

基于双深度 Q 网络的智能决策系统研究

况立群¹,冯 利¹,韩 燮¹,贾灵昊²,郭广行³

(1. 中北大学 大数据学院,山西 太原 030051;

2. 北方自动控制技术研究所,山西 太原 030006;

3. 太原师范学院 地理科学学院,山西 太原 030006)

摘 要:目前智能决策系统中的经典算法智能化程度较低,而更为先进的强化学习算法应用于复杂决策任务又会导致存储上的维度灾难问题。针对该问题,提出了一种基于双深度 Q 网络的智能决策算法,改进了目标 Q 值计算方法,并将动作选择和策略评估分开进行,从而获得更加稳定有效的策略。智能体对输入状态进行训练,输出一个较优的动作来驱动智能体行为,包括环境感知、动作感知及任务协同等,继而在复杂度较高的决策环境中顺利完成给定任务。基于 Unity3D 游戏引擎开发了虚拟智能对抗演练的验证系统,对演练实时状态和智能体训练结果进行可视化,验证了双深度 Q 网络模型的正确性和稳定性,有效解决了强化学习算法存在的灾难问题。该智能决策算法有望在策略游戏、对抗演练、任务方案评估等领域发挥作用。

关键词:深度强化学习;深度 Q 网络;对抗演练;仿真训练;Unity3D

中图分类号:TP391.9

文献标识码:A

文章编号:1673-629X(2022)02-0137-06

doi:10.3969/j.issn.1673-629X.2022.02.022

Research on Intelligent Decision-making System Based on Double Deep Q-Network

KUANG Li-qun¹,FENG Li¹,HAN Xie¹,JIA Jiong-hao²,GUO Guang-xing³

(1. School of Data Science and Technology, North University of China, Taiyuan 030051, China;

2. North Automatic Control Technology Institute, Taiyuan 030006, China;

3. School of Geography Science, Taiyuan Normal University, Taiyuan 030006, China)

Abstract:At present, the classical algorithms in intelligent decision-making systems have a lower degree of intelligence, and the application of more advanced reinforcement learning algorithms to complex decision-making tasks will lead to dimensional disasters on storage. Aiming at this problem, an intelligent decision-making algorithm based on double depth Q-network is proposed. The calculation method of target Q value is improved, and the action selection and strategy evaluation are carried out separately, so as to obtain more stable and effective strategies. The agent trains the input state and outputs a better action to drive the agent behavior, including environment perception, action perception and task coordination, and then successfully completes the given task in a more complex decision-making environment. Based on Unity3D game engine, a verification system for virtual intelligent confrontation drill is developed, which visualizes the real-time states of the drill and the agent training results, verifies the correctness and stability of the double deep Q-network model, and effectively solves the disaster problem of reinforcement learning algorithms. The intelligent decision algorithm proposed is expected to play a role in strategy games, confrontation drills, mission plan evaluations and other fields.

Key words:deep reinforcement learning; deep Q-network; confrontation drill; simulation training; Unity3D

0 引言

决策的主要任务是对诸多影响因素进行综合分析,进而产生效益最好且所需付代价最小的可行方案。通常需要从多个角度来综合考量,仅仅依靠人力决策

将面临着数据集庞大、模型设计复杂、预测准确性差、受人为因素影响大等问题^[1]。智能决策算法可有效解决上述问题,其可靠性高,受个人经验和思维能力等主观影响较小,面对庞大的数据计算问题通过计算机辅

收稿日期:2021-04-01

修回日期:2021-08-02

基金项目:国家级装备预研项目(41401020402)

作者简介:况立群(1976-),男,博士,教授,CCF 会员(48959M),通讯作者,研究方向为人工智能与计算机视觉;冯 利(1994-),男,硕士研究生,研究方向为仿真与可视化。

助决策也可迎刃而解。随着计算机技术的发展和决策算法的逐渐成熟,智能决策算法被广泛应用于医疗、教育、交通以及军事等各个领域^[2-3],智能决策系统在诸多领域中发挥着越来越重要的作用。

传统的智能决策算法一般采用动态规划、决策树、遗传算法等理论方法,陈建凯^[4]提出了区间值属性的单调决策树算法,有效提高了单调分类问题的计算效率。王辉等人^[5]提出了一种基于动态规划的改进算法,提高了算法收敛速度和寻优能力。但在智能决策领域这些借助人经验的经典算法智能化程度通常较低,而且遗传算法对新空间的探索能力有限,容易出现“早熟”问题;动态规划占据过多的空间,对计算机资源造成浪费;决策树受主观性影响较大,可能导致决策失误。

