration and interference, the NEO-M8N module, when combined with real-time kinematic (RTK) positioning technology, can receive data from up to 30 satellites; the horizontal dilution accuracy (HDOP) is often between 0.5 and 0.6, which means that the GPS is in good condition; that is, the positioning error (static) is less than 10 cm .

地面控制站 (GCS) 软件和飞行控制板协同工作，可以选择并安装适当的固件版本。操作员可以通过任务规划软件轻松获取飞行信息，这有助于操作员了解当前旋翼机状态，并在任务期间记录飞行数据。这使我们能够获取六旋翼无人机如果坠毁或遇到其他紧急情况的信息。NEO-M8N 模块高度兼容；因此，它被广泛应用于无人机定位系统中。GNSS 可以同时接收 GPS、伽利略、GLONASS 和北斗卫星信号，同时访问它们。无人机硬件的安全开关也位于同一模块上。在没有遮挡和干扰的有利区域，NEO-M8N 模块结合实时动态 (RTK) 定位技术，可以接收到多达 30 颗卫星的数据；水平稀释精度 (HDOP) 通常在 0.5 到 0.6 之间，这意味着 GPS 状态良好；也就是说，定位误差 (静态) 小于 10 cm 。

The accuracy of waypoints in path planning is very important. In this study, the RTK positioning technology is used. RTK base stations are set up in the experimental field to wait for their convergence and are then connected to the chip on the hexacopter, so that the error can be reduced to 10 cm . Figure 2 shows the ground station of the RTK. The working procedure of the proposed approach is shown in Figure 3.

路径规划中航点的准确性非常重要。在本研究中，使用了 RTK 定位技术。在实验场设立 RTK 基站，等待其收敛，然后连接到六旋翼机上的芯片，从而将误差减小到 10 cm 。图 2 显示了 RTK 的地面站。所提出方法的工作流程如图 3 所示。

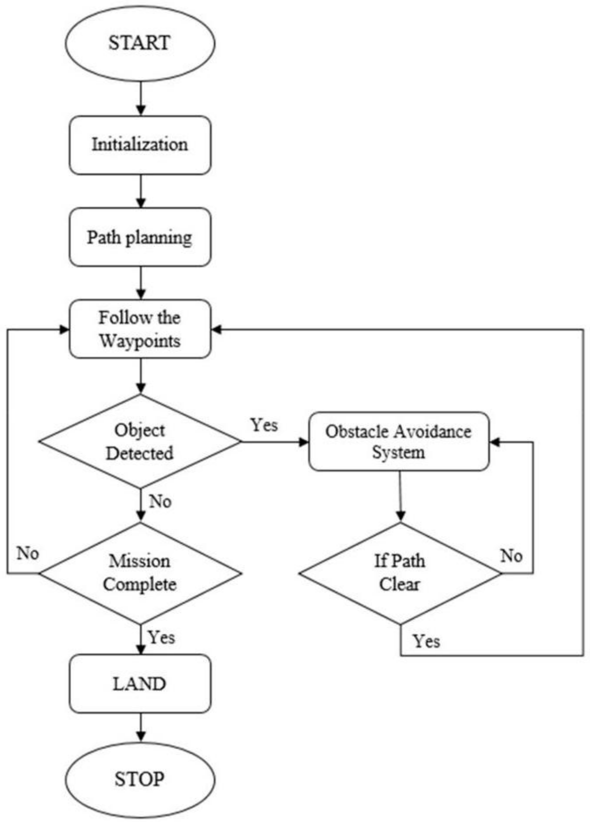


Figure 3. Working procedure of the proposed approach.
图 3. 所提出方法的工作流程。

In this study, we used a self-assembling hexacopter, which is mainly made from lightweight and strong carbon fiber. Compared with the quadcopter, the hexacopter utilizes a sensor that is more resistant to strong wind interference and offers greater force-bearing. The main body of the hexacopter is shown in Figure 4. The on-board computer is used for the path-planning experiment and obstacle-avoidance experiment. Ultrasonic sensors are set on the side of the hexacopter. The hexacopter is also used in aquaculture projects, considering that we can implement reinforcement learning algorithms in the future for hexacopter obstacle avoidance and other tasks that require a lot of computing to detect objects. For integration, we used the NVIDIA Jetson Xavier NX development kits and modules. The use of deep Q-learning in the virtual environment of Microsoft AirSim requires a great deal of computer resources; therefore, a graphics card with better graphics computing capabilities and sufficient memory, etc. is required. The sensor used for obstacle detection on the side of the UAV’s body is the ultrasonic ranging module, HC-SR04. The HC-SR04 has a ranging accuracy of 3 mm and can measure distances from 2 cm to 400 cm .

在本研究中，我们使用了一种自组装六旋翼无人机，该无人机主要由轻质且坚固的碳纤维制成。与四旋翼无人机相比，六旋翼无人机使用了一种更能抵抗强风干扰的传感器，并提供了更大的承重能力。六旋翼无人机的主体如图 4 所示。机载计算机用于路径规划实验和避障实验。在六旋翼无人机的侧面安装了超声波传感器。考虑到我们将来可以在六旋翼无人机上实施强化学习算法，用于避障和其他需要大量计算来检测物体的任务，六旋翼无人机也被用于水产养殖项目。在集成方面，我们使用了 NVIDIA Jetson Xavier NX 开发套件和模块。在 Microsoft AirSim 虚拟环境中使用深度 Q 学习需要大量的计算机资源；因此，需要具有更好的图形计算能力和足够内存的显卡等。用于无人机侧面障碍检测的传感器是超声波测距模块 HC-SR04。HC-SR04 的测距精度为 3 mm ，并且能够测量 2 cm 到 400 cm 的距离。

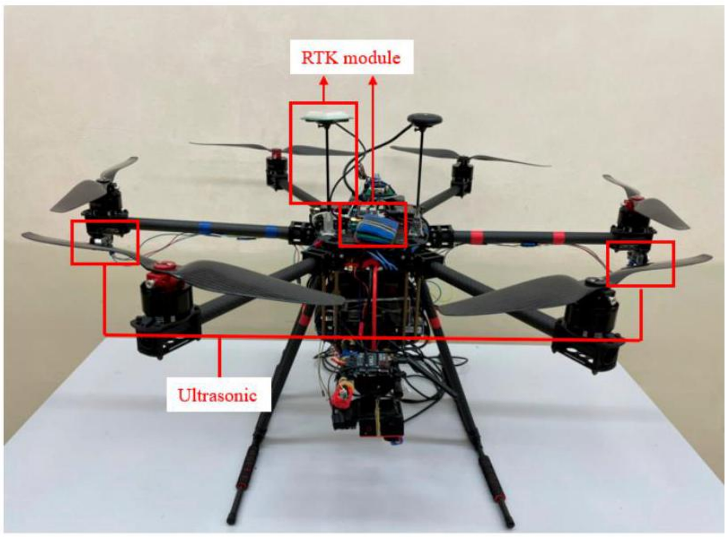


Figure 4. The hexacopter setup.
图 4. 六旋翼无人机设备。

3. Path Planning

3. 路径规划

One of the most complex problems that are widely explored in robotics concerns path planning. There are many algorithms that have been applied to path planning, such as the improved ACO [23], used to navigate in a maze-like environment, the ACO combined with chaos theory [24], used to find a path away from threatened areas, and the improved PSO [25], used to navigate a ship in the surrounding environment of an island. When the hexacopter performs tasks, there are often huge static obstacles in the execution area, such as mountains, buildings, no-fly zones (NFZ), etc. Pathfinding in a known environment is known as global path planning. The main purpose is to spend less time and reach the destination without passing through restricted areas. Compared with the traditional A* algorithm, the reinforcement learning algorithm can solve more complex problems. However, when the model corrects the error, the chance of the same error is very small. This technique is the first choice for long-term results. This method can collect information about the environment and interact with the specific environment of the problem, which makes this method of problem-solving flexible. The learning model of reinforcement learning is very similar to that of human learning. Therefore, this method will be closer to perfect expectations. In this section, all obstacle settings are treated as a priori information, which represents a known environment. Before the hexacopter takes off, we calculate the path and plan the best path. Using the known information, two search algorithms based on reinforcement learning are applied to generate optimal routes in this study. As a result, the hexacopter can fly through the given waypoints. The introduction of these two algorithms is given as follows.

机器人学中广泛探讨的最复杂问题之一涉及路径规划。已经有许多算法应用于路径规划，例如改进的蚁群算法 [23]，用于在迷宫式环境中导航，结合混沌理论的蚁群算法 [24]，用于寻找远离威胁区域的路径，以及改进的粒子群优化算法 [25]，用于在岛屿周边环境中导航船只。当六旋翼无人机执行任务时，执行区域中常常存在巨大的静态障碍物，如山脉、建筑物、禁飞区 (NFZ) 等。在已知环境中寻找路径被称为全局路径规划。其主要目的是花费更少的时间，在不经过限制区域的情况下到达目的地。与传统的 A* 算法相比，强化学习算法可以解决更复杂的问题。然而，当模型纠正错误时，再次犯同样错误的机会非常小。这种技术是长期结果的首选方法。这种方法可以收集有关环境的信息，并与问题的特定环境互动，这使得这种问题解决方法具有灵活性。强化学习的学习模型与人类学习模型非常相似。因此，这种方法将更接近完美的期望。在本节中，所有障碍物设置都被视为先验信息，代表了一个已知环境。在六旋翼无人机起飞前，我们计算路径并规划最佳路径。利用已知信息，本研究中应用了两种基于强化学习的搜索算法来生成最优路线。因此，六旋翼无人机能够飞越给定的航点。以下是对这两种算法的介绍。

3.1. Markov Decision Process (MDP)

3.1. 马尔可夫决策过程 (MDP)

In reinforcement learning, the three basic elements are state, action, and reward. These three elements are the messages that the agent and environment communicate with each other, but they still lack communication channels, as shown in Figure 5. This pipeline is created through the mathematical model of the Markov decision process (MDP) [26].

在强化学习中，三个基本元素是状态、动作和奖励。这三个元素是智能体和环境之间交流的信息，但它们仍然缺乏通信渠道，如图 5 所示。这个管道是通过马尔可夫决策过程 (MDP) 的数学模型创建的 [26]。

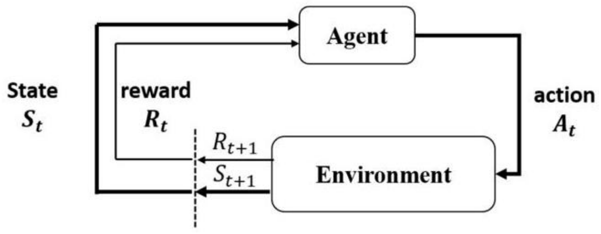


Figure 5. The interaction between the agent and the environment.
图 5. 智能体与环境的交互。

The agent interacts with the environment, as shown in Figure 5. The environment gives the state and reward at time t to the agent, and the agent then determines the behavior to be performed using this information. Then, the environment determines the state and reward for the next time through the behavior of the agent. All the processes of MDP are random and the next state transition is only related to the current state. At time t , action A_t is made according to S_t , and the state S_{t+1} at the next time, $t+1$ is uncertain. We call this environment transition probability the MDP dynamic. The equation is as follows:

如图 5 所示，智能体与环境交互。环境在时间 t 给智能体提供状态和奖励，智能体然后使用这些信息确定要执行的行为。接着，环境通过智能体的行为确定下一个时间点的状态和奖励。MDP 的所有过程都是随机的，下一个状态转换仅与当前状态相关。在时间 t ，根据 S_t 进行动作 A_t ，下一个时间点的状态 S_{t+1} ，$t+1$ 是不确定的。我们称这种环境转换概率为 MDP 动态。方程如下:

$$p\left(s', r \mid s, a\right) = \Pr\left(S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a\right). \tag{1}$$

After taking action a in the state s , the probability id that the next state is s' and the reward is r . The next-state transition of MDP is only related to the current state; we call this the Markov Property, which can be expressed using the following equation:

在状态 s 执行动作 a 后，下一个状态是 s' 且奖励是 r 的概率。MDP 的下一个状态转换仅与当前状态相关；我们称其为马尔可夫性质，可以用以下方程表示:

$$\Pr\left(S_n \mid S_{n-1}, S_{n-2}, \ldots, S_0\right) = \Pr\left(S_n \mid S_{n-1}\right). \tag{2}$$

The current state contains information about all the states experienced in the past. However, the probability that we are looking for can reject all the past states and focus only on the present state. This greatly reduces the amount of calculation necessary, and we can then use a simple iterative method to find the result. However, since not all reinforcement-learning problems satisfy the Markov property, we can assume that the Markov property is satisfied to achieve approximate results.

当前状态包含过去经历的所有状态的信息。然而，我们寻找的概率可以忽略所有过去的状态，只关注当前状态。这大大减少了必要的计算量，然后我们可以使用简单的迭代方法找到结果。但是，由于并非所有的强化学习问题都满足马尔可夫性质，我们可以假设满足马尔可夫性质以获得近似结果。

3.2. Bellman Equation

3.2. 贝尔曼方程

The Bellman equation [27] is also called the dynamic programming equation. Many common reinforcement learning algorithms are based on the Bellman equation, and the goal of reinforcement learning can be expressed as a value function. Calculating the solution of this value function can indirectly identify the best solution for reinforcement learning. However, the solution of the value function must be simplified by the Bellman equation. The Bellman equation is as follows:

贝尔曼方程 [27] 也被称为动态规划方程。许多常见的强化学习算法都是基于贝尔曼方程的，强化学习的目标可以表示为一个值函数。计算这个值函数的解可以间接地确定强化学习的最佳解。然而，值函数的解必须通过贝尔曼方程进行简化。贝尔曼方程如下:

$$v(s) = \mathbb{E}\left[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s\right] \tag{3}$$

where R_{t+1} represents the immediate reward values and γ is the discount factor $(0 \leq \gamma \leq 1)$. However, it is uncertain, when moving from state S_t to state S_{t+1} , if there will be an expectation symbol in the outermost layer. By observing the equation, we can see that $v(s)$ is composed of two parts. One is the immediate reward expectation of this state. According to the definition of the immediate reward, this will not be related to the next state. The other is the value expectation of the state in the next moment, which can be obtained according to the probability distribution of the state at the next moment. If S_{t+1} is expressed as any possible state at the next moment in state S_t , then the Bellman equation can be written as:

其中 R_{t+1} 表示即时奖励值，γ 是折扣因子 $(0 \leq \gamma \leq 1)$ 。然而，在从状态 S_t 转移到状态 S_{t+1} 时，不确定是否在最外层会有一个期望符号。通过观察方程，我们可以看到 $v(s)$ 由两部分组成。一个是这个状态的即时奖励期望。根据即时奖励的定义，这将与下一个状态无关。另一个是下一时刻状态的值期望，可以根据下一时刻状态的概率分布获得。如果 S_{t+1} 表示在状态 S_t 下的下一个时刻的任何可能状态，那么贝尔曼方程可以写成:

$$v(s) = R_s + \gamma \sum_{S_{t+1} \in S} P_{S_t S_{t+1}} v(S_{t+1}) . \tag{4}$$

The value function of each state is calculated iteratively through the Bellman equation. Below, we give an example showing the calculation of the value function of a state to help with understanding the process, as shown in Figure 6.

每个状态的值函数是通过贝尔曼方程迭代计算的。下面，我们给出了一个计算状态值函数的示例，以帮助理解这个过程，如图 6 所示。

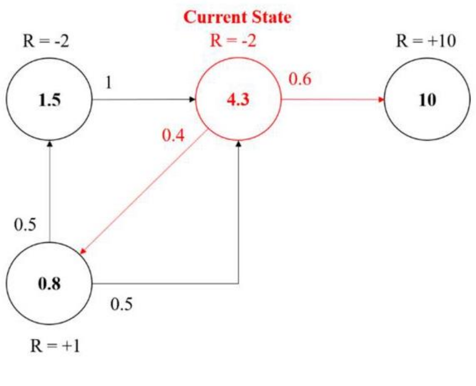


Figure 6. Example of the calculation of value function

图 6. 值函数计算的示例

After substituting the value into the calculation of Equation (4), the value of the current state can be obtained, as shown in Equation (5):

在将值代入方程 (4) 的计算后，可以得到当前状态的值，如方程 (5) 所示:

$$4.3 = -2 + 0.6 \times 10 + 0.4 \times 0.8. \tag{5}$$

However, when the current state is 4.3, the subsequent state value function is 0.8 and 10. In fact, these values can be arbitrarily initialized and updated through learning, thereby functioning in the same way as the weight parameters in neural networks. They are initialized arbitrarily at the beginning of the process and are then updated backward through loss.

然而，当当前状态为 4.3 时，后续状态值函数为 0.8 和 10。实际上，这些值可以通过学习任意初始化并更新，从而像神经网络的权重参数一样发挥作用。它们在过程开始时任意初始化，然后通过损失反向更新。

3.3. Policy

3.3. 策略

In both the Q-learning algorithm and the SARSA algorithm, a particular policy is required for action selection. In this study, two policies were applied and are introduced here. The first is the most intuitive, the greedy policy [28]. Assuming the situation encountered below is shown in Figure 7, there are 4 directions to choose from in this state, and the highest 0.9Q -value is selected according to the Q -table; this is the greedy policy mentioned above, to which we add random probability. In this way, during the training process, there is a chance that the agent of ε will choose a random direction, so that the model has the opportunity to learn new experiences. This is the ε -greedy policy [29].

在 Q-learning 算法和 SARSA 算法中，都需要一个特定的策略来进行动作选择。在本研究中，应用了两种策略，并在此介绍。第一种是最直观的，即贪心策略 [28]。假设遇到的以下情况如图 7 所示，在该状态下有 4 个方向可以选择，根据 0.9Q -值表选择最高的 Q -值；这就是上面提到的贪心策略，我们为其添加随机概率。这样，在训练过程中，ε 的代理有机会选择一个随机方向，从而使模型有机会学习新的经验。这就是 ε -贪心策略 [29]。

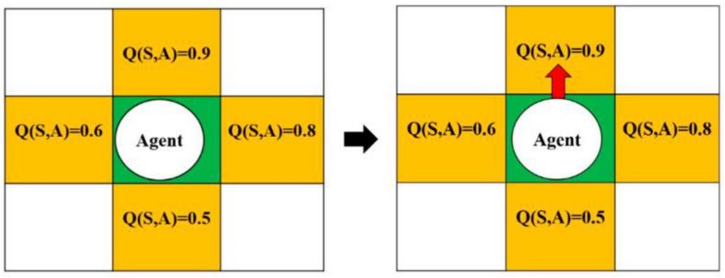


Figure 7. Operation of the greedy policy.

图 7. 贪心策略的操作。

In Figure 8, there is a probability that the agent in this state will randomly choose the direction so that the agent has the opportunity to learn new experiences and will not always choose the same action.

在图 8 中，该状态下的代理有一定的概率随机选择方向，这样代理就有机会学习新的经验，而不会总是选择相同的动作。

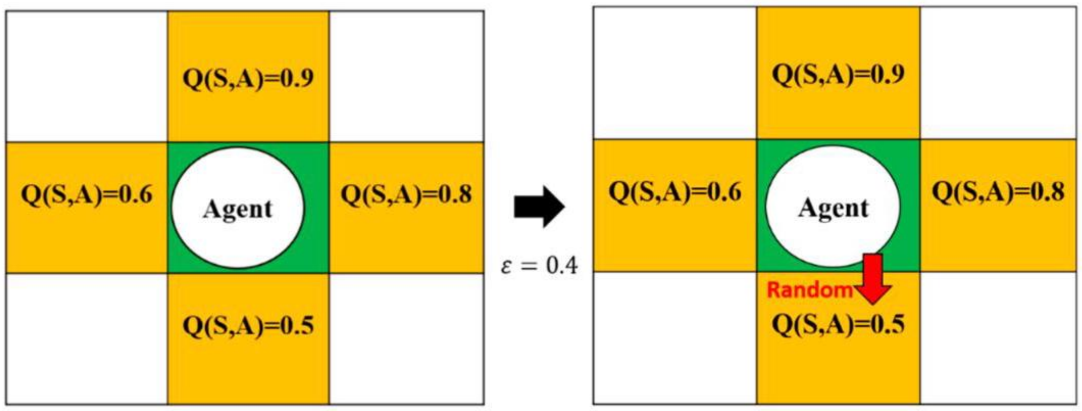


Figure 8. Operation of the ε - greedy policy.
图 8. ε -贪心策略的操作。

3.4. Q-Learning Algorithm

3.4. Q-Learning 算法

Q-learning is a time difference (TD) method and is also a classic method in reinforcement learning [30]. The Q-learning algorithm uses the newly obtained reward and the original Q-value to update the current Q-value, which process is also called value iteration Then, it creates a Q-table and updates the Q-table with the rewards of each action, as shown in Algorithm 1. The Q-learning algorithm procedure is shown in Figure 9. The inputs of this algorithm are state S (the current position coordinates on the map), action A (the direction of movement), and reward R (the reward from the current position to the next position). The outputs are the values of the Q-table.

