北京理工大學

本科生毕业设计（论文）

基于深度强化学习的自动驾驶避障技术研究

Research on Automatic Driving Obstacle Avoidance Based on Deep Reinforcement Learning

学　　院：自动化学院

专　　业：自动化

学生姓名：黄宸睿

学　　号：1120181506

指导教师：宋春雷

2022 年 5 月 16 日

原创性声明

本人郑重声明：所呈交的毕业设计（论文），是本人在指导老师的指导下独立进行研究所取得的成果。除文中已经注明引用的内容外，本文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。

特此申明。

本人签名：　　日 期：　　年　月　日

关于使用授权的声明

本人完全了解北京理工大学有关保管、使用毕业设计（论文）的规定，其中包括：①学校有权保管、并向有关部门送交本毕业设计（论文）的原件与复印件；②学校可以采用影印、缩印或其它复制手段复制并保存本毕业设计（论文）；③学校可允许本毕业设计（论文）被查阅或借阅；④学校可以学术交流为目的，复制赠送和交换本毕业设计（论文）；⑤学校可以公布本毕业设计（论文）的全部或部分内容。

本人签名：　　日 期：　　年　月　日

指导老师签名：　　日 期：　　年　月　日

基于深度强化学习的自动驾驶避障技术研究

摘　要

本文……。

关键词：北京理工大学；本科生；毕业设计（论文）

Research on Automatic Driving Obstacle Avoidance Based on Deep Reinforcement Learning

Abstract

In order to study……

Key Words: BIT；Undergraduate；Graduation Project (Thesis)

目　录

第 1 章 绪论

1.1 研究背景与意义

自动驾驶系统的决策模块需要先进的决策算法保证安全性、智能性、有效性。目前传统算法的解决思路是以价格昂贵的激光雷达作为主要传感器，依靠人工设计的算法从复杂环境中提取关键信息, 根据这些信息进行决策和判断。该算法缺乏一定的泛化能力，不具备应有的智能性和通用性。深度强化学习的出现有效地改善了传统算法泛化性不足的问题, 这给智能驾驶领域带来新的思路。

强化学习 (Reinforcement Learning, RL) 通过与环境交互，学习状态到行为的映射关系。如图1-1所示，在一个离散时间序列 $t = 0, 1, 2, \ldots$ 中，智能体需要完成某项任务。在每一个时间 t，智能体都能从环境中接受一个状态 S_t，并通过动作 a_t 与环境继续交互，环境会产生新的状态 S_{t+1}，同时给出一个立即回报 r_{t+1}。如此循环下去，智能体与环境不断地交互，从而产生更多数据（状态和回报），并利用新的数据进一步改善自身的行为。

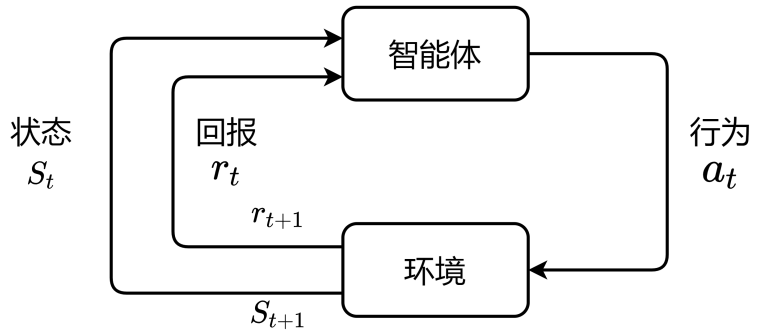

图 1-1 强化学习原理

目前，强化学习在策略选择的理论和算法方面已经取得了很大的进步，然而直接从高维感知输入（如图像、语音等）中提取特征，学习最优策略，对强化学习来说依然是一个挑战。

深度强化学习 (Deep Reinforcement Learning, DRL) 结合了深度神经网络和强化学习的优势, 可以用于解决智能体在复杂高维状态空间中的感知决策问题[1][2]。2016 年，基于深度强化学习和蒙特卡洛树搜索的 AlphaGo 击败了人类顶尖职业棋手，引起了全世界的关注[3]。2017 年, DeepMind 在《Nature》上公布了最新版 AlphaGo 论

文, 介绍了更强的围棋人工智能: AlphaGo Zero。它不需要人类专家知识, 只使用纯粹的深度强化学习技术和蒙特卡罗树搜索, 经过 3 天自我对弈就以 100 比 0 击败了上一版本的 AlphaGo。AlphaGo Zero 证明了深度强化学习的强大能力, 也必将推动以深度强化学习为代表的人工智能领域的进一步发展。基于深度强化学习在棋局与游戏上的成功，最近的研究大多注重于深度强化学习在各个领域中的扩展与应用。

综上所述，深度强化学习方法与深度神经网络强大的特征提取能力相结合，可以实现端到端的控制与决策，具有较强的通用性。通过网络自主学习的方式，减少了对系统动力学建模与数学解析的复杂度，相比于传统依据规则的决策方式更加便捷。随着人工智能的兴起和强化学习在轮式机器人相关领域的成功应用，基于深度强化学习的自动驾驶决策与控制方法为自动驾驶决策提供了新的解决方案，这使得对其研究更具有理论指导意义和实际应用价值。