目前智能决策领域更多采用强化学习来驱动智能体决策行为,可有效弥补传统决策算法所面临的不足^[6]。雷莹等人^[7]开发出一种合作 Markov 决策系统,提出了一种寻找最优策略对的学习算法,智能体可交替执行行为,实现多智能体之间任务协同的决策演练。强化学习中的经典算法 Q-learning^[8]是一种离线策略的学习算法,根据策略所选取的动作与当前环境相互交互,产生相应的奖励值以及下一步的状态来不断地更新 Q 表,得到更优的 Q 函数,从而探索出更优的应对环境新态势的决策方案。然而,在空间和时间多样化、复杂化的决策任务中,任务状态已经庞大到无法通过 Q 表进行存储,导致维度灾难问题。

阿尔法围棋^[9]是近几年由 Google DeepMind 开发的基于深度强化学习的人工智能围棋软件,其通过与自身进行上千万次的博弈来提升棋艺,击败了人类顶级围棋棋手,而且在对弈的过程中竟运用了许多创新招式。这为该文研究智能决策算法提供了新的思路,即研究基于深度强化学习算法的深度神经网络来解决维度灾难问题。该文通过虚拟的对抗演练系统对智能决策算法进行验证,智能体根据地图数据、环境数据以及敌方的兵力布置等信息,利用双深度 Q 网络的深度强化学习算法与环境相互交互,不断地试错学习,在尽量躲避敌方火力覆盖范围的情况下,短时间确定敌军目标,并以最优的策略压制、歼灭敌军,击毁敌军基地。该文应用于智能决策算法领域,为实施智能决策系统提供典型案例。

1 双深度 Q 网络算法

1.1 深度 Q 网络算法

强化学习中的经典算法 Q-Learning 虽然可以求出最优策略,但是无法解决维度灾难问题。该算法为了实现在状态动作函数的不断迭代,将状态动作函数的

值储存于内存中,但是计算机的内存空间又极其有限。假如在一个大小为 $50 * 50$ 的地理场景中添加 10 个智能体,那么仅仅是智能体的位置就有约 10^{33} 种不同的情况,显然大多数计算机都无法存储如此巨大的状态表。

深度 Q 网络算法 (deep Q network, DQN) 是由 Mnih 等人将卷积神经网络与传统的强化学习中的 Q 学习算法相结合所提出的^[10-11],使用神经网络来表示价值近似函数,将深度学习的概念融入强化学习中,采用价值近似函数去对状态价值函数 Q 进行近似估计,故而解决了 Q-Learning 算法的维度灾难问题^[12]。

1.1.1 DQN 模型结构

DQN 的模型结构如图 1 所示。

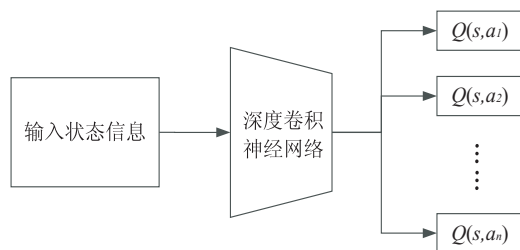


图 1 DQN 模型结构

该模型运用的是 Q-Learning 的一种变种训练深度卷积神经 (CNN), 算法模型的输入是初始状态信息, 将该输入信息离散化后经过 CNN 的非线性变换, 最终在输出层产生每个动作相应的 Q 值。

1.1.2 目标网络

DQN 采用近似表示值函数的优化目标, 将值函数参数化表示为 $Q(\varphi(S), A, \omega)$, 每次迭代的优化目标 Q 值为:

$$Y_j^{\text{DQN}} = R_j + \gamma \max_a Q(\varphi(S_j'), A_j, \omega) \quad (1)$$

其中, $\varphi(S_j')$ 为下一时刻的状态, A_j 为所有可能的动作, ω 为目标网络的参数。预测 Q 估计的当前值网络使用的是最新的参数, 而预测 Q 现实的目标网络使用的是之前的参数。经过若干次迭代, 将当前值的参数复制给目标网络, 通过最小化当前 Q 值和目标 Q 值之间的均方误差来更新网络参数。DQN 的损失函数为:

$$L = E_{\varphi(S), A, R, \varphi(S')} [(Y_i^{\text{DQN}} - Q(\varphi(S_j), A_j, \omega))^2] \quad (2)$$