Q 学习是一种时差 (TD) 方法，也是强化学习中的经典方法 [30]。Q 学习算法使用新获得的奖励和原始的 Q 值来更新当前的 Q 值，这一过程也被称为值迭代。然后，它创建一个 Q 表，并使用每个动作的奖励来更新 Q 表，如算法 1 所示。Q 学习算法的过程如图 9 所示。该算法的输入是状态 S (地图上的当前位置坐标)、动作 A (移动方向) 和奖励 R (从当前位置到下一个位置的奖励)。输出是 Q 表的值。

Algorithm 1 Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$
算法 1 Q 学习 (异策略 TD 控制) 用于估计 $\pi \approx \pi_*$
Initialize $Q(s,a), \forall s \in S, a \in \mathcal{A}(s)$, arbitrarily, and $Q($ terminal state, $\cdot) = 0$
初始化 $Q(s,a), \forall s \in S, a \in \mathcal{A}(s)$ ，任意地，和 $Q($ terminal state, $\cdot) = 0$
Repeat (for each episode):
重复 (对于每个剧集):
Initialize S
初始化 S
Repeat (for each step of episode):
重复 (对于剧集中的每一步):
Choose A from S using policy derived from Q (e.g., ϵ -greedy)
从 S 中选择 A ，使用由 Q 导出的策略 (例如，ϵ -贪婪)
Take action A , observe R, S'
执行动作 A ，观察 R, S'

$$Q(S,A) \leftarrow Q(S,A) + \alpha \left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$$

$S \leftarrow S'$;
until S is terminal
直到 S 是终止状态

The Q-learning algorithm is off-policy. This different policy means that the action policy and the evaluation policy are not the same. The action policy in Q-learning is the ε -greedy policy, whereas the policy to update the Q-table is the greedy policy. The equation for calculating the Q-value is as follows, and is based on the Bellman equation:

Q 学习算法是异策略的。这种不同的策略意味着动作策略和评估策略不是相同的。Q 学习中的动作策略是 ε -贪婪策略，而更新 Q 表的策略是贪婪策略。计算 Q 值的公式如下，基于贝尔曼方程:

$$Q^{new}(s_t, a_t) \leftarrow (1-\alpha) \cdot Q(s_t, a_t) + \alpha \cdot \left(r_t + \gamma \cdot \max_a Q(s_{t+1}, a)\right) \tag{6}$$

where r_t represents the reward values that are obtained from state S_t to state S_{t+1}, α is the learning rate $(0 < \alpha \leq 1)$, and γ is the discount factor $(0 \leq \gamma \leq 1)$. When the γ -value is larger, the agent pays more attention to the long-term rewards to be obtained in the future. Finally, the calculated Q-value is stored in the Q-table. The flowchart is shown in Figure 10.

其中 r_t 表示从状态 S_t 到状态 S_{t+1}, α 获得的奖励值, $(0 < \alpha \leq 1)$ 是学习率, γ 是折扣因子 $(0 \leq \gamma \leq 1)$ 。当 γ -值较大时，智能体更多地关注未来可能获得的长远奖励。最后，计算出的 Q 值被存储在 Q 表中。流程图如图 10 所示。

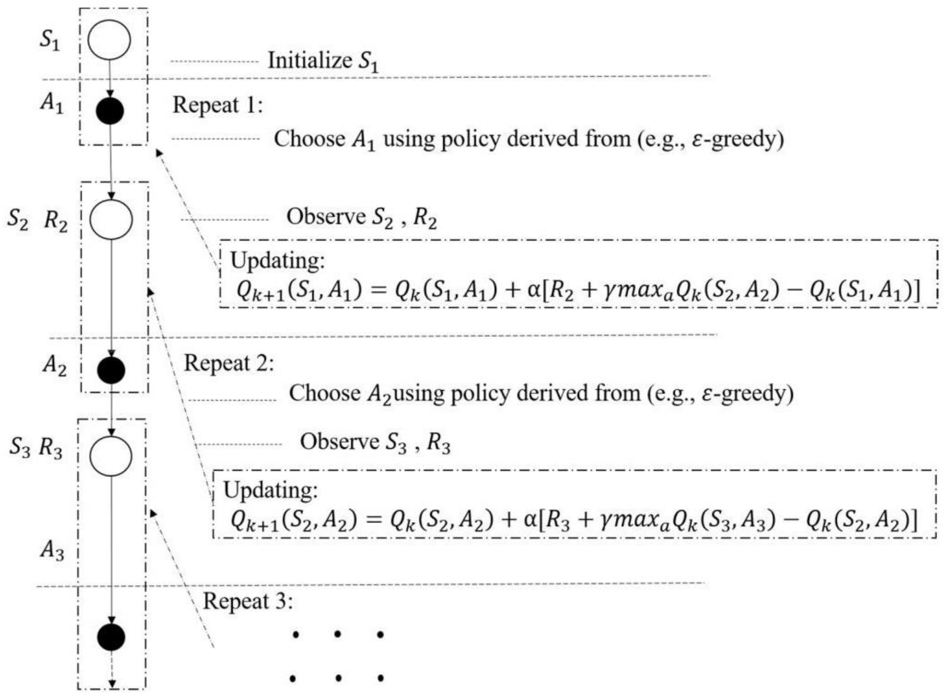

S_1
Initialize S_1
A_1
Repeat 1:
Choose A_1 using policy derived from (e.g., ε-greedy)
S_2 R_2
Observe S_2 , R_2
Updating:
$Q_{k+1}(S_1,A_1) = Q_k(S_1,A_1) + \alpha[R_2 + \gamma max_a Q_k(S_2,A_2) - Q_k(S_1,A_1)]$
A_2
Repeat 2:
Choose A_2using policy derived from (e.g., ε-greedy)
S_3 R_3
Observe S_3 , R_3
Updating:
$Q_{k+1}(S_2,A_2) = Q_k(S_2,A_2) + \alpha[R_3 + \gamma max_a Q_k(S_3,A_3) - Q_k(S_2,A_2)]$
A_3
Repeat 3:

Figure 9. Q-learning algorithm procedure.
图 9. Q 学习算法流程。

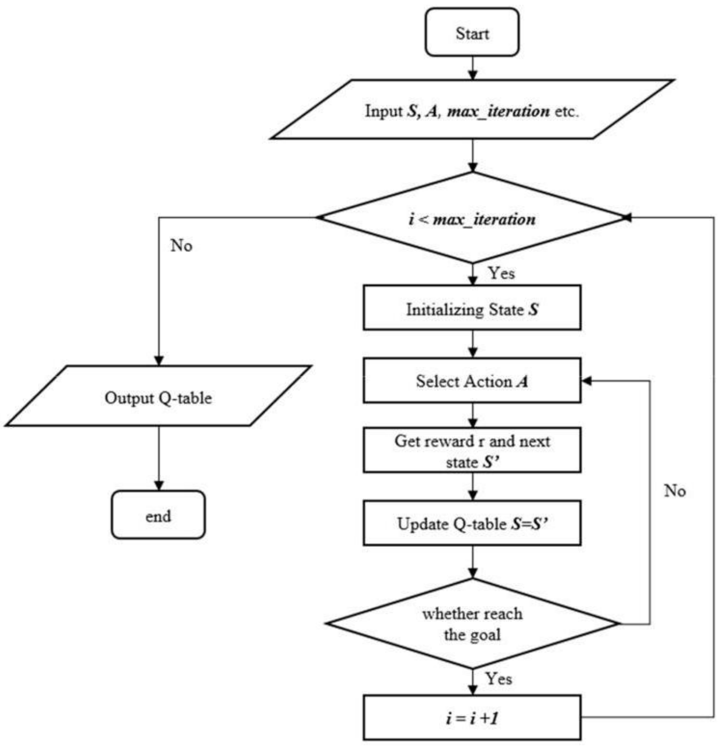

Start
Input ***S, A, max_iteration*** etc.
i < max_iteration
No
Yes
Initializing State ***S***
Select Action ***A***
Get reward r and next state ***S'***
Update Q-table ***S=S'***
No
whether reach the goal
Yes
i = i +1
Output Q-table
end

Figure 10. Flowchart of the Q-learning algorithm.
图 10. Q 学习算法的流程图。

3.5. Reward

3.5. 奖励

In the process of human learning and growth, parents will teach their children what things should be done, or what things are dangerous and should not be done. The right actions will be rewarded, and the dangerous actions will be punished. In Q-learning, it is also necessary to set reward and punishment rules. In this way, when iterating and updating the Q-value, the result will be closer to expectations. When we initialize the reward, the reward rules are as shown in Equation (7). Finally, in order to reach the goal with the least number of steps, each step will be taken as -1 . The schematic diagram of the reward and punishment rules is shown in Figure 11. The rules are flexible and need to be adjusted according to the problem. In other words, the rules of reward and punishment depend on the problem. Q-learning will thus maximize the overall reward as much as possible. Therefore, there is more than one route in Figure 12 that will yield the highest

在人类学习和成长的过程中，父母会教他们的孩子应该做什么事情，或者哪些事情是危险的、不应该做的事情。正确的行动将会得到奖励，而危险的行动将会受到惩罚。在 Q 学习中，同样需要设置奖励和惩罚规则。这样，在迭代和更新 Q 值时，结果将更接近预期。当我们初始化奖励时，奖励规则如方程 (7) 所示。最终，为了以最少的步数达到目标，每一步都将设置为 -1。奖励和惩罚规则的示意图如图 11 所示。这些规则是灵活的，需要根据问题进行调整。换句话说，奖励和惩罚规则取决于问题本身。Q 学习将尽可能地最大化总体奖励。因此，在图 12 中，有不止一条路径能够获得最高分。

score.

分数。

$$\text{Reward} = \begin{cases} -1 & \text{every step} \\ -200 & \text{enter the gray blocks} \\ +100 & \text{reach the goal} \end{cases} \tag{7}$$

-1	-1	-1	-1	Goal +100
-1	-200	-200	-1	-1
-1	-200	-200	-1	-1
START 0	-1	-1	-1	-1

-1	-1	-1	-1	目标 +100
-1	-200	-200	-1	-1
-1	-200	-200	-1	-1
开始 0	-1	-1	-1	-1

Figure 11. The schematic diagram of the reward and punishment rules.
图 11. 奖励和惩罚规则的示意图。

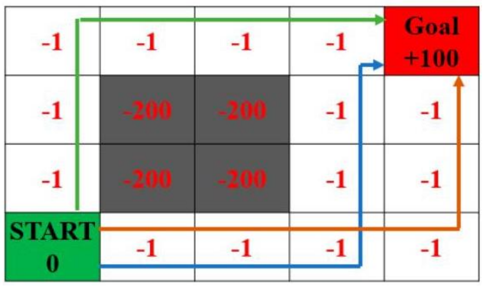


Figure 12. The routes in the reward and punishment rules.
图 12. 奖励和惩罚规则中的路径。

The ultimate goal of Q-Learning is to obtain feedback, meaning that the sum of all rewards on the trajectory during the training process needs to be saved. Therefore, the Q-table of the two-dimensional table is designed to store the $Q(S, A)$, which can store the expected maximum reward value for each action in each state in the future. In this way, we can know the best action for each state. The table is known as the Q-table and is shown in Figure 13.

Q 学习的最终目标是获得反馈，这意味着在训练过程中轨迹上所有奖励的总和需要被保存。因此，设计了二维表格的 Q 表来存储 $Q(S,A)$ ，它可以存储每个状态下每个动作在未来预期的最大奖励值。这样，我们可以知道每个状态的最佳动作。这个表格被称为 Q 表，如图 13 所示。

Grid Map Q-Table

网格地图 Q 表

(0,0)	(1,0)	(2,0)	(3,0)	(4,0)
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)

position	Up	Down	Left	Right
(0,0)				
(0,1)				
(0,2)				
(0,3)				
(0,4)				
(0,5)				
(0,6)				
(x, y)				

位置	上	下	左	右
(0,0)				
(0,1)				
(0,2)				
(0,3)				
(0,4)				
(0,5)				
(0,6)				
(x, y)				

Figure 13. Grid map stored in a Q-table.

图 13。存储在 Q 表中的网格地图。

There are four actions in the Q-table in the grid map (up, down, left, and right actions). The row represents the state, while the value of each cell will be the maximum reward value expected in the future for a specific state and action.

在网格地图的 Q 表中，有四个动作 (上、下、左、右动作)。行表示状态，而每个单元格的值将是特定状态和动作在未来预期的最大奖励值。

3.6. State-Action-Reward-State-Action (SARSA) Algorithm

3.6. 状态-动作-奖励-状态-动作 (SARSA) 算法

SARSA is also a time difference (TD) method and is a classic method in reinforcement learning [31]. The SARSA algorithm is similar to the Q-learning algorithm; the main difference is the way in which it updates the Q-value, as shown in Algorithm 2. The SARSA algorithm procedure is shown in Figure 14. The inputs of this algorithm are state S (the current position coordinates on the map), action A (the moving direction), and reward R (the reward for moving from the current position to the next position). The outputs are the values of the Q-table.

SARSA 也是一种时间差 (TD) 方法，是强化学习中的经典方法 [31]。SARSA 算法与 Q 学习算法相似；主要区别在于它更新 Q 值的方式，如算法 2 所示。SARSA 算法流程如图 14 所示。该算法的输入是状态 S (地图上的当前位置坐标)、动作 A (移动方向) 和奖励 R (从当前位置移动到下一个位置的奖励)。输出是 Q 表的值。

Algorithm 2 Sarsa (on-policy TD control) for estimating $Q \approx q_*$

算法 2 Sarsa(策略 TD 控制) 用于估计 $Q \approx q_*$

Initialize $Q(s,a), \forall s \in S, a \in \mathcal{A}(s)$, arbitrarily, and $Q($ terminal state, $\cdot) = 0$

初始化 $Q(s,a), \forall s \in S, a \in \mathcal{A}(s)$ ，任意，和 $Q($ terminal state, $\cdot) = 0$

```
Repeat (for each episode):
重复 (对每个剧集):
Initialize $S$
初始化 $S$
Choose $A$ from $S$ using policy derived from $Q$ (e.g., $\epsilon$ -greedy)
从 $S$ 中选择 $A$ ，使用由 $Q$ 演导的策略 (例如，$\epsilon$ -贪婪)
Repeat (for each step of episode):
重复 (对于每个剧集中的步骤):
Take action $A$ , observe $R, S'$
执行动作 $A$ ，观察 $R, S'$
Choose $A'$ from $S'$ using policy derived from $Q$ (e.g., $\epsilon$ -greedy)
从 $S'$ 中选择 $A'$ ，使用从 $Q$ 演绎出的策略 (例如，$\epsilon$ -贪婪)
$Q(S, A) \leftarrow Q(S, A) + \alpha\left[R + \gamma Q\left(S', A'\right) - Q(S, A)\right]$
$S \leftarrow S'; A \leftarrow A'$;
until $S$ is terminal
直到 $S$ 是终止状态
```

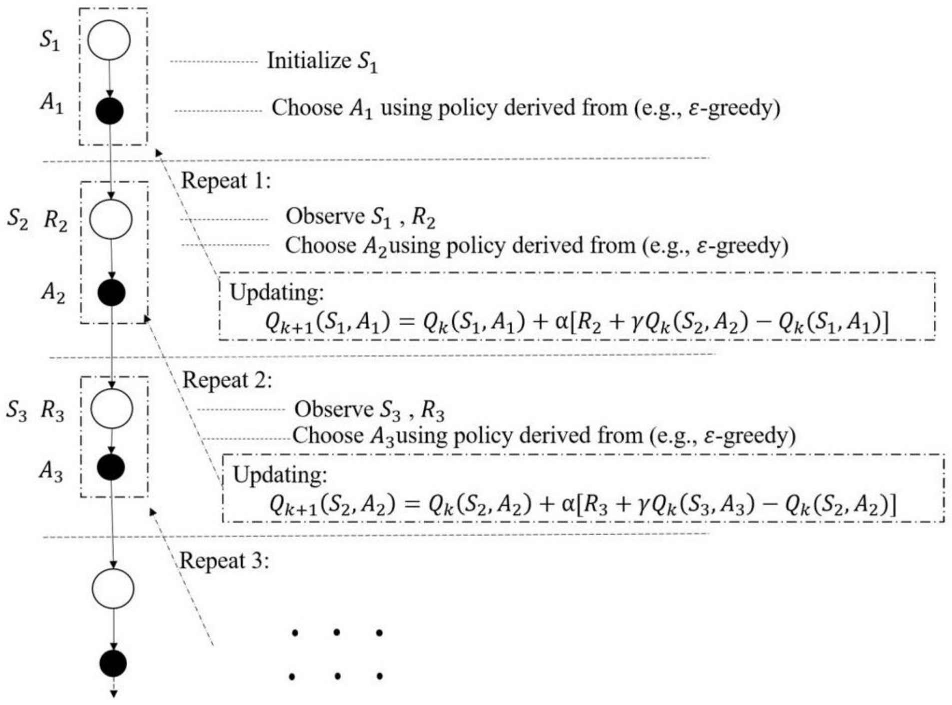

S_1
A_1
Initialize S_1
Choose A_1 using policy derived from (e.g., ε-greedy)
Repeat 1:
S_2 R_2
A_2
Observe S_1 , R_2
Choose A_2using policy derived from (e.g., ε-greedy)
Updating:
$Q_{k+1}(S_1, A_1) = Q_k(S_1, A_1) + \alpha[R_2 + \gamma Q_k(S_2, A_2) - Q_k(S_1, A_1)]$
Repeat 2:
S_3 R_3
A_3
Observe S_3 , R_3
Choose A_3using policy derived from (e.g., ε-greedy)
Updating:
$Q_{k+1}(S_2, A_2) = Q_k(S_2, A_2) + \alpha[R_3 + \gamma Q_k(S_3, A_3) - Q_k(S_2, A_2)]$
Repeat 3:
• • •
• • •

Figure 14. The SARSA algorithm procedure.
图 14. SARSA 算法流程。

This algorithm mainly incorporates the five elements of reinforcement learning: state (S), action(A), reward(R), discount factor, γ , and exploration rate, ε , and uses the optimal action value function to find the best path. It can be seen that in the updated formula of $Q(S, A)$, the on-policy update method is adopted, and both the action policy and the update policy are an ε -greedy policy. This means that the SARSA algorithm will directly adopt the policy that is currently considered the best in the current state. The equation for calculating the Q-value is as follows:

此算法主要融合了强化学习的五个要素: 状态 (S)、动作 (A)、奖励 (R)、折扣因子 γ 和探索率 ε ，并使用最优动作价值函数来寻找最佳路径。可以看出，在 $Q(S, A)$ 的更新公式中，采用的是同策略更新方法，动作策略和更新策略都是 ε -贪婪策略。这意味着 SARSA 算法将直接采用在当前状态下认为最佳的策略。计算 Q 值的公式如下:

$$Q^{\text{new}}\left(s_t, a_t\right) \leftarrow (1-\alpha) \cdot Q\left(s_t, a_t\right) + \alpha \cdot \left(r_t + \gamma \cdot Q\left(s_{t+1}, a\right)\right) \tag{8}$$

where r_t represents the reward values that are obtained from state S_t to state S_{t+1}, α is the learning rate $(0 < \alpha \leq 1), \gamma$ is the discount factor $(0 \leq \gamma \leq 1)$. Finally, the Q-value is updated and stored in the Q-table via an iterative calculation.