对式 3 中的参数 ω 求偏导, 得到的损失函数梯度为:

$$\begin{aligned} \nabla L(\omega) &= E_{\varphi(S), A, R, \varphi(S')} [(Y_j^{\text{DQN}} - Q(\varphi(S_j), A_j, \omega)) \\ &\quad \nabla Q(\varphi(S_j), A_j, \omega)] \end{aligned} \quad (3)$$

DQN 采用增加目标网络的方式降低了当前 Q 值和目标 Q 值的相关性, 从而提高了算法稳定性。DQN 训练流程如图 2 所示。

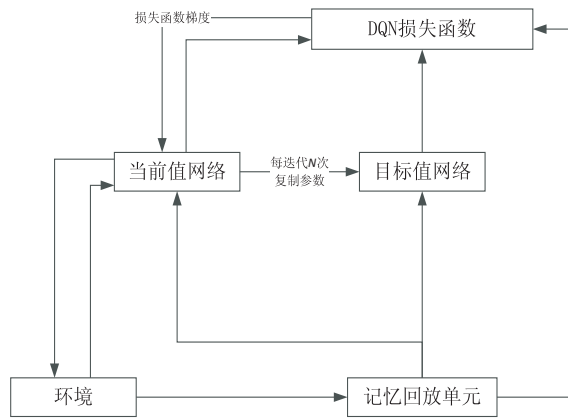


图2 DQN训练流程

1.1.3 记忆回放机制

与Q-Learning算法相比较,DQN还具备一个较大的优势,DQN采用记忆回放机制训练状态价值网络,因为状态之间的关联性较大,经验回放可以打破样本间彼此的关联性,所以可以使神经网络的训练更加收敛和稳定,可有效地提高学习效率。

1.2 双深度Q网络算法

由于DQN在进行值函数优化目标时,动作选择和策略评估都是基于目标值网络的参数,因此DQN算法在学习过程中常常会出现过高估计Q值的问题,即每次迭代选取的都是下一个状态中最大Q值所对应的动作。为了解决DQN算法在学习过程中估计值过高的问题,该文在DQN和双Q-learning算法^[13-15]的基础上提出一种双深度Q网络(double DQN,DDQN)算法^[14-15],将动作选择和策略评估分开进行,使用当前值网络的参数来选择最优动作,使用目标值网络的参数来评估该最优动作,以便估计出更加准确的Q值,获得更稳定有效的策略。DDQN的优化目标Q值为:

$$Y_j^{\text{DDQN}} = R_j + \gamma Q(\varphi(S'_j), A_j; \omega'), \quad (4)$$

$$\arg\max_a Q(\varphi(S'_j), A_j; \omega), \omega'$$

该文设计的DDQN算法详细描述如下:

(1)初始化Q网络参数 ω 及目标Q网络 Q' 的参数 $\omega' = \omega$,并初始化所有状态和动作所对应的价值Q。初始化经验回放单元D。

(2)for i to T ,进行迭代。

①初始化状态序列中第一个状态 S ,其特征向量为 $\varphi(S)$ 。

②将 $\varphi(S)$ 作为Q网络中的输入,得到所有动作所对应输出的Q值,再通过 ϵ -贪婪法选择对应的动作 A 。

③在状态 S 下选择并执行当前动作 A ,得到下一个状态的特征向量 $\varphi(S')$ 以及奖励 R ,判断是否为终止状态is_end。

④将五元组 $\{\varphi(S), A, R, \varphi(S'), \text{is_end}\}$ 存入经

验回放单元D中。

⑤令 $S = S'$ 。

⑥从经验回放单元D中采集 m 个样本,计算当前目标Q值 y_j ,其中 $j = 1, 2, \dots, m$,则:

$$y_j = \begin{cases} R_j, & \text{is_end} = \text{true} \\ R_j + \gamma Q'(\varphi(S'_j), A_j; \omega'), & \text{is_end} = \text{false} \end{cases}$$

⑦根据均方差损失函数 $\frac{1}{m} \sum_{j=1}^m (y_j - Q(\varphi(S_j), A_j; \omega))^2$,采用反向传播更新Q网络的所有参数 ω 。

⑧当 $i \% P = 1$,则更新目标Q网络参数 $\omega' = \omega$ 。

⑨判断 S' 是否为终止状态,若是终止状态则结束当前轮迭代,否则转到步骤②。

1.3 算法参数设置

1.3.1 记忆回放单元的相关参数设置

该研究着眼于任务的复杂度,将记忆回放单元的大小设置为1000,每次训练神经网络所用到的回放记忆单元的状态个数为50个。在每次获取到动作所对应的激励值后,系统将执行动作前的智能体状态、执行动作后的智能体状态、动作和奖励值插入记忆回放单元。若记忆回放单元已满,则弹出当前存储的最早的状态。每次训练神经网络时将随机在记忆回放单元中抽取50条记录以完成对神经网络的训练。

1.3.2 神经网络的初始化及更新

设隐藏层每层偏置 b 的值为0.01,连接每层网络的矩阵为 ω ,该研究使用正态分布初始化神经网络的参数。隐藏层神经网络的深度和广度过大或者过大会使拟合的效果饱和或负增长,该研究根据当前任务的复杂度设置较为适中的神经网络大小,以便该神经网络在各种情况下都适用。后续实验中由于大多数任务所经历的状态数集中在10000以内,故将隐藏层的第一层广度设置为2500,之后每一层的广度依次递减,可有效节省计算时间和空间。

2 虚拟智能对抗演练系统

为了验证双深度Q网络算法的有效性,设计了一套虚拟的智能对抗演练系统,己方作战单位智能体根据地图数据、环境数据以及兵力部署等信息,利用DDQN算法在作战环境交互试错中不断训练学习,在尽量躲避敌方火力覆盖范围的情况下,短时间确定敌军目标,并以最优的策略压制、歼灭敌军,击毁敌军基地。

2.1 系统总体架构

由于执行算法程序比较耗费计算机资源,采用客户/服务器模式可有效节省硬件开销,提高数据处理能

力与算法执行效率,系统总体架构如图 3 所示。

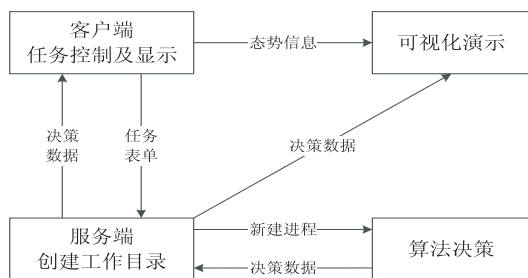


图 3 系统总体架构

客户端实现任务控制及决策推演结果折线图的显示。客户端编辑任务指令,提交任务后系统会将任务表单信息序列化,使用 HTTP 协议发送给服务端。服务端则进行数据反序列化,建立任务工作目录,并新建一个进程执行算法程序。算法决策过程同步更新到任务的工作目录下。客户端每隔 3 秒从服务器获取算法决策结果,以更新算法推演折线图。算法决策结束后,Unity3D 程序根据客户端的态势信息以及服务端任务工作目录的决策数据对智能体训练结果进行可视化。

客户端任务录入采用数据绑定的方式,将 View 层和 Model 层的数据相互绑定,Model 层的数据随着用户的键入而改变,同时重载 JavaFx 框架下的 UnaryOperator 类以实现用户对用户输入内容的限定,保证了系统的易用性与安全性。与此同时,系统采用动态创建控件的方式,以实现用户可以根据需要动态地添加多个任务对象的信息。服务端在接收到客户端传来的反序列化后的任务数据后,会更新任务记录表并创建以对应 ID 命名的文件夹,并持久化存储相关任务数据信息。

2.2 系统功能模块

智能决策系统包括任务管理模块、算法决策模块和可视化演示模块。任务管理模块实现客户端任务控制等功能;算法决策模块实现服务端创建工作目录、算法推演等功能,对输入的任务环境信息进行计算分析,最终训练得到一个具备高智能行为的智能体模型,并输出当前最优决策序列;可视化演示模块根据环境态势信息数据以及当前最优决策序列实现对算法推演结果进行可视化演示功能。系统功能模块如图 4 所示。

2.3 算法决策设计

采用 DDQN 算法对输入的数据进行处理后输出决策数据,在算法完成一次迭代后,程序会更新迭代次数文件以支持客户端显示当前迭代进度,若当前迭代结果为最优解,则更新存储算法推演结果的文件。系统决策训练的总体控制流程如图 5 所示。