其中 r_t 表示从状态 S_t 到状态 S_{t+1}, α 获得的奖励值，$(0 < \alpha \leq 1)$, γ 是学习率，$(0 \leq \gamma \leq 1)$ 是折扣因子。最后，通过迭代计算更新 Q 值并将其存储在 Q 表中。

3.7. SARSA Algorithm, Compared with the Q-Learning Algorithm

3.7. SARSA 算法，与 Q 学习算法的比较

Comparing the two algorithms in the grid world shows that there will be different paths due to the different Q-value update methods. An example is shown in Figure 15.

在网格世界中比较两种算法可以看出，由于 Q 值更新方法的不同，将会产生不同的路径。图 15 中展示了一个示例。

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
start	-100							Goal

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1
开始	-100							目标

Figure 15. Example in the grid-world environment.

图 15. 网格世界环境中的示例。

In this environment, one point is lost for every square moved. When stepping into the black area (obstacle), the algorithm will lose one hundred points and return to the starting point. Finally, it will learn the path with the highest score to reach the goal. The method uses both the SARSA algorithm and Q-learning algorithm for learning. The results are shown in Figures 16 and 17; the dashed line is the optimal route of each algorithm.

在此环境中，每移动一个方块就会失去一个积分。当步入黑色区域 (障碍物) 时，算法将失去一百个积分并返回起点。最终，它将学会分数最高的路径以达到目标。该方法同时使用 SARSA 算法和 Q 学习算法进行学习。结果展示在图 16 和图 17 中；虚线是每个算法的最优路径。

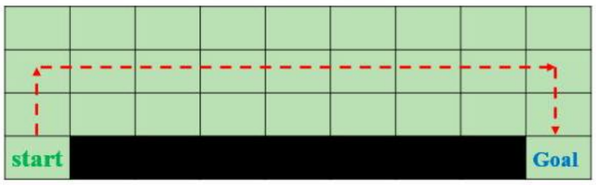


Figure 16. The SARSA algorithm route.

图 16。SARSA 算法的路径。

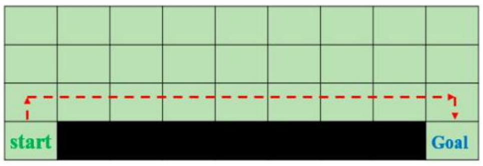


Figure 17. The Q-learning algorithm route.

图 17。Q 学习算法的路径。

The action policy of the Q-learning algorithm is based on the ε -greedy policy; only the globally optimal action is considered in the Q-value update method. The SARSA algorithm can only find a suboptimal path, which is intuitively safer. Because the SARSA algorithm has the same action policy and evaluation policy, it considers the possibilities of all actions (ε -greedy). When it is close to the black area, there is a certain probability that it will choose to take a step toward the black area, so it is less likely to choose the path around the black area. Although the Q-learning algorithm has the ability to learn the globally optimal path, its convergence is slow, while the SARSA algorithm has a lower learning effect than the Q-learning algorithm, but its convergence is faster and simpler. Therefore, we need to consider the different issues.

Q 学习算法的动作策略基于 ε -贪婪策略；在 Q 值更新方法中只考虑全局最优动作。SARSA 算法只能找到一条次优路径，这在直观上更安全。因为 SARSA 算法具有相同的动作策略和评估策略，它考虑了所有动作的可能性 (ε -贪婪)。当它接近黑色区域时，有一定的概率会选择向黑色区域迈出一步，因此它不太可能选择绕着黑色区域走的路径。尽管 Q 学习算法有能力学习全局最优路径，但其收敛速度较慢，而 SARSA 算法的学习效果低于 Q 学习算法，但其收敛速度更快且更简单。因此，我们需要考虑不同的问题。

4. Obstacle Avoidance

4. 避障

In recent years, drones have become popular in both the consumer and professional fields. Therefore, great research attention has been paid to the obstacle-avoidance function of UAVs. In order to ensure flight safety, although drones have GPS signals, there are still some small static or dynamic obstacles, such as trees, walls, and birds. None of these obstacles can be ascertained from the GPS data. In order to solve this problem, the obstacle avoidance system first needs accurate positioning information of the UAV, relative to the obstacle. Several methods have been proposed, such as the use of deep reinforcement learning, based on obstacle avoidance training and testing in a virtual environment [18,32], or the use of depth cameras with computer vision detection to avoid obstacles [33,34]. After obtaining information about the environmental obstacles, the UAV will adopt the corresponding avoidance direction and keep the flight stable. The main purpose of the current research is to collect water-quality information regarding the cage net in the ocean. Therefore, several long obstacles of sufficient height have been placed to simulate the situation when encountering sails at sea. The ground also has up- and down-slopes to simulate the environment of sea waves. The following describes the system architecture and the details of training in a virtual environment.

近年来，无人机在消费和专业领域都变得非常流行。因此，研究者们对无人机的避障功能给予了大量的研究关注。为了确保飞行安全，尽管无人机拥有 GPS 信号，但仍然存在一些无法通过 GPS 数据确定的较小的静态或动态障碍物，例如树木、墙壁和鸟类。为了解决这个问题，避障系统首先需要获取无人机相对于障碍物的准确位置信息。已经提出了几种方法，例如基于虚拟环境中的避障训练和测试使用深度强化学习 [18,32]，或者使用深度相机结合计算机视觉检测来避障 [33,34]。在获取到环境障碍物信息后，无人机将采取相应的避障方向并保持飞行的稳定。当前研究的主要目的是收集海洋中笼网的水质信息。因此，放置了一些足够高的长障碍物来模拟海上遇到帆船的情况。地面也设置了上下坡来模拟海浪环境。以下描述了系统架构和在虚拟环境中训练的细节。

In the previous section, we mentioned the Q-learning algorithm. First, we suppose that when a complex problem is encountered, the Q-table needs to store very large quantities of states and actions. This will slow down the training speed and reduce the effect. Obstacle avoidance is a complex problem of this type. Therefore, a neural network is used for the calculation of the Q-value, known as the deep Q-learning network (DQN). The relationship of the DQN with the environment is shown in Figure 18. The inputs of the deep neural network are the depth images from the UAV's side. The outputs of the deep neural network are the Q-values.

在上一节中，我们提到了 Q 学习算法。首先，我们假设当遇到一个复杂问题时，Q 表需要存储大量的状态和动作，这将减慢训练速度并降低效果。避障就是这类复杂问题之一。因此，使用神经网络来计算 Q 值，这就是所谓的深度 Q 学习网络 (DQN)。DQN 与环境的关系如图 18 所示。深度神经网络的输入是来自无人机侧面的深度图像。深度神经网络的输出是 Q 值。

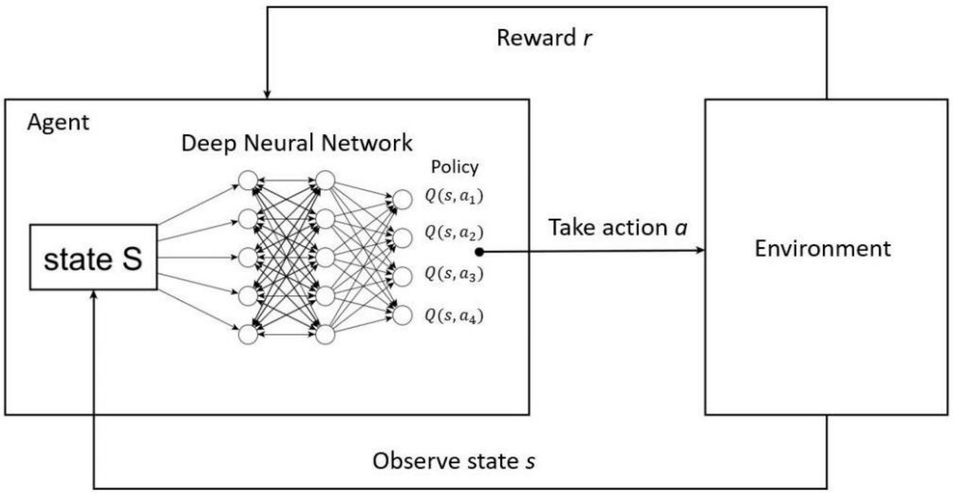


Figure 18. Interaction between the DQN and the environment.
图 18. DQN 与环境之间的交互。

4.1. Deep Q-Learning Algorithm

4.1. 深度 Q 学习算法

The deep Q-learning algorithm was originally designed for a brick-and-mortar game [35]; it is a method for passing Q-learning through the approximate value function of a neural network. It has achieved results that surpass human-level players in the Atari 2600 game; the algorithm is also suitable for the problem of avoiding obstacles. DQN has two models with the same structure; one is a Q-network and the other is a target Q-network. The parameters of the target Q-network will remain for a set period of time, the purpose of which is to avoid algorithm divergence. The DQN algorithm will update the Q-network parameter value to the target Q-network after a certain period of time. The DQN algorithm diagram is shown in Figure 19.

深度 Q 学习算法最初是为砖块游戏 [35] 设计的；它是一种通过神经网络的近似价值函数传递 Q 学习的方法。在 Atari 2600 游戏中，它取得了超越人类水平玩家的成果；该算法也适用于避障问题。DQN 有两个结构相同的模型；一个是 Q 网络，另一个是目标 Q 网络。目标 Q 网络的参数将保持一段时间不变，目的是为了避免算法发散。DQN 算法将在一定时间后更新 Q 网络参数值到目标 Q 网络。DQN 算法图示如图 19 所示。

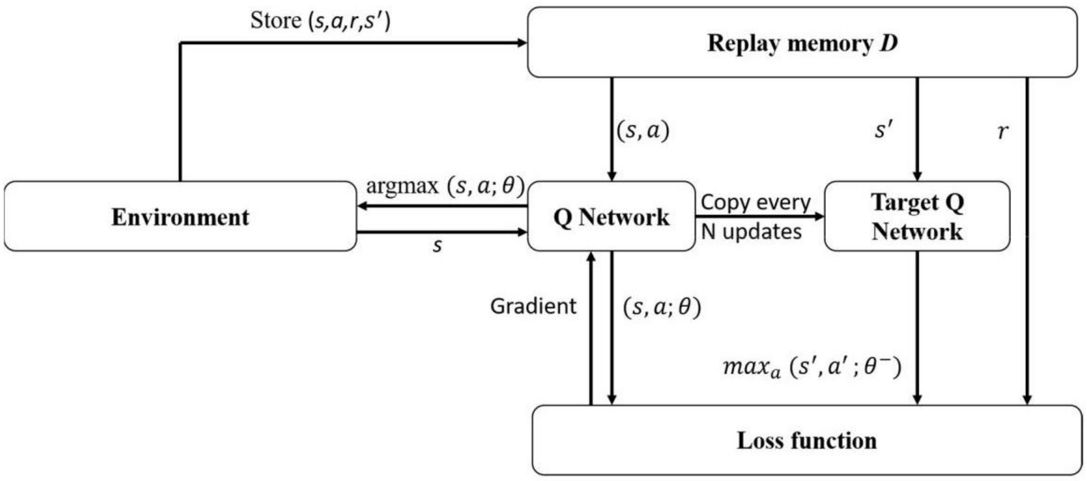


Figure 19. The DQN algorithm diagram.
图 19. DQN 算法图示。

The core of DQL is the error between two neural networks. According to the target network, we should select the action with the largest Q-value and the corresponding $\max\mathrm{Q}\,(a')$. We then multiply the discount factor, γ , by this maximum value and add r , to establish the target value of the neural network. The square error between the target value of the target Q-network and $Q\,(a)$ of the Q-network is the error of the deep neural network. Finally, we use the error calculated by the loss function to train the neural network. The inputs of this algorithm are state s (image pixel information), action a (moving direction), and reward r (the reward for moving from the current position to the next position). The outputs are the values of the Q-table. The DQL algorithm is shown as Algorithm 3 [35].

DQL 的核心是两个神经网络之间的误差。根据目标网络，我们应该选择具有最大 Q 值的动作及其对应的 $\max\mathrm{Q}\,(a')$ 。然后我们将折扣因子 γ 乘以这个最大值并加上 r ，以建立神经网络的目标值。目标 Q 网络的预测值与 Q 网络的 $Q\,(a)$ 之间的平方误差是深度神经网络的误差。最后，我们使用损失函数计算出的误差来训练神经网络。该算法的输入是状态 s (图像像素信息)、动作 a (移动方向) 和奖励 r (从当前位置移动到下一个位置的奖励)。输出是 Q 表的值。DQL 算法如算法 3 [35] 所示。

```
Algorithm 3 Deep Q-learning with Experience Replay
算法 3 经验回放深度 Q 学习
Initialize replay memory $D$ to capacity $N$
初始化回放记忆 $D$ 到容量 $N$
Initialize action-value function $Q$ with random weights
使用随机权重初始化动作价值函数 $Q$
For episode $= 1, M$ do
对于每一幕 $= 1, M$ 执行
    Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi\,(s_1)$ for $t = 1, T$ do
    初始化序列 $s_1 = \{x_1\}$ 和预处理序列 $\phi_1 = \phi\,(s_1)$ 对于 $t = 1, T$ 执行
        With probability $\epsilon$ select a random action select a random action $a_t$
        以 $\epsilon$ 的概率选择一个随机动作选择一个随机动作 $a_t$
        otherwise select $a_t = \max_a Q^*\left(\phi\,(s_t)\,, a; \theta\right)$
        否则选择 $a_t = \max_a Q^*\left(\phi\,(s_t)\,, a; \theta\right)$
        Execute action $a_t$ in emulator and observe reward $r_t$ and image $x_{t+1}$
        在模拟器中执行动作 $a_t$ 并观察奖励 $r_t$ 和图像 $x_{t+1}$
        Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi\,(s_{t+1})$
        设置 $s_{t+1} = s_t, a_t, x_{t+1}$ 并预处理 $\phi_{t+1} = \phi\,(s_{t+1})$
        Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $D$
        将转换 $(\phi_t, a_t, r_t, \phi_{t+1})$ 存储在 $D$ 中
        Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $D$
        从 $D$ 中随机抽样一个过渡的小批量 $(\phi_j, a_j, r_j, \phi_{j+1})$
        Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q\,(\phi_{j+1}, a'; \theta) & \text{for non } - \text{ terminal } \phi_{j+1} \end{cases}$
        设置 $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q\,(\phi_{j+1}, a'; \theta) & \text{for non } - \text{ terminal } \phi_{j+1} \end{cases}$
        Perform a gradient descent step on $\left(y_j - Q\,(\phi_j, a_j; \theta)\right)^2$ according to Equation (3)
        根据 (3) 式对 $\left(y_j - Q\,(\phi_j, a_j; \theta)\right)^2$ 执行梯度下降步骤
    end for
    结束循环
end for
结束循环
```

The reason for experience replay is that the deep neural network, as a supervised learning model, requires the data to meet independent and identical distributions. The samples obtained by the Q-learning algorithm are related to the data both before and after. In order to break the correlation between these datasets, the experience replay method breaks this correlation via storage and sampling. The following equations are the equation of the gradient descent method and the equation for the loss function:

经验回放的原因在于，深度神经网络作为一种监督学习模型，需要数据满足独立同分布。由 Q 学习算法获得的样本与之前和之后的数据相关。为了打破这些数据集之间的相关性，经验回放方法通过存储和采样来打破这种相关性。以下方程是梯度下降方法的方程和损失函数的方程:

$$g_l = \left(r + \gamma \max_{a'} Q\left(s', a'; \theta_l^-\right) - Q\left(s, a; \theta_l\right)\right) \nabla_\theta Q\left(s, a; \theta\right) \tag{9}$$

$$L_t\left(\theta_t\right)=E\left[\left(r+\gamma \max_{a'} Q\left(s', a'; \theta_t^{-}\right)-Q\left(s, a, ; \theta_t\right)\right)^2\right] \tag{10}$$

The stochastic gradient descent method can be used to optimize the weight, minimize the loss function, and gradually approach the optimal Q function to satisfy Bellman’s equation. If the loss function is 0, it can be considered that the approximation function has reached the optimal solution.

可以使用随机梯度下降方法来优化权重，最小化损失函数，并逐渐接近最优 Q 函数以满足贝尔曼方程。如果损失函数为 0，可以认为近似函数已经达到了最优解。

4.2. Microsoft AirSim Environment

4.2. Microsoft AirSim 环境

AirSim [36] is an open-source platform for drones, cars, and other simulators that were built based on Unreal Engine 4. It can also offer a cross-platform for simulation control through the PX4 flight controller. AirSim can interact with vehicles or drones in the simulation program via programming languages, such as Python, and can use these APIs to retrieve images and obtain the status to control the vehicle. The use of reinforcement learning methods for UAV obstacle avoidance training requires multiple collisions. Training in the real world will inevitably take a great deal of time and cost, and it will also cause doubts about flight safety. Therefore, most of the reinforcement learning training will be carried out in a virtual environment. In this study, we presented the obstacle avoidance results within the AirSim virtual environment. The environment from the AirSim is shown in Figure 20.

AirSim [36] 是一个基于虚幻引擎 4 构建的开源平台，用于无人机、汽车和其他模拟器。它还可以通过 PX4 飞行控制器提供跨平台的模拟控制。AirSim 可以通过编程语言 (如 Python) 与模拟程序中的车辆或无人机进行交互，并可以使用这些 API 获取图像并获得车辆的状态以进行控制。使用强化学习方法进行无人机避障训练需要多次碰撞。在实际世界中的训练不可避免地需要大量时间和成本，并且还会对飞行安全产生疑问。因此，大部分的强化学习训练将在虚拟环境中进行。在本研究中，我们展示了在 AirSim 虚拟环境中的避障结果。AirSim 的环境如图 20 所示。

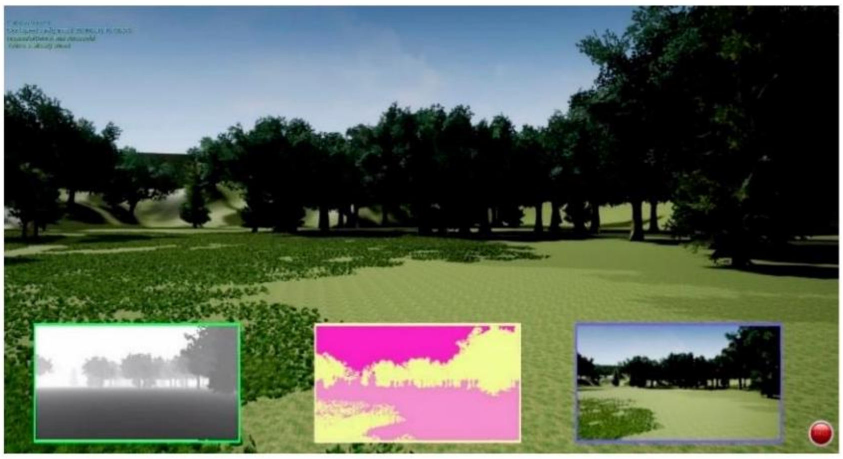


Figure 20. AirSim environment screenshot.
图 20. AirSim 环境截图。

4.3. Obstacle Avoidance System in the AirSim Environment

4.3. AirSim 环境中的避障系统

A particular variable, D , known as replay memory, is used in the DQL algorithm to store the current state, current action, current action reward, and next state. When training the neural network, we randomly select batch-size samples from variable D to train the neural network. In this study, there are an RGB camera and a depth camera in front of the virtual drone. In the case of drones, the RGB image is the current state feature, and the depth image is used to obtain the distance information regarding the surrounding obstacles and to obtain the rewards for the current action selection. The DQL obstacle avoidance flow chart is shown in Figure 21.