其中单个智能体特训目的是在多智能体任务协同模型中,很难确保每个智能体均获取到正向奖励值的训练问题。系统每隔一定轮数便驱使单个智能体独立

完成攻击任务,这样便尽可能地使每一个智能体都可以获得正向的激励。

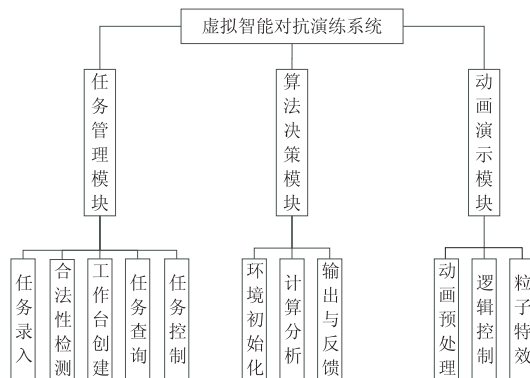


图 4 系统功能模块

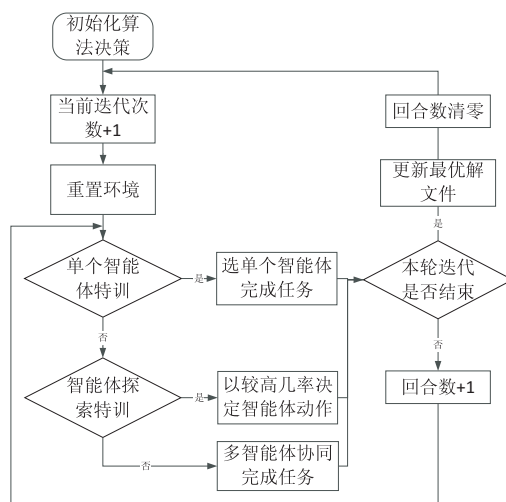


图 5 算法决策总控流程

与此同时,智能体探索特训是指系统每隔一定轮数会以较高的随机值训练智能体,以此使智能体探索更多未知的情况,从而迭代出更优的解法。

2.4 奖励函数设计

奖励函数的设计将直接影响智能体训练的优劣程度,设置合理的奖励参数对于智能决策系统尤为重要。

在本系统的决策任务中,当智能体对敌方基地实施攻击并成功时,奖励值为 40 000,攻击失败奖励值为 -1 000;当智能体对敌方作战单位实施攻击并攻击成功时,奖励值为 500,攻击失败奖励值为 -100;智能体机动过程中,若发生碰撞奖励值为 -1 000;若所处位置为敌方作战单位可攻击范围内,奖励值为 -1 000;智能体更靠近敌方基地,奖励值为 100,否则奖励值为 0。

2.5 对抗逻辑设计

系统将每个智能体定义为一个独立的训练对象,每个智能体具备单独的神经网络,不同的智能体共用同一套环境以确保可以协同作战。而敌方作战单位则采用固定脚本方式来控制其行为,通过调整敌方作战单位火力密度、抗毁伤能力、攻击范围以及地图数据等方法改变战场态势,达到智能体在不同的任务态势下

进行训练的目的。

2.6 输入输出设计

该系统输入为地图、智能体、敌方作战单位等元素的初始状态数据,每个智能体的状态定义如表1所示。

表1 智能体状态数组定义

序号	状态组下标	解释
1	0	基地毁伤值
2	1	该智能体横坐标
3	2	该智能体纵坐标
4	3	该智能体毁伤值
5	$3+3*i+1(i \geq 0)$	敌方第 <i>i</i> +1个作战单位横坐标
6	$3+3*i+2(i \geq 0)$	敌方第 <i>i</i> +1个作战单位纵坐标
7	$3+3*i+3(i \geq 0)$	敌方第 <i>i</i> +1个作战单位毁伤值

输出包括智能体动作数据文件及算法迭代结果文件。该系统中的动作分为智能体的移动和攻击动作。移动包括上、下、左、右、左上、右上、左下、右下八个动作,由编号0~7表示。攻击动作又分为两种,其一是对基地的攻击动作,其编号为8,其二是对敌方作战单位的攻击动作,编号从9开始,总个数为敌方作战单位的数量,分别对应智能体攻击敌方各个作战单位的动作。

算法完成一次迭代后会更新迭代次数文件以支持客户端显示当前迭代进度,若当前迭代结果是最优解,则更新当前最优步数的结果文件。

2.7 可视化演示设计

可视化演示程序根据任务管理客户端传来的数据对各个对象的属性进行初始化。若存在多个种类相同的对象,则调用 Unity 的 Instantiate 方法完成对象的复制与实例化。

智能体的移动采用 Unity 的 Navigation 寻路功能实现。Navigation 寻路功能可以控制移动对象的转向、移动速度等,将小尺度的移动细节利用 Navigation 处理可以降低代码的复杂度,同时也会使演示动画更加流畅自然。

3 实验仿真结果

(1) 任务信息。

通过客户端界面新建任务,可编辑地图初始信息。任务创建后点击单个任务,客户端会再次请求服务器,获取该任务详细信息,如图6所示。

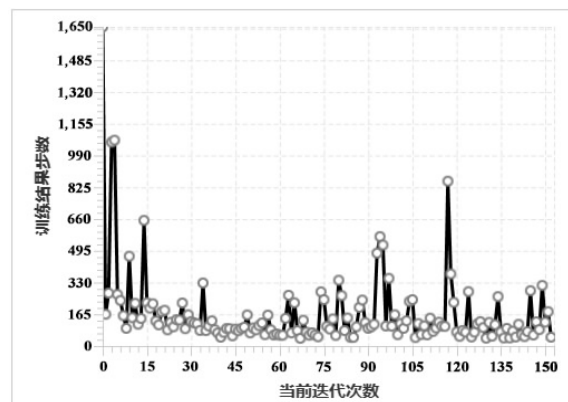
(2) 算法推演结果。

客户端每隔三秒会向服务器发送请求获取对应任务的演算情况,以刷新算法演算过程的迭代折线图。

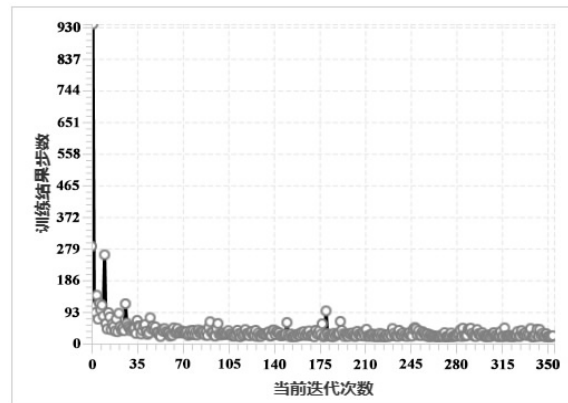
DQN 算法训练折线图如图7(a)所示。

任务列表	地图大小: 50*50 基地位置: (40,40) 基地大小: 1*1
任务0	
任务1	智能体0: 初始坐标:(0,0) 生命值: 1
任务2	智能体1: 初始坐标:(0,2) 生命值: 1
任务3	智能体2: 初始坐标:(0,4) 生命值: 1
任务4	智能体3: 初始坐标:(0,6) 生命值: 1
任务5	障碍物0: 坐标:(45,45)
任务6	障碍物1: 坐标:(45,44)
任务7	障碍物2: 坐标:(45,43)
任务8	障碍物3: 坐标:(45,42)
	障碍物5: 坐标:(45,40)
任务7	敌方作战单位0: 坐标:(39,40) 生命值: 1 攻击距离: 2 攻击方向: 下
任务8	敌方作战单位1: 坐标:(40,39) 生命值: 1 攻击距离: 2 攻击方向: 上

图6 任务详细信息界面



(a)DQN 算法



(b)DDQN 算法

图7 DQN 和 DDQN 算法训练折线图

相同的环境态势,更换 DDQN 算法重新对神经网络进行训练,DDQN 算法训练折线图如图7(b)所示。

在初始态势相同的情况下,分别使用 DQN 算法和 DDQN 算法对神经网络依次进行训练,训练结束后,通过对两个算法的训练折线图进行比较,可以观察到 DDQN 算法训练迭代到200余次,步数稳定在40步左右,相反,DQN 算法训练结果步数上下浮动较大,训练结果不能够有效收敛。