特定的变量 D ，称为重放记忆，在 DQL 算法中用于存储当前状态、当前动作、当前动作奖励和下一个状态。在训练神经网络时，我们从变量 D 中随机选择批量样本训练神经网络。在本研究中，虚拟无人机前有一个 RGB 相机和一个深度相机。在无人机的案例中，RGB 图像是当前状态特征，深度图像用于获取关于周围障碍物的距离信息，并为当前动作选择获取奖励。DQL 避障流程图如图 21 所示。

We use the backpropagation method, by means of the error between the target value and the current value, to train the neural network. After that, we output the best predicted Q-value for action selection. Figure 22 shows the neural network architecture. The image adopts $227 \times 227 \times 3$ as the input; it passes through the convolutional network layer for feature extraction, then passes through the fully connected layer for classification, and finally outputs 25Q -values for each action selection.

我们使用反向传播方法，通过目标值与当前值之间的误差来训练神经网络。之后，我们输出每个动作选择的最佳预测 Q 值。图 22 展示了神经网络架构。图像采用 $227 \times 227 \times 3$ 作为输入；它通过卷积网络层进行特征提取，然后通过全连接层进行分类，最后为每个动作选择输出 25Q 值。

The action space is a 5×5 size, as shown in Figure 23. There are 25 action options; the flight action of the drone is calculated via angle equations, then we output the pitch and yaw control commands to the UAV. θ_l and ϕ_j represent the moving angles for the horizontal and vertical directions. These angle equations are calculated as follows:

动作空间大小为 5×5 ，如图 23 所示。有 25 个动作选项；无人机的飞行动作通过角度方程计算，然后我们输出俯仰和偏航控制命令到无人机。θ_l 和 ϕ_j 分别代表水平和垂直方向的移动角度。这些角度方程计算如下:

$$\theta_l = \left(\frac{FOV_h}{N^2} \times \frac{i - \left(N^2 - 1\right)}{2} \right) \tag{11}$$

$$\phi_j = \left(\frac{FOV_v}{N^2} \times \frac{i - \left(N^2 - 1\right)}{2} \right) \tag{12}$$

where i and $j \in \left\{0, 1, \ldots, \frac{N^2-1}{2}\right\}$ are the locations, as shown in Figure 23. The size of the action space is $N \times N.FOV_h$ and FOV_v are information on the horizontal and vertical attitudes, respectively, taken from the drone.

其中 i 和 $j \in \left\{0, 1, \ldots, \frac{N^2-1}{2}\right\}$ 是位置，如图 23 所示。动作空间的大小为 $N \times N.FOV_h$ ； FOV_v 分别是无人机水平姿态和垂直姿态的信息。

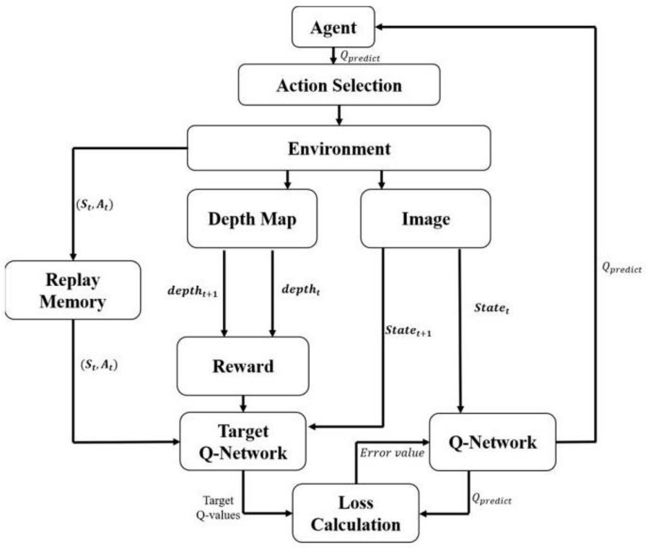


Figure 21. DQL obstacle avoidance flow chart.

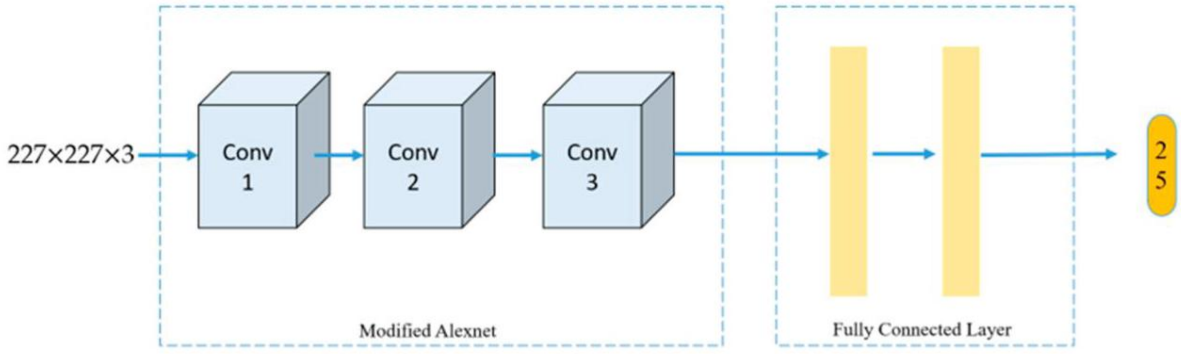


Figure 22. Network architecture.

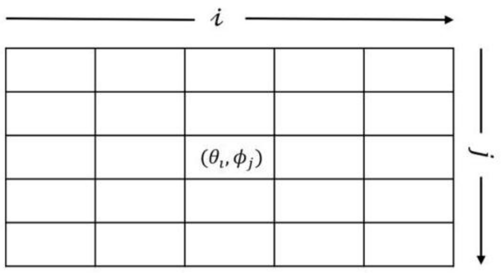


Figure 23. The 5×5 action space.
图 23。5×5 动作空间。

4.4. Ultrasonic Sensor

4.4. 超声波传感器

The principle behind employing the HC-SR04 to measure distance is to use an I/O trigger to send a high-level pulse signal of more than $10\mu s$, so that8periodic 40kHz square waves are automatically generated in the module; we then wait for the receiver to detect the signal after the bumped object rebounds it, as shown in Figure 24.

使用 HC-SR04 测量距离的原理是利用 I/O 触发发送超过 $10\mu s$ 的高电平脉冲信号，从而在模块中自动生成周期性 40kHz 的方波；然后等待接收器检测到被障碍物反弹回来的信号，如图 24 所示。

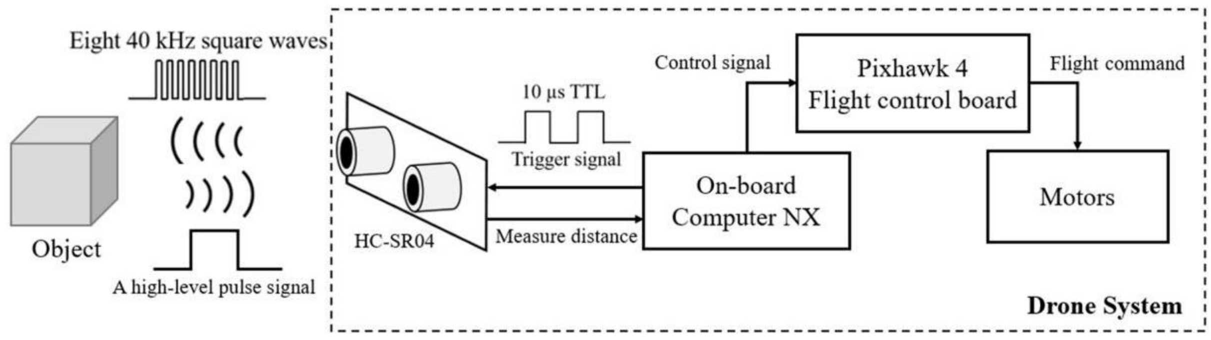


Figure 24. Illustration of the ultrasonic process.

图 24. 超声波处理过程的示意图。

The equation of the sound velocity in air, with the temperature in degrees Celsius, T_c , can be written as follows:

空气中声速的方程，以摄氏度表示的温度 T_c ，可以写成如下形式:

$$v_{\text{ sound in air }} \cong 331.4 + 0.607 \times T_c \left(\frac{m}{s}\right) \tag{13}$$

Therefore, the distance to the object can be estimated according to the time when the sound wave travels back. We assume that the temperature is 25 degrees Celsius and that the air condition of the environment is dry; after calculating the sound wave velocity equation, which is mentioned above, we can achieve 346.5 m/s or 0.03465 cm/μs , which means that the time the sound wave takes per centimeter is about 28.86 microseconds. Then, according to the width of the high-level pulse signal, $w_{\text{ recieve }}$, which is proportional to the detection distance, the distance, d_{obj} , from the object is given as follows:

因此，可以根据声波返回的时间来估算到物体的距离。我们假设温度为 25 摄氏度，环境中的空气条件为干燥；在计算上述提到的声波速度方程后，我们可以得到 346.5 m/s 或 0.03465 cm/μs ，这意味着声波每厘米所需的时间约为 28.86 微秒。然后，根据高电平脉冲信号的宽度 $w_{\text{ recieve }}$ ，它与检测距离成正比，从物体到传感器的距离 d_{obj} 可以表示如下:

$$\frac{d_{obj} = \frac{1}{2} \times w_{\text{ recieve }}}{28.86} \tag{14}$$

Here, in order to make the distance measured by the ultrasonic sensor module more stable, the received measurement value is filtered. This filtering method was first proposed by the authors of [37].

这里，为了使超声波传感器模块测量的距离更稳定，对接收到的测量值进行了滤波。这种滤波方法最初是由文献 [37] 的作者提出的。

The steps are as follows.

步骤如下。

Step 1. Set the maximum tolerance as 0.3 cm .

步骤 1. 设置最大容差为 0.3 cm 。

Step 2. Send impulse signal and get five data each time from the ultrasonic sensor, then add in the temporary data list.

步骤 2. 发送脉冲信号，每次从超声波传感器获取五个数据，然后将数据添加到临时数据列表中。

Step 3. Sort the received data and calculate the median value of the data list.

步骤 3. 对接收到的数据进行排序，并计算数据列表的中位数。

Step 4. If there is a previous value, go to step 5, or go to step 6 .

步骤 4. 如果有前一个值，则转到步骤 5，否则转到步骤 6。

Step 5. Calculate the difference in the current value and the previous value; if the difference is bigger than the maximum tolerance, then the output is equal to the previous result, plus or minus the maximum tolerance.

第 5 步。计算当前值与先前值的差异；如果差异大于最大容差，则输出等于先前结果，加上或减去最大容差。

Step 6. Record the current value as the previous value and output the result and go to step 2.

第 6 步。将当前值记录为先前值，输出结果并转到第 2 步。

Comparing the original measurement distance with the filtered measurement distance, it is obviously improved, and the value can be stabilized for different distances. The different distances are compared, and the error and the time consumption are calculated. The data are shown in Table 1. However, the ultrasonic sensor module has a chance of becoming stuck; the measure values were stopped at the same value. The steps for the solution are presented as follows:

比较原始测量距离与滤波后的测量距离，显然有所改进，且该值可以在不同距离下稳定。比较不同距离，并计算误差和时间消耗。数据如表 1 所示。然而，超声波传感器模块有卡住的可能；测量值停止在同一数值。解决方案的步骤如下:

Step 1. Set to enter the protection mode when the output value is continuous and the same.

第 1 步。当输出值连续且相同时，设置为进入保护模式。

Step 2. Save 30 values to the temporary data list.

第 2 步。将 30 个值保存到临时数据列表中。

Step 3. Compare the first 29 items of data with the last 29 items of data in the temporary data list.

第 3 步。比较临时数据列表中前 29 个数据项与最后 29 个数据项。

Step 4. If the values are the same, close and restart the measurement.

第 4 步。如果值相同，关闭并重新开始测量。

Table 1. The HC-SR04 sensor, with a filter test.

表 1。HC-SR04 传感器，带滤波测试。

Dist real (cm)	Distest (cm)	Error (cm)	Error Percentage	Cost Time (s)
35	34.97	0.03	0.08%	0.0205
60	59.88	0.12	0.20%	0.0311
75	75.69	0.69	0.92%	0.0363
100	100.45	0.45	0.45%	0.0429
200	198.79	1.21	0.605%	0.0812
250	246.42	3.58	1.432%	0.0965
300	297.28	2.72	0.906%	0.1203
350	346.58	3.42	0.977%	0.1299
400	394.56	5.44	1.36%	0.1483
450	443.42	6.58	1.462%	0.1712

实际距离 (cm)	测试距离 (cm)	误差 (cm)	误差百分比	成本时间 (秒)
35	34.97	0.03	0.08%	0.0205
60	59.88	0.12	0.20%	0.0311
75	75.69	0.69	0.92%	0.0363
100	100.45	0.45	0.45%	0.0429
200	198.79	1.21	0.605%	0.0812
250	246.42	3.58	1.432%	0.0965
300	297.28	2.72	0.906%	0.1203
350	346.58	3.42	0.977%	0.1299
400	394.56	5.44	1.36%	0.1483
450	443.42	6.58	1.462%	0.1712

Finally, after stabilizing the measured value of the ultrasonic sensor, the UAV must perform an avoiding distance maneuver, d_{avoid} , in the opposite direction and then output different drone speeds, V_{drone} , according to the measured distance, d_{measure} . The ultrasonic dodge rules are set as follows:

最后，在稳定超声波传感器的测量值后，无人机必须执行避障距离机动，d_{avoid} ，朝相反方向，并根据测量距离输出不同的无人机速度，V_{drone} ，d_{measure} 。超声波避障规则设置如下:

$$\begin{cases} 250 < d_{\text{measure}} \leq 300 & V_{\text{drone}} = 0.5\frac{m}{s} \\ 150 < d_{\text{measure}} \leq 250 & V_{\text{drone}} = 1\frac{m}{s} \\ 0 < d_{\text{measure}} \leq 150 & V_{\text{drone}} = 3\frac{m}{s} \end{cases} \tag{15}$$

$$d_{\text{avoid}} = 300 - d_{\text{measure}} \tag{16}$$

5. Experimental Results

5. 实验结果

This section is divided into three parts. The first is used to compare the path-planning part in the grid world and use a better algorithm to fly in the real world. The second is an experimental flight with ultrasonic-assisted obstacle avoidance in the real world. The last part is intended to demonstrate the dodge test in a simulated forest environment after the reinforcement learning training is completed.

本节分为三个部分。第一部分用于比较在网格世界中的路径规划部分，并使用更好的算法在现实世界中飞行。第二部分是在现实世界中使用超声波辅助避障的实验飞行。最后一部分旨在演示在完成强化学习训练后在模拟森林环境中的避障测试。

5.1. Path Planning

5.1. 路径规划

Both SARSA and Q-learning are algorithms based on MDP. The two are very similar, and so they are often compared. In Experiment 1, we observed the path difference of the two algorithms in the same grid world, as shown in Figure 25.

SARSA 和 Q 学习都是基于 MDP 的算法。两者非常相似，因此经常被比较。在实验 1 中，我们观察到两种算法在相同网格世界中的路径差异，如图 25 所示。

As can be seen from the above figures, the path of the Q-learning algorithm is significantly better than that of the SARSA algorithm. In Figure 26a, it can be seen that the reward part of the Q-learning algorithm is obviously higher and is higher with respect to iteration, and the number of steps in the exploration target gradually converges. Conversely, the SARSA algorithm shows no significant reduction in the number of steps to explore the target and may fall into a local optimum, as shown in Figure 26b. Although the reward part has a high score, it is not stable. In Figure 24, the reward points after the iteration are gradually accumulated, and the reward point of the last iteration is the sum of the maximum value and the minimum value. Figure 27 shows the Q-table after training in Environment 1. In the process of updating the Q-value, the Q-learning algorithm will select the action with the largest Q-value among all the actions in the next state. The SARSA algorithm is more intuitive in terms of choosing the best action that is currently considered; the Q-learning algorithm will try to maximize the overall benefit. Therefore, it is more suitable for exploring this type of path-planning problem, which only has starting-point and end-point information, with unknown environmental obstacle information. The path length and exploration time are shown in Table 2.

从上述图中可以看出，Q 学习算法的路径明显优于 SARSA 算法。在图 26a 中，可以看出 Q 学习算法的奖励部分明显更高，并且随着迭代次数的增加而更高，探索目标的步数逐渐收敛。相反，SARSA 算法在探索目标的步数上没有显著减少，并且可能会陷入局部最优，如图 26b 所示。尽管奖励部分得分较高，但并不稳定。在图 24 中，迭代后的奖励点逐渐累积，最后一次迭代的奖励点是最大值和最小值的总和。图 27 显示了在环境 1 中训练后的 Q 表。在更新 Q 值的过程中，Q 学习算法将在下一个状态的所有动作中选择 Q 值最大的动作。SARSA 算法在选择当前认为的最佳动作方面更为直观；Q 学习算法将尝试最大化整体收益。因此，它更适合于探索这种类型的路径规划问题，该问题只有起点和终点信息，环境障碍信息未知。路径长度和探索时间如表 2 所示。

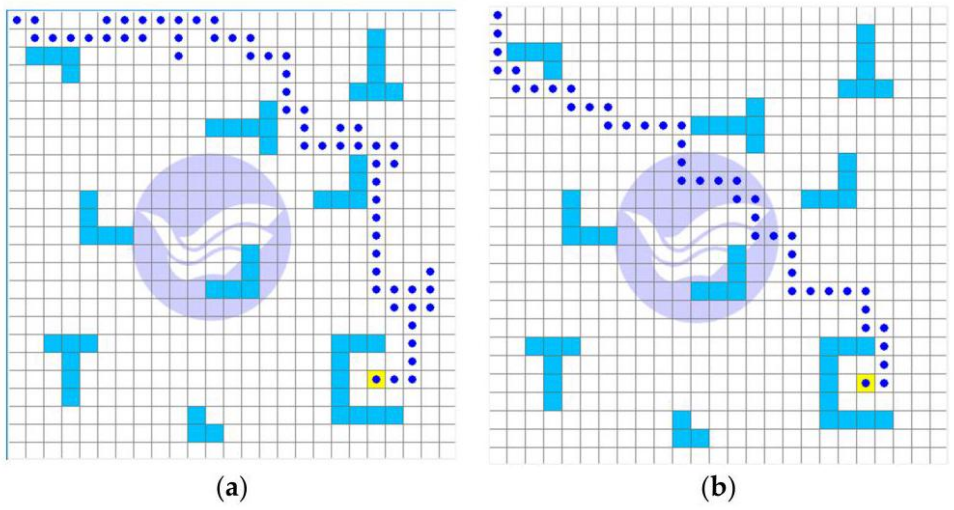


Figure 25. The experiment in Environment 1: (a) SARSA algorithm, (b) Q-learning algorithm.
图 25。环境 1 中的实验:(a) SARSA 算法，(b) Q 学习算法。

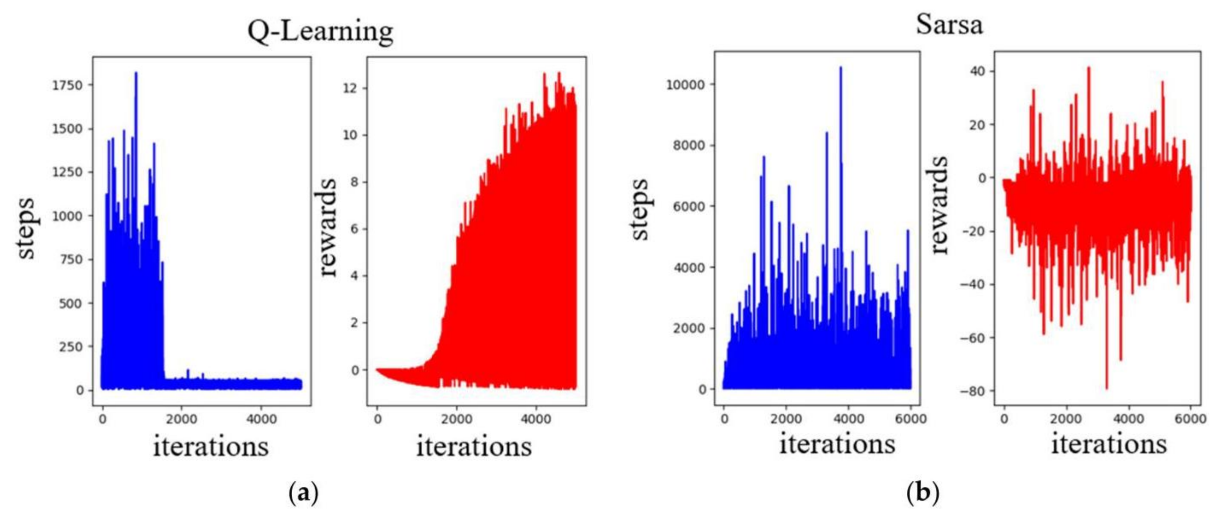


Figure 26. Training results of Q-Learning and SARSA in Environment 1, (a) is Q-Learning training result, (b) is SARSA training result.
图 26。环境 1 中 Q 学习和 SARSA 的训练结果，(a) 是 Q 学习的训练结果，(b) 是 SARSA 的训练结果。
Table 2. The results compared for Environment 1.
表 2. 环境 1 的结果比较。

Environment 1	SARSA	Q-Learning
path length	62 steps	42 steps
exploration time	32 min	15 min

环境 1	SARSA	Q-学习
路径长度	62 步	42 步
探索时间	32 分钟	15 分钟

In the second experiment, the obstacles in the grid world have been changed. The central part is blocked by new obstacles and the path becomes narrow just before the target point. The path of the SARSA algorithm is shown in Figure 28a, and the path of the Q-learning algorithm is shown in Figure 28b. The path of the Q-learning algorithm is still better than that of the SARSA algorithm.
在第二次实验中，网格世界中的障碍物已经改变。中心部分被新的障碍物阻挡，目标点前的路径变得狭窄。SARSA 算法的路径如图 28a 所示，Q 学习算法的路径如图 28b 所示。Q 学习算法的路径仍然优于 SARSA 算法。
Q-learning
Q 学习
SARSA

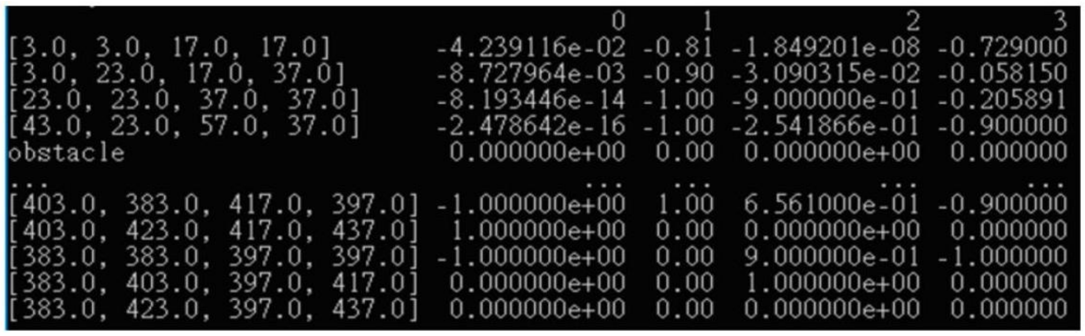

```
                                          0      1              2         3
[3.0, 3.0, 17.0, 17.0]         -4.239116e-02 -0.81 -1.849201e-08 -0.729000
[3.0, 23.0, 17.0, 37.0]        -8.727964e-03 -0.90 -3.090315e-02 -0.058150
[23.0, 23.0, 37.0, 37.0]       -8.193446e-14 -1.00 -9.000000e-01 -0.205891
[43.0, 23.0, 57.0, 37.0]       -2.478642e-16 -1.00 -2.541866e-01 -0.900000
obstacle                        0.000000e+00  0.00  0.000000e+00  0.000000
...                                      ...   ...           ...       ...
[403.0, 383.0, 417.0, 397.0] -1.000000e+00  1.00  6.561000e-01 -0.900000
[403.0, 423.0, 417.0, 437.0]  1.000000e+00  0.00  0.000000e+00  0.000000
[383.0, 383.0, 397.0, 397.0] -1.000000e+00  0.00  9.000000e-01 -1.000000
[383.0, 403.0, 397.0, 417.0]  0.000000e+00  0.00  1.000000e+00  0.000000
[383.0, 423.0, 397.0, 437.0]  0.000000e+00  0.00  0.000000e+00  0.000000
```

Figure 27. The Q-table for Q-learning and SARSA in Environment 1.
图 27. 环境 1 中 Q 学习与 SARSA 的 Q 表。

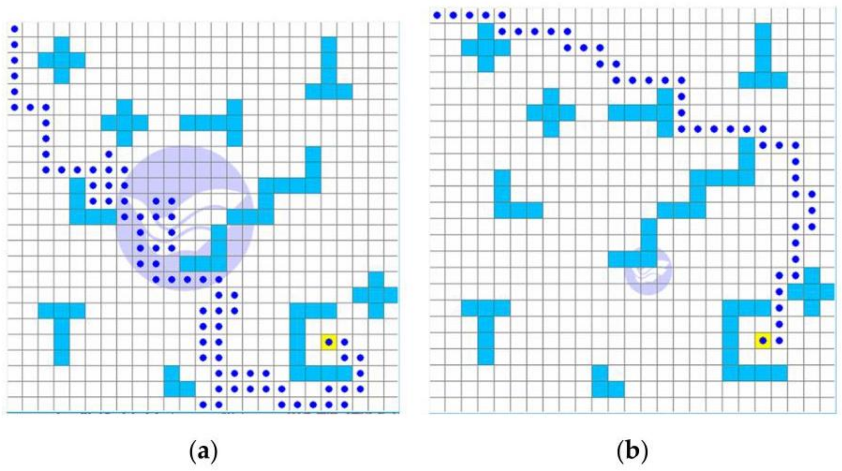


Figure 28. The experiment in Environment 2: (a) SARSA algorithm, (b) Q-learning algorithm.
图 28. 环境 2 的实验:(a) SARSA 算法，(b) Q 学习算法。

More obstacles are in place in Environment 2; therefore, the task of finding the best path becomes more difficult. In terms of rewards, the Q-learning algorithm still maintains the maximum benefit for exploration, and the difference from the SARSA algorithm can be seen more clearly. Both algorithms took a lot of time to complete the exploration, but the results of the Q-learning algorithm still converged at the end of the simulation. The training results of Q-learning and SARSA are shown in Figure 29. The same reason applies as in the previous environment result; the reward points after the iteration have accumulated, and the sum of the highest score and the lowest score is the score for each iteration. Figure 30 shows the Q-table after training in Environment 2. The path length and exploration time are shown in Table 3.

环境 2 中设置了更多障碍物；因此，寻找最佳路径的任务变得更加困难。在奖励方面，Q 学习算法仍然为探索保持最大利益，与 SARSA 算法的差异可以更清楚地看到。两种算法都花了很长时间来完成探索，但 Q 学习算法的结果仍然在模拟结束时收敛。Q 学习和 SARSA 的训练结果如图 29 所示。与之前环境结果相同的原因；迭代后的奖励点已经累积，最高分和最低分的总和是每次迭代的分数。图 30 显示了环境 2 训练后的 Q 表。路径长度和探索时间如表 3 所示。

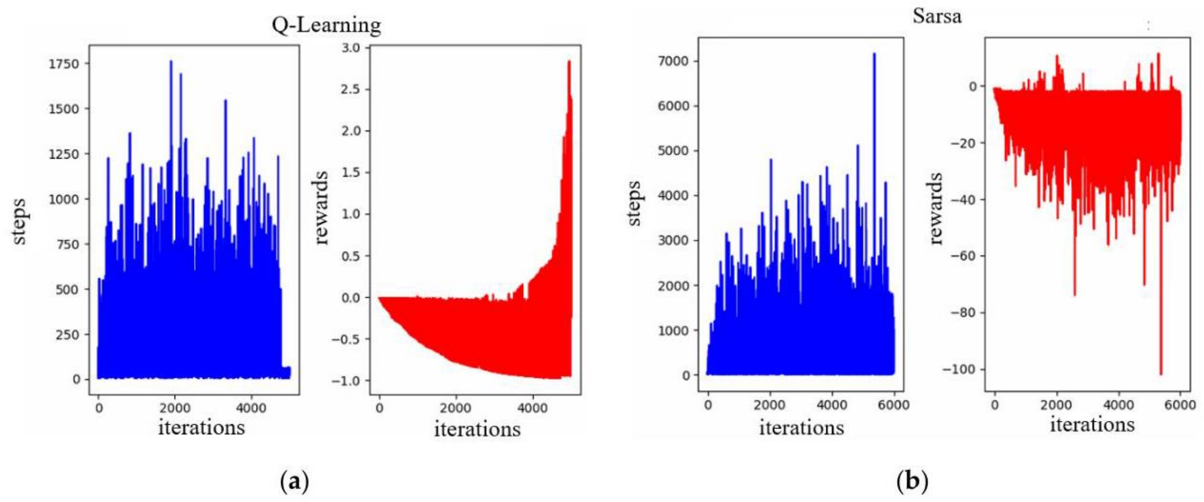


Figure 29. Training results of Q-learning and SARSA in Environment 2.
图 29. 环境 2 中 Q 学习和 SARSA 的训练结果。

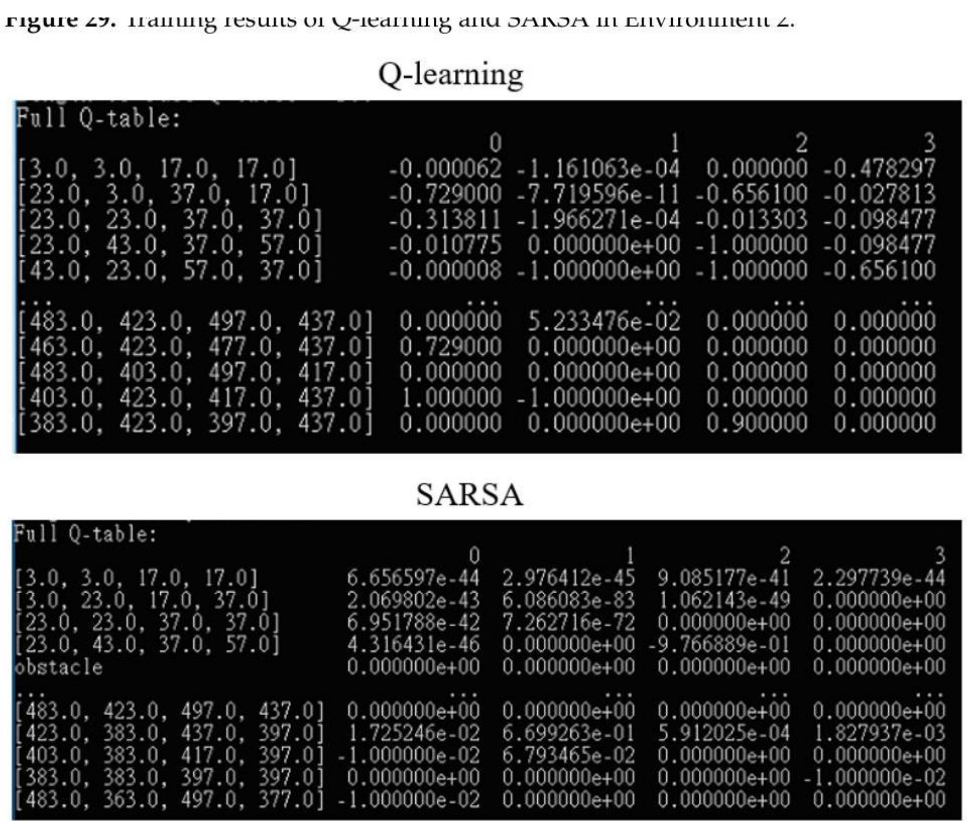


Figure 29. Training results of Q-learning and SARSA in Environment 2.

Q-learning

```
Full Q-table:
                                  0              1         2         3
[3.0, 3.0, 17.0, 17.0]       -0.000062 -1.161063e-04  0.000000 -0.478297
[23.0, 3.0, 37.0, 17.0]      -0.729000 -7.719596e-11 -0.656100 -0.027813
[23.0, 23.0, 37.0, 37.0]     -0.313811 -1.966271e-04 -0.013303 -0.098477
[23.0, 43.0, 37.0, 57.0]     -0.010775  0.000000e+00 -1.000000 -0.098477
[43.0, 23.0, 57.0, 37.0]     -0.000008 -1.000000e+00 -1.000000 -0.656100
...                                ...           ...       ...       ...
[483.0, 423.0, 497.0, 437.0]  0.000000  5.233476e-02  0.000000  0.000000
[463.0, 423.0, 477.0, 437.0]  0.729000  0.000000e+00  0.000000  0.000000
[483.0, 403.0, 497.0, 417.0]  0.000000  0.000000e+00  0.000000  0.000000
[403.0, 423.0, 417.0, 437.0]  1.000000 -1.000000e+00  0.000000  0.000000
[383.0, 423.0, 397.0, 437.0]  0.000000  0.000000e+00  0.900000  0.000000
```

SARSA

```
Full Q-table:
                                         0             1             2             3
[3.0, 3.0, 17.0, 17.0]        6.656597e-44  2.976412e-45  9.085177e-41  2.297739e-44
[3.0, 23.0, 17.0, 37.0]       2.069802e-43  6.086083e-83  1.062143e-49  0.000000e+00
[23.0, 23.0, 37.0, 37.0]      6.951788e-42  7.262716e-72  0.000000e+00  0.000000e+00
[23.0, 43.0, 37.0, 57.0]      4.316431e-46  0.000000e+00 -9.766889e-01  0.000000e+00
obstacle                      0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
...                                    ...           ...           ...           ...
[483.0, 423.0, 497.0, 437.0]  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
[423.0, 383.0, 437.0, 397.0]  1.725246e-02  6.699263e-01  5.912025e-04  1.827937e-03
[403.0, 383.0, 417.0, 397.0] -1.000000e-02  6.793465e-02  0.000000e+00  0.000000e+00
[383.0, 383.0, 397.0, 397.0]  0.000000e+00  0.000000e+00  0.000000e+00 -1.000000e-02
[483.0, 363.0, 497.0, 377.0] -1.000000e-02  0.000000e+00  0.000000e+00  0.000000e+00
```

Figure 30. The Q-table of Q-learning and SARSA in Environment 2.
图 30. 环境 2 中 Q 学习和 SARSA 的 Q 表。
Table 3. The results compared for Environment 2.
表 3. 环境 2 的结果比较。

Environment 2	SARSA	Q-Learning
path length	140 steps	46 steps
exploration time	45 min	25 min

环境 2	SARSA	Q-学习
路径长度	140 步	46 步
探索时间	45 分钟	25 分钟

In Experiment 1, the path points explored by the Q-learning algorithm are converted into real-world latitude and longitude coordinates, according to a given scale. In Figure 31, the yellow line represents the planned path, and the purple line represents the actual trajectory of the hexacopter. In the actual environment, we use a square canvas to represent the position of the no-fly zone. The hexacopter bypasses the no-navigation zone, according to the converted waypoint coordinates, and reaches the target point on the other side. The waypoints are the green points in the right-hand figure of Figure 31. The enlarged working area is shown in the left-hand figure of Figure 31. After reaching the destination, the drone will return to the take-off point.

在实验 1 中，Q 学习算法探索的路径点根据给定的比例转换为现实世界的纬度和经度坐标。在图 31 中，黄色线表示计划路径，紫色线表示六旋翼无人机的实际轨迹。在实际环境中，我们使用一个方形画布来表示禁飞区的位置。六旋翼无人机根据转换后的航点坐标绕过不可航行区，并到达另一侧的目标点。航点是在图 31 右侧图中的绿色点。放大的工作区域显示在图 31 的左侧图中。到达目的地后，无人机将返回起飞点。

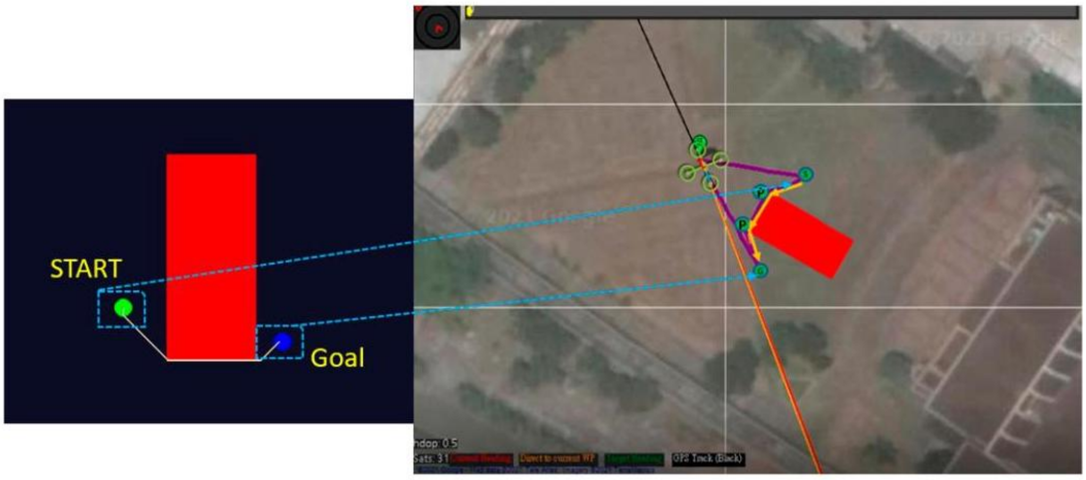


Figure 31. The Q-learning path on the map for Experiment 1.
图 31。实验 1 中地图上的 Q 学习路径。

In Experiment 2, the no-fly zone was increased into two areas, shown as the red square and rectangle in Figure 32. The enlarged working area is shown in the left-hand figure of Figure 32. The planned path is the yellow line, which is indicated in the left-hand figure of Figure 32. The UAV follows the converted latitude and longitude coordinates, which are the green points marked in the right-hand figure of Figure 32 for the path of flight.

在实验 2 中，禁飞区增加到两个区域，如图 32 中的红色正方形和矩形所示。放大的工作区域显示在图 32 的左侧图中。计划路径是黄色线，这在图 32 的左侧图中标出。无人机遵循转换后的纬度和经度坐标，这些坐标在图 32 右侧图中以绿色点标记，用于飞行路径。

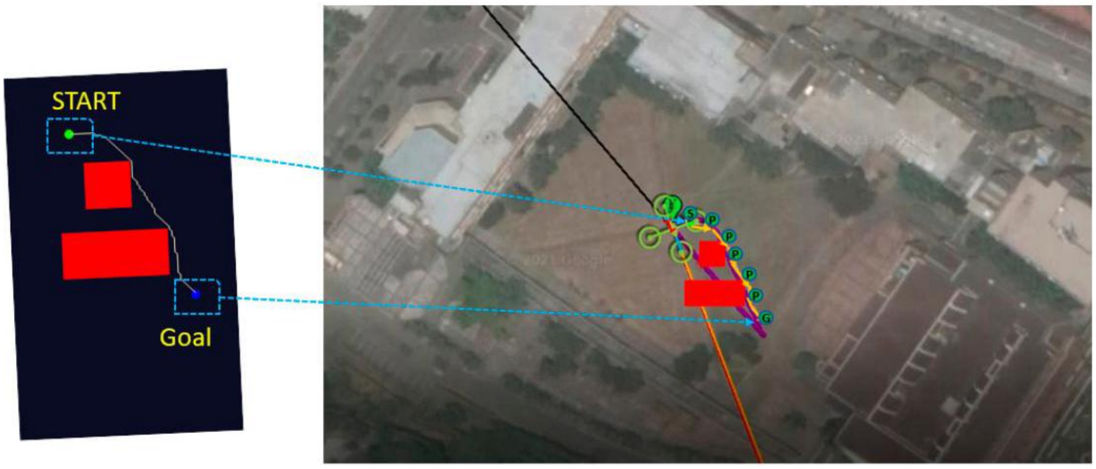


Figure 32. The Q-learning path on the map in Experiment 2.
图 32。实验 2 中地图上的 Q 学习路径。

5.2. Obstacle Avoidance

5.2. 避障

The hexacopter's front obstacle avoidance method is demonstrated using the Microsoft AirSim simulation software environment. In this simulation environment, many trees are placed as imaginary as sails on the sea, and there are undulations given to the terrain to simulate the waves on the sea. The drone continues to fly around in the virtual environment until it collides with an object.