实验结果显示,DDQN 算法较 DQN 算法更稳定有效,更适用于该研究。

(3) 可视化演示。

算法演算结束后, 执行 Unity 程序对演算结果进行可视化, 演示中第一人称视角如图 8 所示。

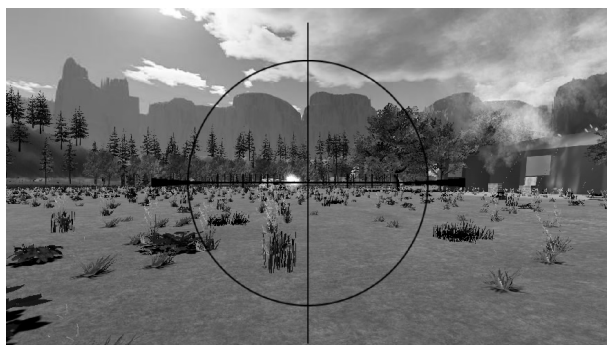


图 8 第一人称视角

4 结束语

该文研究了基于强化学习的智能决策算法, 并在虚拟智能对抗演练系统中进行了验证。验证系统集成了客户/服务端、可视化演示和深度强化学习算法, 智能体通过训练学习能够在短时间内确定目标, 并以较快的时间压制、歼灭目标, 同时避免在敌方火力覆盖范围内行动, 有效降低了智能体的毁伤程度。该文设计的 DDQN 决策算法有望在策略游戏、对抗演练、任务方案评估等领域发挥作用。

同时, 该算法还有进一步的完善空间, 在多智能体任务协同中没有将整个训练环境视为一个训练对象, 而是将每个智能体定义为一个独立的训练对象, 使各个智能体共用同一个训练环境来达到任务协同的目的。这种方法虽然大幅提高了计算速度, 但是会导致某些智能体训练不充分而使决策结果接近最优值却无法达到最优值。此外, 对于复杂的任务环境, 需要考虑针对于不同决策目标的诸多因素。

参考文献:

[1] 李琛, 黄炎焱, 张永亮, 等. Actor-Critic 框架下的多智能体决策方法及其在兵棋上的应用[J]. 系统工程与电子技术, 2021, 43(3): 755-762.

- [2] 李腾, 曹世杰, 尹思薇, 等. 应用 Q 学习决策的最优攻击路径生成方法[J]. 西安电子科技大学学报: 自然科学版, 2021, 48(1): 160-167.
- [3] 冯树民, 黄秋菊, 张宇, 等. 驾驶人“感知-决策-操控”行为模型[J]. 交通运输系统工程与信息, 2021, 21(1): 41-47.
- [4] 陈建凯, 王鑫, 何强, 等. 区间值属性的单调决策树算法[J]. 模式识别与人工智能, 2016, 29(1): 47-53.
- [5] 王辉, 宋昌统. 基于自适应状态聚集 Q 学习的移动机器人动态规划方法[J]. 计算机测量与控制, 2014, 22(10): 3419-3422.
- [6] 吴宜珈, 徐鹏. 基于强化学习的美军指控系统的发展及启示[J]. 火力与指挥控制, 2020, 45(10): 8-11.
- [7] 雷莹, 许道云. 一种合作 Markov 决策系统[J]. 计算机技术与发展, 2020, 30(12): 8-14.
- [8] JIANG Lan, HUANG Hongyun, DING Zuohua. Path planning for intelligent robots based on deep Q-learning with experience replay and heuristic knowledge[J]. IEEE/CAA Journal of Automatica Sinica, 2020, 7(4): 1179-1189.
- [9] 赵冬斌, 邵坤, 朱圆恒, 等. 深度强化学习综述: 兼论计算机围棋的发展[J]. 控制理论与应用, 2016, 33(6): 701-717.
- [10] 刘全, 翟建伟, 章宗长, 等. 深度强化学习综述[J]. 计算机学报, 2018, 41(1): 1-27.
- [11] 刘建伟, 高峰, 罗雄麟. 基于值函数和策略梯度的深度强化学习综述[J]. 计算机学报, 2019, 42(6): 1406-1438.
- [12] DURYEA E, GANGER M, HU W. Exploring deep reinforcement learning with multi Q-learning[J]. Intelligent Control and Automation, 2016, 7(4): 129-144.
- [13] DUAN Xiaoyu, YING Shi, YUAN Wanli, et al. QLLog: a log anomaly detection method based on Q-learning algorithm[J]. Information Processing & Management, 2021, 58(3): 102540.
- [14] 潘耀宗, 张健, 杨海涛, 等. 战机自主作战机动双网络智能决策方法[J]. 哈尔滨工业大学学报, 2019, 51(11): 144-151.
- [15] 吴金金, 刘全, 陈松, 等. 一种权重平均值的深度双 Q 网络方法[J]. 计算机研究与发展, 2020, 57(3): 576-589.