使用 Microsoft AirSim 仿真软件环境演示了六旋翼无人机的正面避障方法。在这个仿真环境中，许多树被放置成仿佛海上的帆船，地形也被起伏处理以模拟海浪。无人机在虚拟环境中继续飞行，直到与物体碰撞。

In this simulation flight test, the input of the photo is 320×180 in size; the parameters are shown in Table 4. The number of training iterations was set to 140,000 . We set the batch size to 32. If the batch size is increased, the convergence speed can be accelerated, but the hardware memory will not be able to load the data. The discount factor, γ , was set to 0.95, and the learning rate, α , was set to 0.00006 . The smaller the learning rate is, the slower the convergence speed is.

在这次模拟飞行测试中，输入的图片大小为 320×180 ；参数显示在表 4 中。训练迭代次数设置为 140,000 次。我们将批量大小设置为 32。如果批量大小增加，收敛速度可以加快，但硬件内存将无法加载数据。折扣因子 γ 设置为 0.95，学习率 α 设置为 0.00006。学习率越小，收敛速度越慢。

Table 4. Training parameters.

表 4. 训练参数。

Iterations	Batch Size	Discount Factor γ	Learning Rate α
140,000	32	0.95	0.000006

迭代次数	批次大小	折扣因子 γ	学习率 α
140,000	32	0.95	0.000006

In the training process, if the number of training times exceeds 150,000, the loss function will start to diverge, and the training effect will worsen. The loss function graph is shown in Figure 33.

在训练过程中，如果训练次数超过 150,000 次，损失函数将开始发散，训练效果将变差。损失函数图显示在图 33 中。

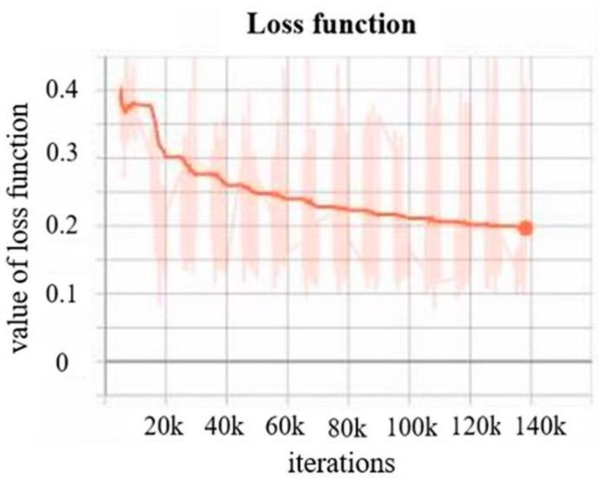


Figure 33. The graph showing the loss function.

图 33. 显示损失函数的图表。

After many iterations, the Q-value gradually increases to achieve the overall maximum benefit, as shown in Figure 34.

经过多次迭代，Q 值逐渐增加以达到整体最大效益，如图 34 所示。

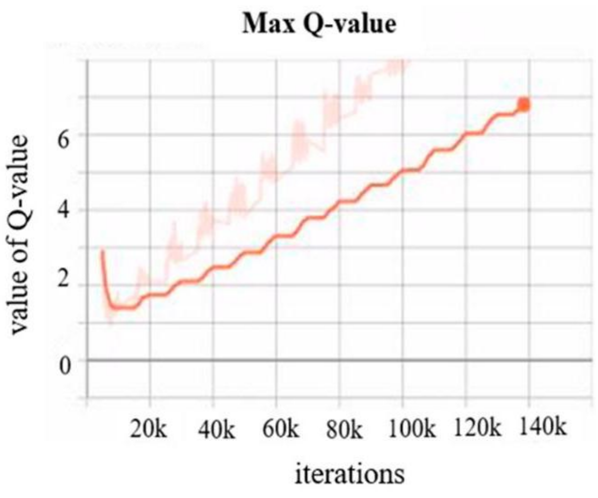
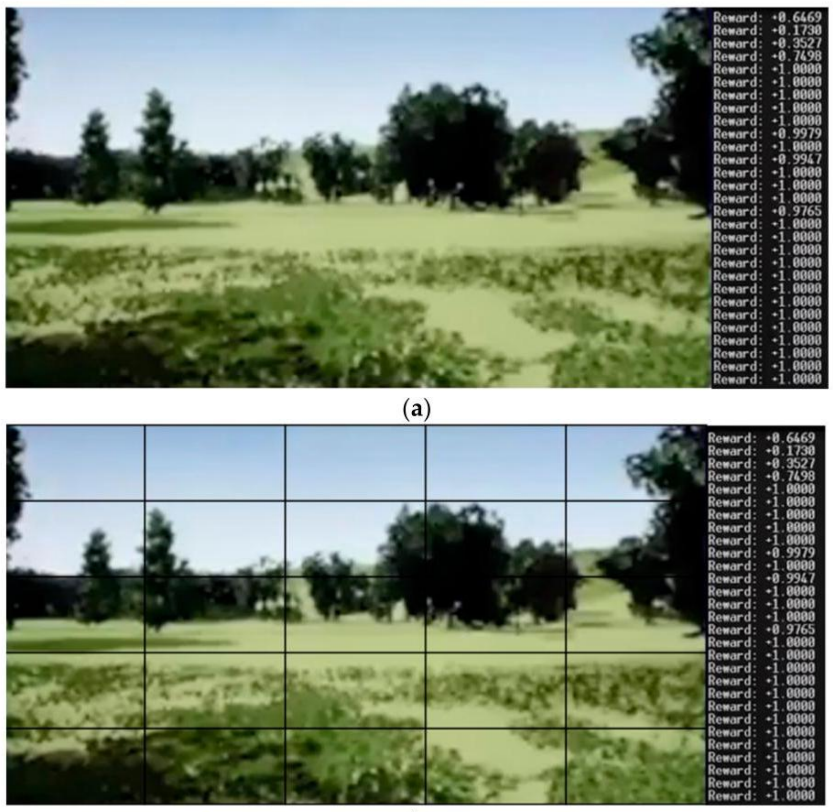


Figure 34. The maximum Q-value.
图 34. 最大 Q 值。

In the previous section, we mentioned a table with a 5 × 5 action space. The task of combining the action space with the input image is shown in Figure 35. There are mostly empty blocks placed in front of the drone; therefore, the rewards are continuously set to achieve the maximum score of +1 . In terms of rewards, the scores are based on the proportion of colors in the pixels in the processed depth image, as shown in Figure 36.

在上一节中，我们提到了一个具有 5 × 5 动作空间的表格。将动作空间与输入图像结合的任务显示在图 35 中。在无人机前方放置了大多数空块；因此，奖励持续设置以达到最高分 +1。在奖励方面，分数基于处理后的深度图像中像素颜色的比例，如图 36 所示。



```
Reward: +0.6469
Reward: +0.1730
Reward: +0.3527
Reward: +0.7498
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +0.9979
Reward: +1.0000
Reward: +0.9947
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +0.9765
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
```

(a)

```
Reward: +0.6469
Reward: +0.1730
Reward: +0.3527
Reward: +0.7498
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +0.9979
Reward: +1.0000
Reward: +0.9947
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +0.9765
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
Reward: +1.0000
```

(b)

Figure 35. The first-person views of the drone and the rewards, next to: (a) the original RGB image, (b) the same image, divided into a 5 × 5 action space.

图 35. 无人机的第一人称视角和旁边的奖励:(a) 原始 RGB 图像，(b) 同一图像，被划分为 5×5 动作空间。

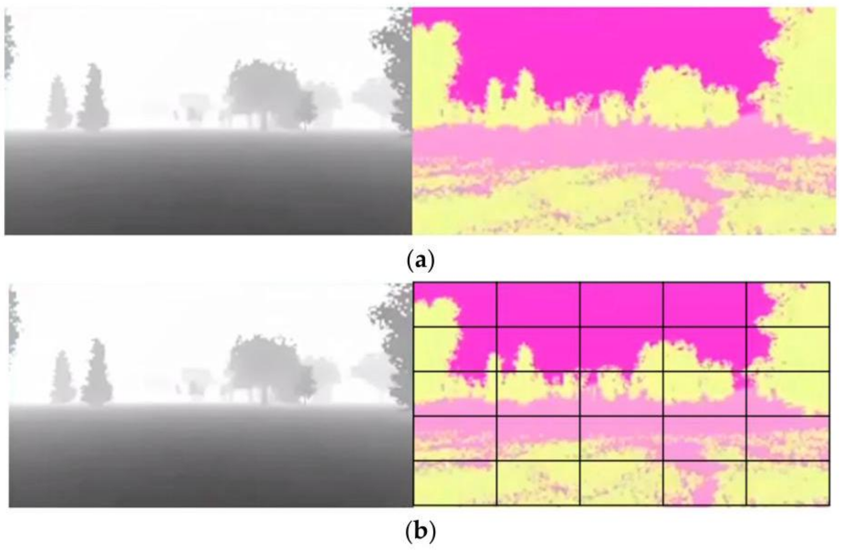


Figure 36. The depth image, divided into a 5×5 action space. (a) The original depth image. (b) The processed depth image.
图 36. 被划分为 5×5 动作空间的深度图像。(a) 原始深度图像。(b) 处理后的深度图像。

Figure 37 shows a scene full of trees and undulating terrain. Figure 38a is the first-person-view (FPV) depth image of the drone in Figure 37, while Figure 38b is the processed depth image and the yellow part represents proximity, so that the distances can be displayed more clearly.

图 37 展示了一个充满树木和起伏地形的一幕。图 38a 是图 37 中无人机的第一人称视角 (FPV) 深度图像，而图 38b 是处理后的深度图像，黄色部分代表近距离，以便距离可以显示得更清楚。

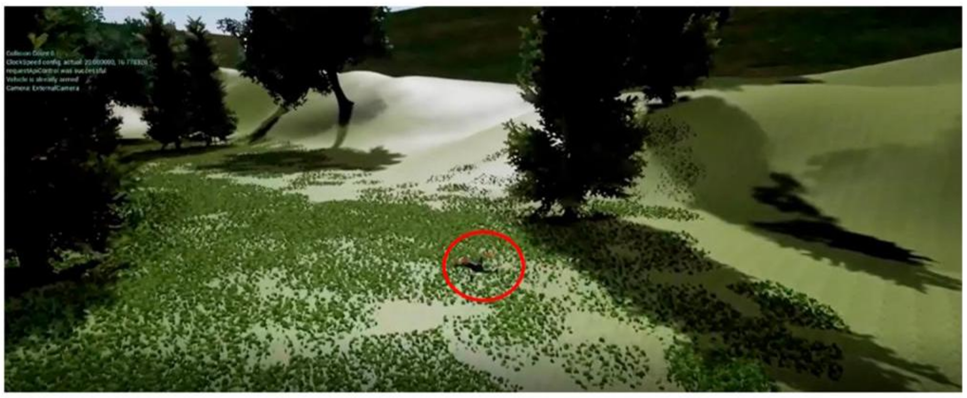


Figure 37. A scene full of trees and undulating terrain.
图 37。一个充满树木和起伏地形的一幕。

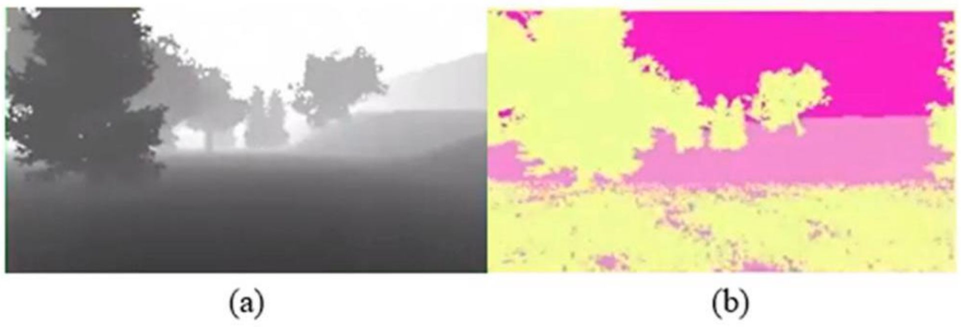


Figure 38. The depth image from Figure 37, (a) is the first-person-view depth image of the drone in Figure 37, (b) is the processed depth image.

图 38。图 37 的深度图像，(a) 是图 37 中无人机的第一人称视角深度图像，(b) 是处理后的深度图像。

At this point, there is a tree directly in front of the drone. The drone sees the tree, as shown in Figure 39a, from a distance, and then starts to select the left-hand direction, based on the Q-table that has been trained. Figure 39b shows that the drone changes direction and the tree is unobstructed in front of the drone. Figure 39c shows the tree that was originally blocking the front of the drone but is now on the right-hand side of the drone. Finally, the UAV completely bypassed the obstacle, as shown in Figure 39d.

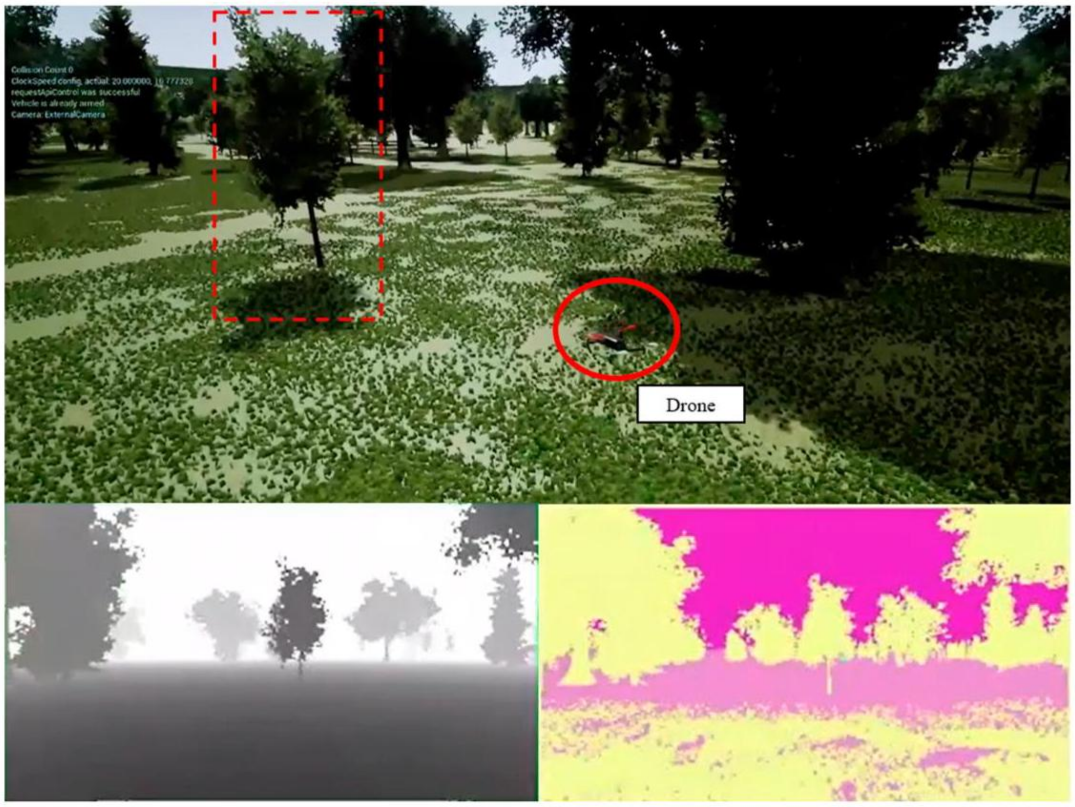
在这一点上，无人机正前方有一棵树。无人机看到了这棵树，如图 39a 所示，从远处开始，然后根据已经训练好的 Q 表选择向左的方向。图 39b 显示无人机改变了方向，树在无人机前方不再阻挡。图 39c 显示了原本阻挡在无人机前方的树现在在无人机的右侧。最后，无人机完全绕过了障碍物，如图 39d 所示。

In the second test, there are two dense trees positioned in front of the drone. It can be seen from the depth map that the two trees can be passed between, as shown in Figure 40a. As the drone moves closer to the trees, it can be seen that the gap in the middle is wide enough to pass through, as shown in Figure 40b. During the crossing, the drone does not fly in a straight line. As shown in Figure 40c, the direction of the UAV is slightly offset, but afterward, the path has been adjusted to the open area. Finally, the UAV has successfully passed through the obstacles, as shown in Figure 40d.

在第二次测试中，无人机前方有两棵浓密的树。从深度图中可以看出，两棵树之间可以穿过，如图 40a 所示。当无人机靠近这两棵树时，可以看到中间的空隙足够宽，可以穿过，如图 40b 所示。在穿越过程中，无人机并不是直线飞行。如图 40c 所示，无人机的方向略有偏移，但之后，路径已经调整到开阔区域。最终，无人机成功穿过了障碍物，如图 40d 所示。

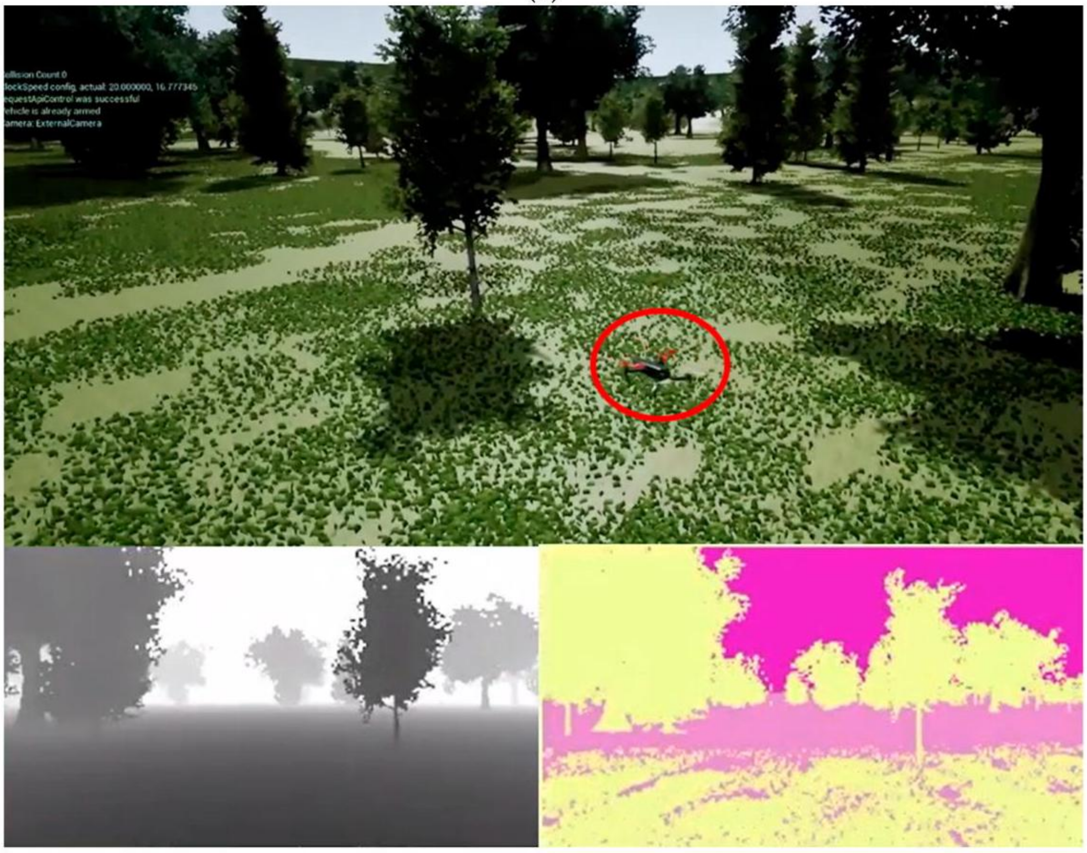
The experiments show that the improved Q-learning algorithm, with the use of new reward and Q-value-updating rules, shows better performance in path planning than the classical reinforcement learning algorithm, SARSA. When we compare the improved Q-learning algorithm with the SARSA algorithm, the improved Q-learning algorithm can reduce computation time by 50% and shorten the path length by 30% . In terms of obstacle avoidance, the proposed deep-learning neural network with deep Q-learning can provide discrete action selections to enable UAVs to avoid obstacles. Real-time visual collision avoidance can be performed successfully. In addition, the proposed ultrasound obstacle avoidance method can achieve a distance error of less than 1.5% and a computing time of less than 0.2 s .

实验表明，通过使用新的奖励和 Q 值更新规则改进的 Q 学习算法，在路径规划方面比经典的强化学习算法 SARSA 表现出更好的性能。当我们比较改进的 Q 学习算法和 SARSA 算法时，改进的 Q 学习算法可以减少计算时间 50% 并缩短路径长度 30% 。在避障方面，提出的结合深度 Q 学习的深度学习神经网络可以为无人机提供离散的动作选择，以实现避障。可以成功执行实时视觉避障。此外，提出的超声波避障方法可以达到小于 1.5% 的距离误差和小于 0.2 s 的计算时间。



```
Collision Count 0
ClockSpeed config, actual: 20.000000, 16.777326
requestApiControl was successful
Vehicle is already armed
Camera: ExternalCamera
```

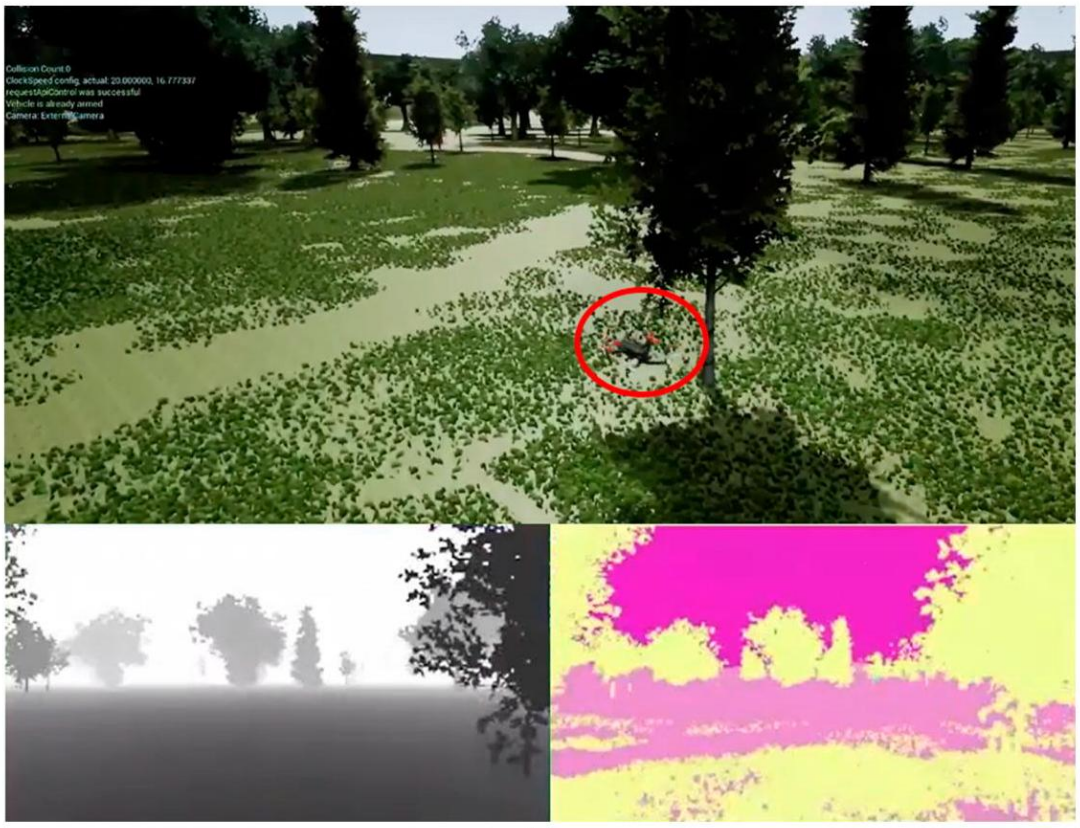
(**a**)



```
ollision Count 0
lockSpeed config, actual: 20.000000, 16.777345
equestApiControl was successful
ehicle is already armed
amera: ExternalCamera
```

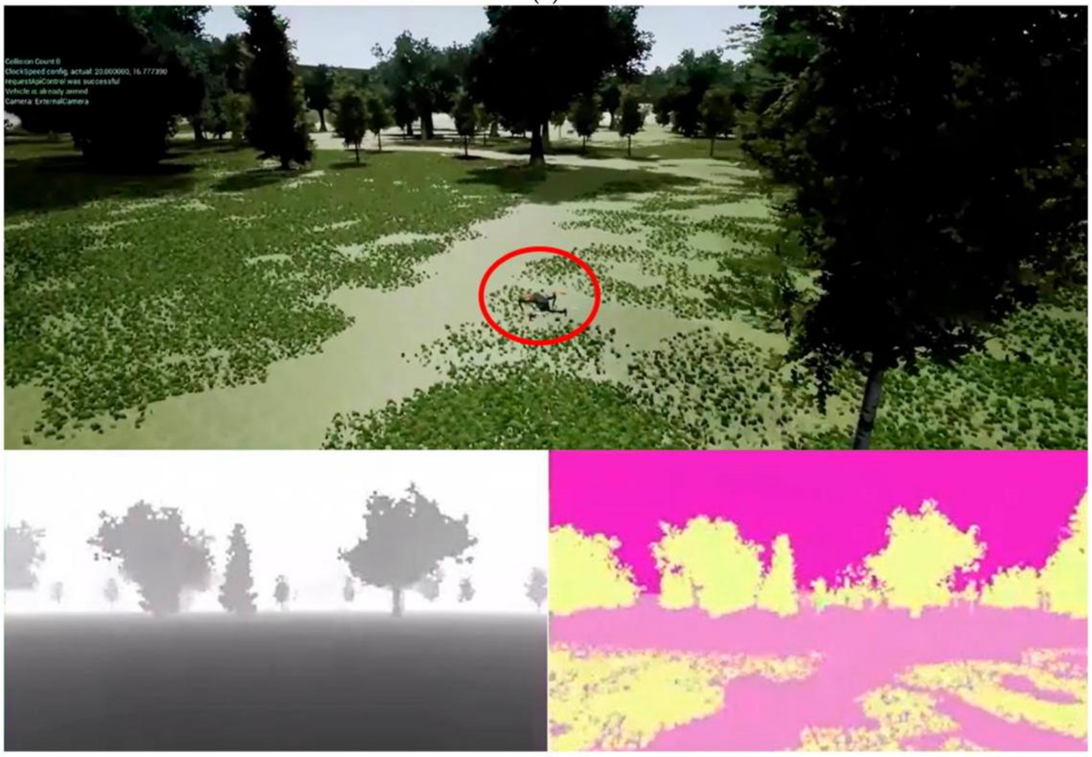
(**b**)

Figure 39. Cont.
图 39. 续



```
Collision Count 0
ClockSpeed config, actual: 20.000000, 16.777337
requestApiControl was successful
Vehicle is already armed
Camera: ExternalCamera
```

(c)

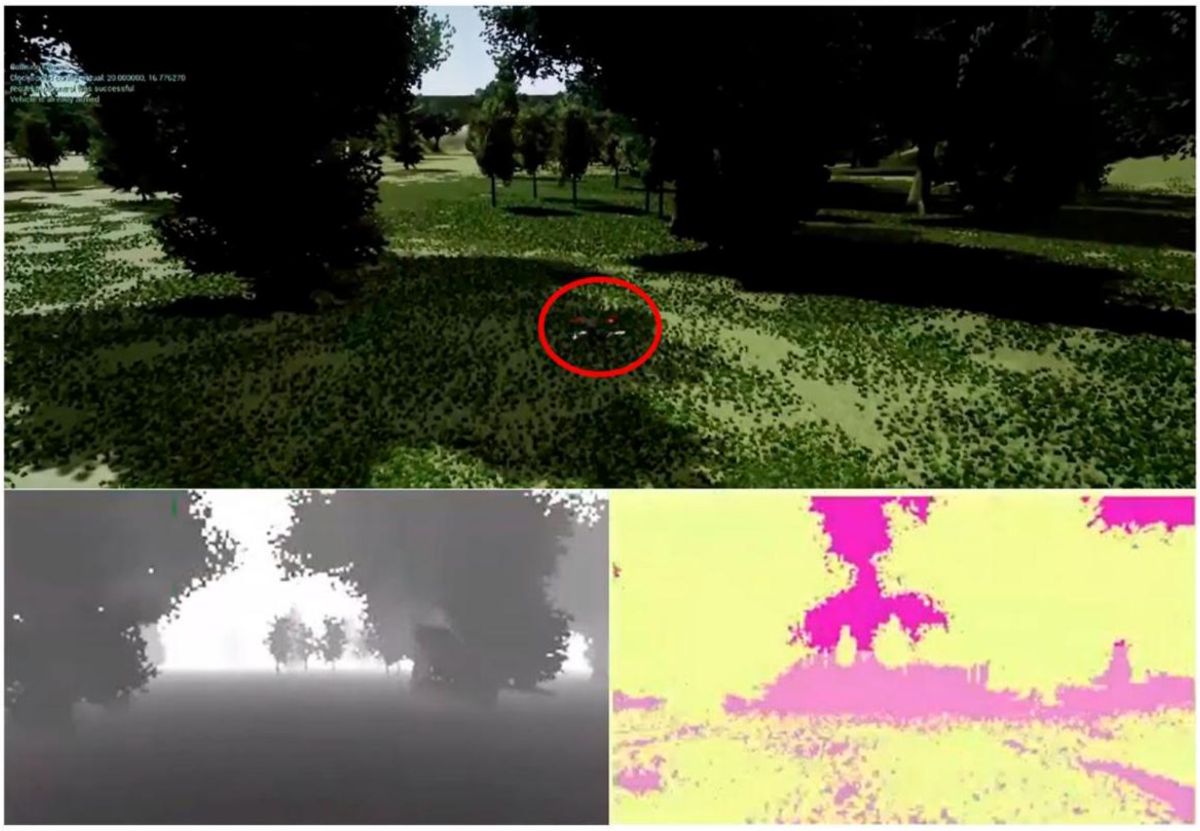


```
Collision Count 0
ClockSpeed config, actual: 20.000000, 16.777390
requestApiControl was successful
Vehicle is already armed
Camera: ExternalCamera
```

(d)

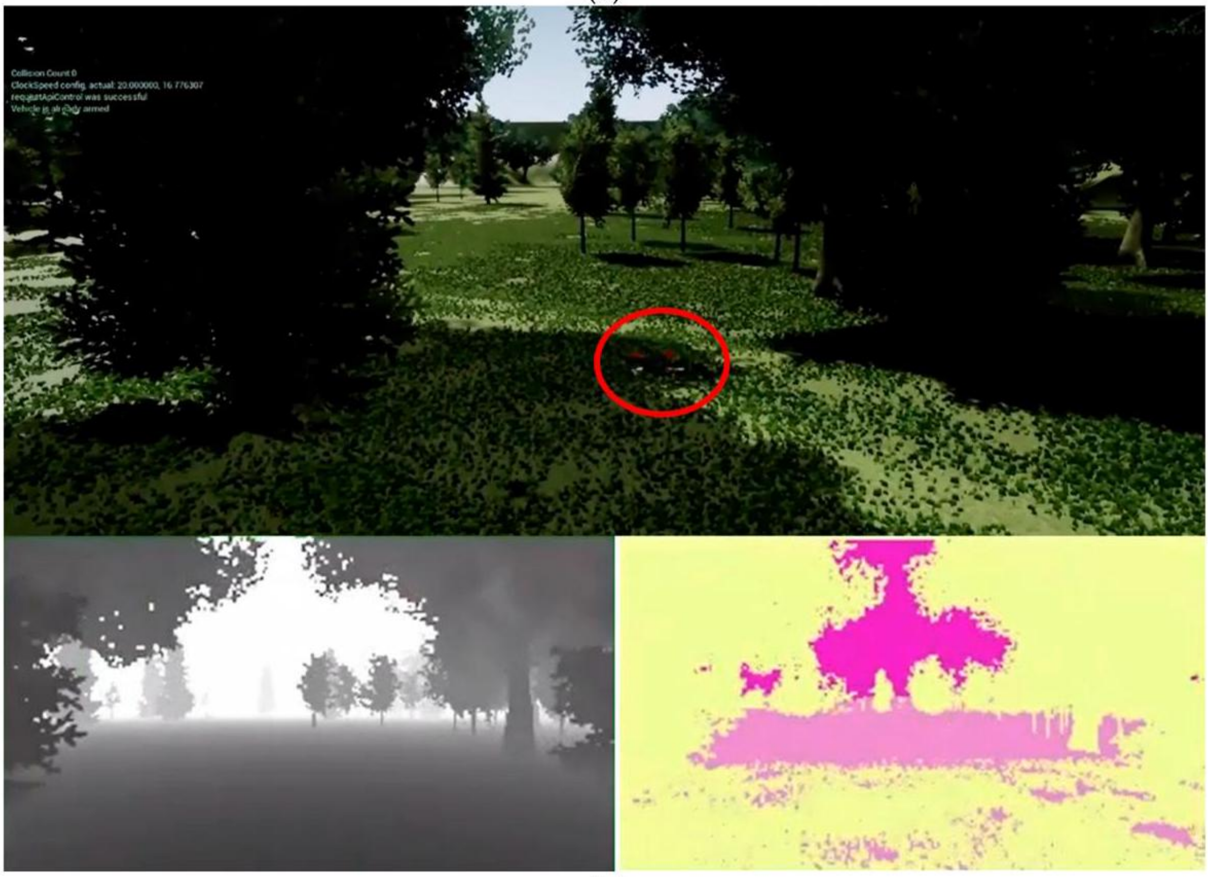
Figure 39. The experiment for testing obstacle avoidance in the simulation environment (part 1), (a) is the starting point, (b) shows the drone changes direction, (c) shows the tree is now on the right-hand side of the drone, (d) shows the UAV completely bypassed the obstacle.

图 39. 模拟环境中测试避障的实验 (第一部分)，(a) 是起点，(b) 显示了无人机改变方向，(c) 显示了树现在在无人机的右侧，(d) 显示了无人机完全绕过障碍物。



```
Collision Count 0
ClockSpeed config, actual: 20.000000, 16.776270
requestApiControl was successful
Vehicle is already armed
```

(**a**)

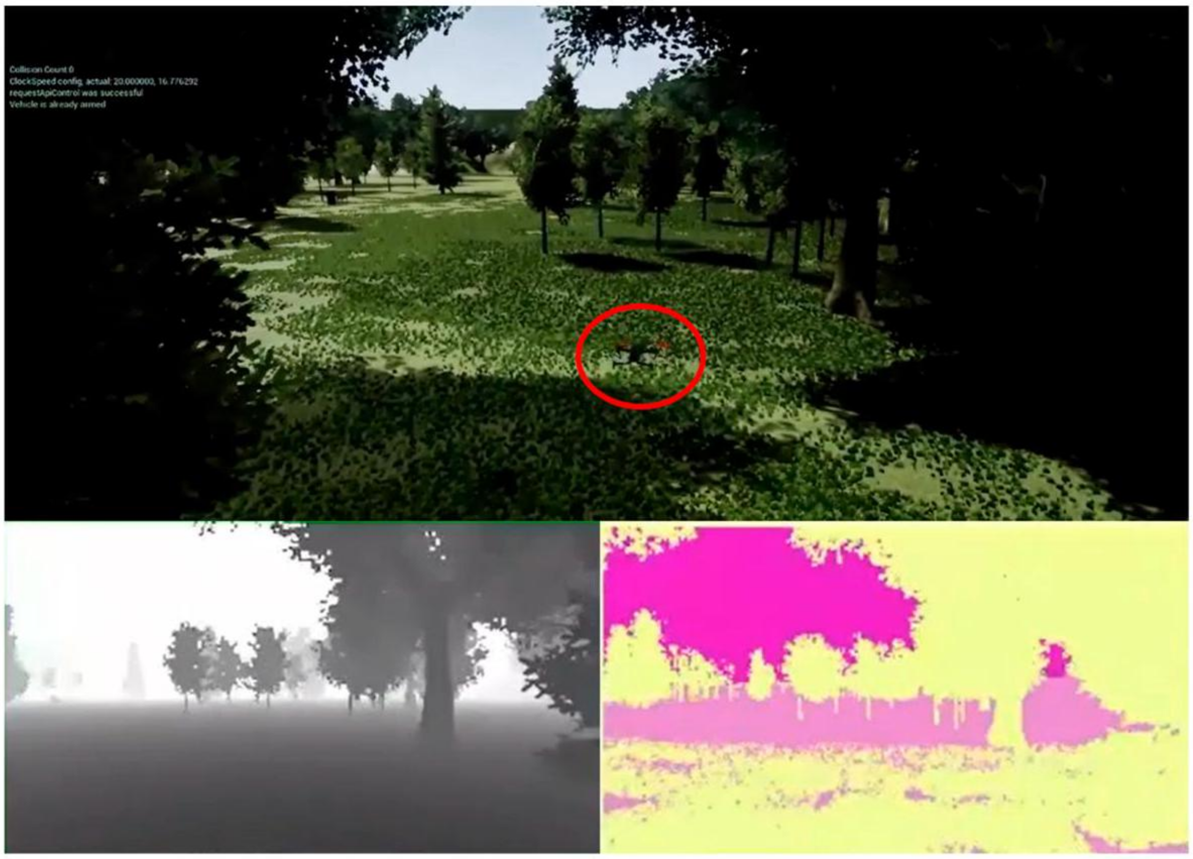


```
Collision Count 0
ClockSpeed config, actual: 20.000000, 16.776307
requestApiControl was successful
Vehicle is already armed
```

(**b**)

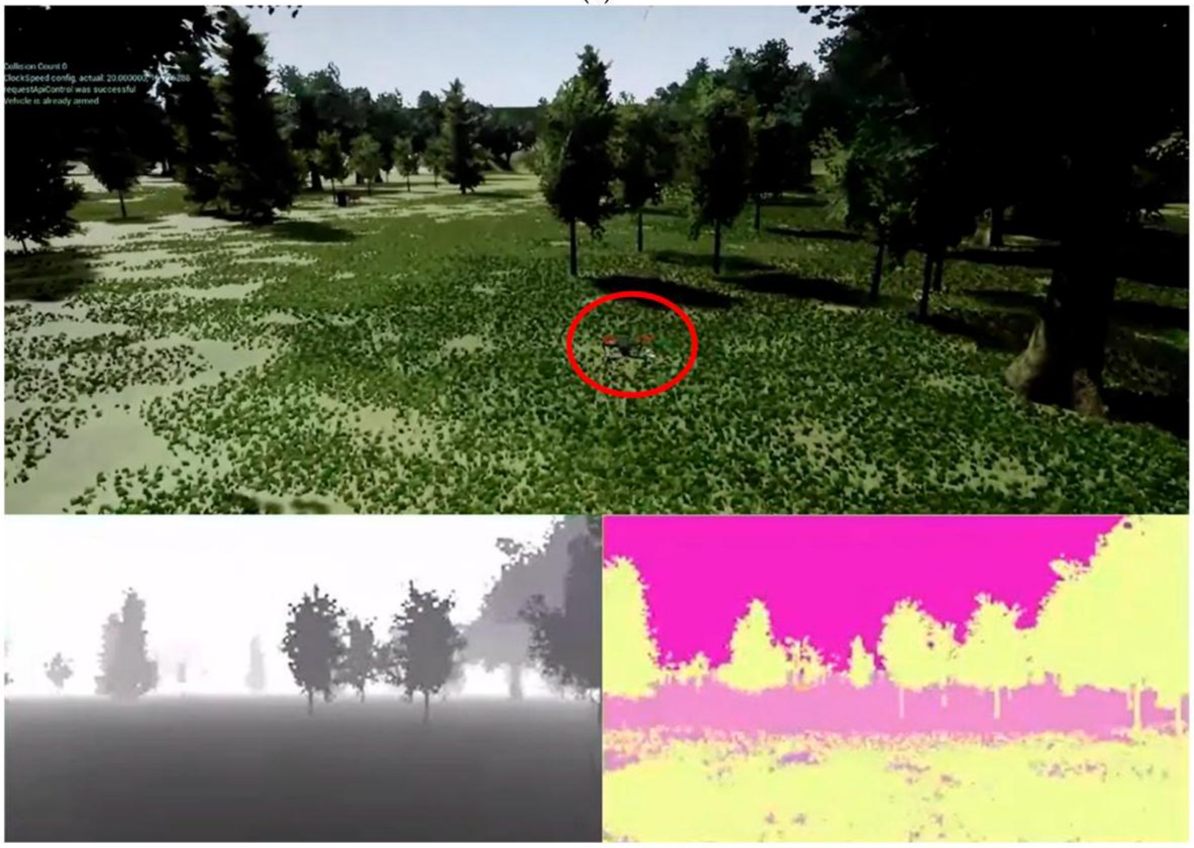
Figure 40. Cont.

图 40. 续



```
Collision Count 0
ClockSpeed config, actual: 20.000000, 16.776292
requestApiControl was successful
Vehicle is already armed
```

(**c**)



```
Collision Count 0
ClockSpeed config, actual: 20.000000, [illegible]286
requestApiControl was successful
Vehicle is already armed
```

(**d**)

Figure 40. The experiment for testing obstacle avoidance in the simulation environment, part 2 , (a) is the starting point, (b) shows the drone moves closer to the trees, (c) shows the drone does not fly in a straight line, (d) shows the UAV has successfully passed through the obstacles.

图 40. 模拟环境中测试避障的实验，第二部分，(a) 是起点，(b) 显示了无人机向树木靠近，(c) 显示了无人机没有直线飞行，(d) 显示了无人机已经成功穿过障碍物。

6. Conclusions

6. 结论

Research studies into UAVs have been booming in the past few years, and the methods of reinforcement learning also present a relatively challenging research field. This paper mainly focuses on the flight processes of UAVs, with a research background of dropping sensors from land to the net-cages used in sea-farming. In this study, UAV path planning using Q-learning and SARSA are placed under new reward and punishment rules, and the Q value updating rules are changed. The results for these two reinforcement learning algorithms are compared. In the real world, the coordinates are converted into latitude and longitude positions and the actual flight route. Using reinforcement learning methods to train UAVs in the real world will be expensive in terms of UAV cost consumption. Therefore, we chose to conduct our research in a virtual environment. The suggested obstacle avoidance method uses a deep Q-learning network (DQN) with an appropriate learning rate, batch size, and iteration, subsequently observing the obstacle-dodging results after training. Depth images from the UAV comprise the inputs of the DQN. The DQN provides discrete action selections for the UAV to avoid obstacles. Real-time visual collision avoidance can thus be performed. For dynamic obstacles to the side of the drone (out of sight), ultrasonic sensors are used; then, the sensing signals are filtered to maintain a stable sensing distance and to adjust UAV displacement after detecting the obstacle. Since drones are susceptible to airflow changes in the external environment and this can affect the fineness of flight, we used a simple control method that maintained a safe distance from objects to ensure that the UAV stayed away from the obstacles.

近年来，无人机 (UAV) 的研究一直在蓬勃发展，强化学习的方法也呈现出相对具有挑战性的研究领域。本文主要关注无人机在海洋养殖中从陆地投放传感器到网箱的飞行过程。在这项研究中，使用 Q 学习和 SARSA 的无人机路径规划被置于新的奖励和惩罚规则之下，并且 Q 值更新规则发生了改变。这两种强化学习算法的结果进行了比较。在现实世界中，坐标被转换为纬度和经度位置以及实际飞行路线。使用强化学习方法在现实世界中训练无人机将导致昂贵的无人机成本消耗。因此，我们选择在虚拟环境中进行我们的研究。建议的避障方法使用具有适当学习率、批量大小和迭代的深度 Q 学习网络 (DQN)，随后观察训练后的避障结果。无人机拍摄的深度图像构成 DQN 的输入。DQN 为无人机提供避免障碍的离散动作选择。因此，可以执行实时视觉避障。对于无人机侧面的动态障碍物 (在视线之外)，使用超声波传感器；然后，对感知信号进行过滤以保持稳定的感知距离，并在检测到障碍物后调整无人机的位移。由于无人机容易受到外部环境气流变化的影响，这可能会影响飞行的精确性，因此我们使用了一种简单的控制方法，保持与物体之间的安全距离，以确保无人机远离障碍物。

Path planning and obstacle avoidance are very important and basic functions for UAVs. The use of the Q-learning algorithm, a reinforcement learning method, can solve many of the complex problems that traditional algorithms cannot solve; therefore, there is plenty of room for development in the field of artificial intelligence. In this study, a Q-learning algorithm is proposed for application to path planning in an unknown environment. In terms of path planning, the reward design and action selection in the SARSA algorithm and Q-learning algorithm are similar to the methods of real-world learning. In terms of updating the Q-table, it is obvious that the Q-learning algorithm offers better benefits because the benefits of the overall path are considered fully. When we compared the modified Q-learning algorithm with the SARSA algorithm, the modified Q-learning algorithm showed better performance by reducing computation time by 50% and path length by 30% . In terms of obstacle avoidance, the application of UAV dodging is presented. Deep learning, when combined with reinforcement learning, can solve many complex problems, and obstacle avoidance is indeed a complex problem. The design method of experience replay allows the drone to access past states and actions that are similar to deep learning methods, which generally provide a label for learning. The side of the drone used for ultrasonic sensing obstacle avoidance will also effectively keep the drone at a safe distance from approaching objects. The proposed obstacle avoidance method can achieve a distance error of less than 1.5% and a computing time of less than 0.2 sec. For path planning and obstacle avoidance in more complex environments, reinforcement learning is expected to show further amazing results; there is no standard answer to this method, therefore the results will be different each time, and this is expected to enable even more amazing results. The main limitation of this work is the obstacle avoidance distance. The proposed system utilizes ultrasonic sensors to measure the distance between the UAV and the obstacles, but the measuring distance is short. In the case of high-speed moving objects, the UAV does not have enough time for computing a path and avoiding dynamic obstacles. In the future, a laser detector with a long measurement range will be needed to replace the ultrasonic sensors.

路径规划和避障是无人机非常重要的基础功能。使用 Q 学习算法，一种强化学习方法，可以解决许多传统算法无法解决的复杂问题；因此，在人工智能领域还有很大的发展空间。在本研究中，提出了一种应用于未知环境路径规划的 Q 学习算法。在路径规划方面，SARSA 算法和 Q 学习算法的奖励设计及动作选择与现实世界学习的方法相似。在更新 Q 表方面，Q 学习算法显然提供了更好的效益，因为它充分考虑了整体路径的效益。当我们比较改进后的 Q 学习算法与 SARSA 算法时，改进后的 Q 学习算法通过减少计算时间 50% 和路径长度 30% 显示出更好的性能。在避障方面，提出了无人机躲避的应用。深度学习与强化学习结合可以解决许多复杂问题，避障确实是一个复杂问题。经验回放的设计方法允许无人机访问类似于深度学习方法的历史状态和动作，这通常为学习提供标签。无人机用于超声波感应避障的一侧也将有效地使无人机与接近的物体保持安全距离。所提出的避障方法可以达到小于 1.5% 的距离误差和小于 0.2 秒的计算时间。在更复杂的环境中进行路径规划和避障时，强化学习预计将展现出进一步惊人的结果；由于这种方法没有标准答案，因此每次的结果都会有所不同，这预计将能够实现更加惊人的结果。这项工作的主要局限性在于避障距离。提出的系统使用超声波传感器测量无人机与障碍物之间的距离，但测量距离较短。在高速移动的物体情况下，无人机没有足够的时间进行路径计算和避开动态障碍物。在未来，需要用长测量范围的激光检测器替换超声波传感器。

Author Contributions: Conceptualization, G.-T.T. and J.-G.J.; methodology, G.-T.T. and J.-G.J.; software, G.-T.T.; validation, G.-T.T. and J.-G.J.; formal analysis, G.-T.T. and J.-G.J.; investigation, G.-T.T. and J.-G.J.; resources, G.-T.T. and J.-G.J.; data curation, G.-T.T.; writing-original draft preparation, G.-T.T.; writing-review and editing, J.-G.J.; visualization, G.-T.T. and J.-G.J.; supervision, J.-G.J.; project administration, J.-G.J.; funding acquisition, J.-G.J. All authors have read and agreed to the published version of the manuscript.

作者贡献: 构思，G.-T.T. 和 J.-G.J.；方法论，G.-T.T. 和 J.-G.J.；软件，G.-T.T.；验证，G.-T.T. 和 J.-G.J.；形式分析，G.-T.T. 和 J.-G.J.；调查研究，G.-T.T. 和 J.-G.J.；资源，G.-T.T. 和 J.-G.J.；数据整理，G.-T.T.；写作-原始草稿准备，G.-T.T.；写作-审查与编辑，J.-G.J.；可视化，G.-T.T. 和 J.-G.J.；监督，J.-G.J.；项目管理，J.-G.J.；资金获取，J.-G.J.。所有作者已阅读并同意发表的稿件版本。

Funding: This research was funded by Ministry of Science and Technology (Taiwan), grant number MOST 110-2221-E-019-075-MY2.

资金: 本研究得到了台湾科技部 (MOST 110-2221-E-019-075-MY2) 的资助。

Institutional Review Board Statement: Not applicable.

机构审查委员会声明: 不适用。

Informed Consent Statement: Not applicable.

知情同意声明: 不适用。

Data Availability Statement: Not applicable.

数据可用性声明: 不适用。

Conflicts of Interest: The authors declare no conflict of interest.

利益冲突: 作者声明没有利益冲突。

References

参考文献

1. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. 1968, 4, 100-107. [CrossRef]

2. Stentz, A. Optimal and efficient path planning for unknown and dynamic environments. Int. J. Robot. Autom. Syst. 1995, 10, 89–100.

3. LaValle, S.M.; Kuffner, J.J. Randomized kinodynamic planning. Int. J. Robot. Res. 2001, 20, 378-400. [CrossRef]

4. Sen, Y.; Zhongsheng, W. Quad-Rotor UAV Control Method Based on PID Control Law. In Proceedings of the 2017 International Conference on Computer Network, Electronic and Automation (ICCNEA), Xi'an, China, 23-26 September 2017; pp. 418-421. [CrossRef]

5. Kamel, B.; Yasmina, B.; Laredj, B.; Benaoumeur, I.; Zoubir, A. Dynamic Modeling, Simulation and PID Controller of Unmanned Aerial Vehicle UAV. In Proceedings of the 2017 Seventh International

Conference on Innovative Computing Technology (INTECH), Porto, Portugal, 12-13 July 2017; pp. 64-69. [CrossRef]

6. Lee, S.; Shim, T.; Kim, S.; Park, J.; Hong, K.; Bang, H. Vision-Based Autonomous Landing of a Multi-Copter Unmanned Aerial Vehicle using Reinforcement Learning. In Proceedings of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS), Dallas, TX, USA, 12-15 June 2018; pp. 108-114. [CrossRef]

7. Sugimoto, T.; Gouko, M. Acquisition of Hovering by Actual UAV Using Reinforcement Learning. In Proceedings of the 2016 3rd. International Conference on Information Science and Control Engineering (ICISCE), Beijing, China, 8-10 July 2016; pp. 148-152. [CrossRef]

8. Lampersberger, T.; Feger, R.; Haderer, A.; Egger, C.; Friedl, M.; Stelzer, A. A 24-GHz Radar with 3D-Printed and Metallized Lightweight Antennas for UAV Applications. In Proceedings of the 2018 48th European Microwave Conference (EuMC), Madrid, Spain, 23-27 September 2018; pp. 1413-1416. [CrossRef]

9. Udvardy, P.; Jancsó, T.; Beszédes, B. 3D Modelling by UAV Survey in a Church. In 2019 New Trends in Aviation Development (NTAD); IEEE: Piscataway, NJ, USA, 2019; pp. 189-192. [CrossRef]

10. Ma, Z.; Ai, B.; He, R.; Wang, G.; Niu, Y.; Zhong, Z. A Wideband Non-Stationary Air-to-Air Channel Model for UAV Communications. IEEE Trans. Veh. Technol. 2020, 69, 1214-1226. [CrossRef]

11. Feng, K.; Li, W.; Ge, S.; Pan, F. Packages Delivery Based on Marker Detection for UAVs. In Proceedings of the 2020 Chinese Control and Decision Conference (CCDC), Hefei, China, 22-24 August 2020; pp. 2094-2099. [CrossRef

12. Silver, D.; Huang, A.; Maddison, C.J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. Mastering the Game of Go with Deep Neural Networks and Tree Search. Nature 2016, 529, 484-489. [CrossRef] [PubMed]

13. Yan, C.; Xiang, X. A Path Planning Algorithm for UAV Based on Improved Q-Learning. In Proceedings of the 2018 2nd International Conference on Robotics and Automation Sciences (ICRAS), Wuhan, China, 23-25 June 2018; pp. 1-5. [CrossRef]

14. Zhang, T.; Huo, X.; Chen, S.; Yang, B.; Zhang, G. Hybrid Path Planning of a Quadrotor UAV Based on Q-Learning Algorithm. I Proceedings of the 2018 37th Chinese Control Conference (CCC), Wuhan, China, 25-27 July 2018; pp. 5415-5419. [CrossRef]

15. Li, R.; Fu, L.; Wang, L.; Hu, X. Improved Q-learning Based Route Planning Method for UAVs in Unknown Environment. In Proceedings of the 2019 IEEE 15th International Conference on Control and Automation (ICCA), Edinburgh, UK, 16-19 July 2019; pp. 118-123. [CrossRef]

16. Hou, X.; Liu, F.; Wang, R.; Yu, Y. A UAV Dynamic Path Planning Algorithm. In Proceedings of the 2020 35th Youth Academic Annual Conference of Chinese Association of Automation (YAC), Zhanjiang, China, 16-18 October 2020; pp. 127-131. [CrossRef]

17. Anwar, A.; Raychowdhury, A. Autonomous Navigation via Deep Reinforcement Learning for Resource Constraint Edge Nodes Using Transfer Learning. IEEE Access 2020, 8, 26549-26560. [CrossRef]

18. Singla, A.; Padakandla, S.; Bhatnagar, S. Memory-Based Deep Reinforcement Learning for Obstacle Avoidance in UAV Wi Limited Environment Knowledge. IEEE Trans. Intell. Transp. Syst. 2021, 22, 107-118. [CrossRef]

19. Duc, N.; Hai, Q.; Van, D.; Trong, T.; Trong, T. An Approach for UAV Indoor Obstacle Avoidance Based on AI Technique with Ensemble of ResNet8 and Res-DQN. In Proceedings of the 2019 6th NAFOSTED Conference on Information and Computer Science (NICS); 2019; pp. 330-335. [CrossRef]

20. Çetin, E.; Barrado, C.; Muñoz, G.; Macias, M.; Pastor, E. Drone Navigation and Avoidance of Obstacles Through Deep Reinforcement Learning. In Proceedings of the 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC), San Diego, CA, USA, 8-12 September 2019; pp. 1-7. [CrossRef]

21. Wu, T.; Tseng, S.; Lai, C.; Ho, C.; Lai, Y. Navigating Assistance System for Quadcopter with Deep Reinforcement Learning. In Proceedings of the 2018 1st International Cognitive Cities Conference (IC3), Okinawa, Japan, 7-9 August 2018; pp. 16-19. [CrossRef]

22. Tu, G.; Juang, J. Path Planning and Obstacle Avoidance Based on Reinforcement Learning for UAV Application. In Proceedings of the 2021 International Conference on System Science and Engineering, Ho Chi Minh City, Vietnam, 26-28 August 2021

23. Zheng, Y.; Wang, L.; Xi, P. Improved Ant Colony Algorithm for Multi-Agent Path Planning in Dynamic Environment. In Proceedings of the 2018 International Conference on Sensing, Diagnostics, Prognostics, and Control (SDPC), Xi'an, China, 15-17 August 2018; pp. 732-737. [CrossRef]

24. Zhang, D.; Xian, Y.; Li, J.; Lei, G.; Chang, Y. UAV Path Planning Based on Chaos Ant Colony Algorithm. In Proceedings of the 2015 International Conference on Computer Science and Mechanical Automation (CSMA), Washington, DC, USA, 23-25 October 2015; pp. 81-85. [CrossRef]

25. Yujie, L.; Yu, P.; Yixin, S.; Huajun, Z.; Danhong, Z.; Yong, S. Ship Path Planning Based on Improved Particle Swarm Optimization. In Proceedings of the 2018 Chinese Automation Congress (CAC), Xi'an, China, 20 November-2 December 2018; pp. 226-230. [CrossRef]

26. Markov, A.A. Extension of the Limit Theorems of Probability Theory to a Sum of Variables Connected in a Chain; Reprinted in Appendix B of: R. Howard. Dynamic Probabilistic Systems, Volume 1: Markov Chains; John Wiley and Sons: Hoboken, NJ, USA, 1971.

27. Sammut, C.; Webb, G.I. (Eds.) Bellman Equation. In Encyclopedia of Machine Learning; Springer: Boston, MA, USA, 2011. [CrossRef]

28. Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. 16 Greedy Algorithms. In Introduction to Algorithms; MIT Press: Cambridge, MA, USA, 2001; p. 370. ISBN 978-0-262-03293-3.

29. Tokic, M. Adaptive ε -greedy Exploration in Reinforcement Learning Based on Value Differences. In KI 2010: Advances in Artificial Intelligence; Lecture Notes in Computer Science, 6359; Springer-Verlag: Berlin/Heidelberg, Germany, 2010; pp. 203-210. [CrossRef]

30. Watkins, C.; Dayan, P. Q-learning. Mach. Learn. 1992, 8, 279-292. [CrossRef]

31. Xu, D.; Fang, Y.; Zhang, Z.; Meng, Y. Path Planning Method Combining Depth Learning and Sarsa Algorithm. In Proceedings of the 2017 10th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, China, 9-10 December 2017; pp. 77-82. [CrossRef]

32. Zhou, B.; Wang, W.; Wang, Z.; Ding, B. Neural Q Learning Algorithm based UAV Obstacle Avoidance. In Proceedings of the 2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC), Xiamen, China, 10-12 August 2018; pp. 1-6. [CrossRef]

33. Xiaodong, Y.; Rui, L.; Yingjing, S.; Xiang, C. Obstacle Avoidance for Outdoor Flight of a Quadrotor Based on Computer Vision. In Proceedings of the 2017 29th Chinese Control and Decision Conference (CCDC), Chongqing, China, 28-30 May 2017; pp. 3841-3846. [CrossRef]

34. Hu, J.; Niu, Y.; Wang, Z. Obstacle Avoidance Methods for Rotor UAVs Using RealSense Camera. In Proceedings of the 2017 Chinese Automation Congress (CAC); 2017; pp. 7151-7155. [CrossRef]

35. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. arXiv 2013, arXiv:1312.5602.

36. Airsim Open Source Platform at Github. Available online: https://github.com/Microsoft/AirSim (accessed on 23 July 2020).

37. Huang, T.; Juang, J. Real-time Path Planning and Fuzzy Based Obstacle Avoidance for UAV Application. In Proceedings of the 2020 International Conference on System Science and Engineering (ICSSE), Kagawa, Japan, 31 August-3 September 2020.