1.2 国内外研究现状

自动驾驶系统 (Automated driving systems, ADS) 保证了安全、舒适和高效的驾驶体验，但近年来的研究表明，除非进一步提高最新技术的鲁棒性，否则自动驾驶系统的潜力便无法完全发挥[4]。目前，大多数的自动驾驶系统将大量的自动驾驶任务划分为若干个子类别，并在各个模块上采用一系列传感器和算法，算法流程如图1-2所示。

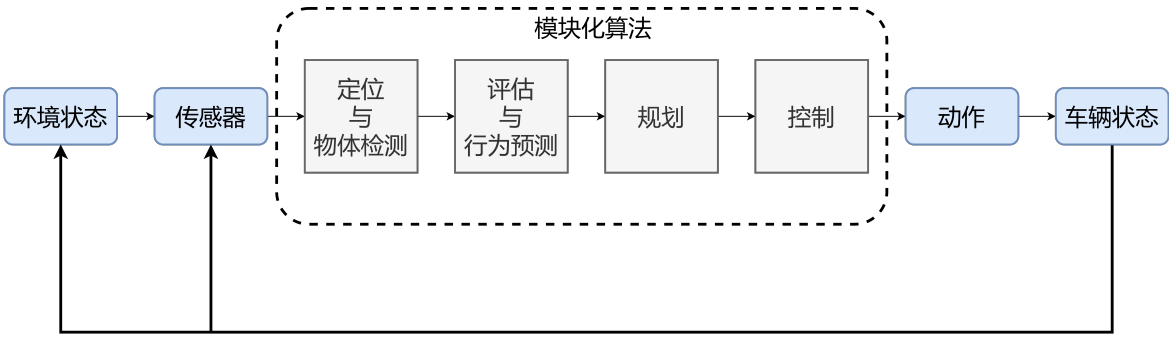

图 1-2 模块化方法流程图

最近，端到端方法开始作为模块化方法的替代出现。端到端驾驶 (End-to-end driving) 又被称作直接感知 (direct perception)[5]，即直接从感知输入产生动作，其算法流程图如图1-3所示。此处的动作可以是方向盘和踏板的连续操作，也可以是一组离散的动作，例如加速和转向。目前有三种主要的端到端方法：直接监督深度学

习[6][7]、神经进化[8](neuroevolution) 和最近的深度强化学习[9]。

图 1-3 端到端方法流程图

深度学习和强化学习的发展使得直接从原始的数据中提取高水平特征进行感知决策变成可能。深度学习起源于人工神经网络。早期研究人员提出了多层感知机的概念, 并且使用反向传播算法优化多层神经网络, 但是由于受到硬件等资源的限制, 神经网络的研究一直没有取得突破性进展。最近几年, 随着计算资源的性能提升和相应算法的发展, 深度学习在人工智能领域取得了一系列重大突破, 包括图像识别、语音识别、自然语言处理等。深度学习由于其强大的表征能力和泛化性能受到众多研究人员的关注, 相关技术在学术界和工业界都得到了广泛的研究与应用。

深度强化学习由数据驱动，不需要构造系统模型，具有很强的自适应能力。普林斯顿大学的 Chen 等使用深度学习算法, 根据摄像头采集的图像数据预测目标的距离, 同时输出操作指令[5]。斯坦福大学的 Zhu 等使用暹罗网络结构, 同时输入当前视角图像和目标物体图像, 并且使用残差网络模型提取特征. 通过 A3C 算法进行训练, 成功控制小车在虚拟场景和现实场景中到达指定地点[10]。国内的 Zhao 等使用深度强化学习算法和注意力机制, 实现了智能驾驶领域车辆的高精度分类[2]。Zhu 基于 TORCS 的真实物理变量, 使用高斯过程强化学习算法 PILCO（Probabilistic Inference for Learning Control）离线训练控制器, 实现车道保持。同时以图像为输入, 使用深度学习算法感知环境信息, 预测本车距离车到中央线距离、偏航角、道路曲率等。最终将 RL 的控制策略和 DL 的特征预测结合, 实现基于图像的车道保持。

以上研究仅由大量的数据驱动而未单独考虑模型的影响，针对自动驾驶避障技术的模型问题，德克萨斯大学奥斯汀分校的 Chen 等提出《Learning to drive from a world on rails》的假设[11]，采用世界模型与车辆模型解耦的方式建立一个前向模型来

帮助改进策略；David Ha 等提出生成循环神经网络以无监督的方式快速训练，通过压缩时空表征来模拟流行的强化学习环境[12]。

针对具体的深度强化学习算法，2012 年，Lange 等人最早在插槽赛车上使用深度强化学习算法并取得了良好的控制效果[13]。2013 年，由 DeepMind 团队提出的 DQN（Deep Q-Network）算法利用深度卷积神经网络直接学习 Atari2600 种游戏的高维度图像，从输入中提取环境的高效描述，来近似最优动作-状态函数，从而习得成功策略[14]。2015 年，Hasselt 等人发现传统的 Q-learning 和 DQN 方法都会普遍过高估计行为值函数 Q 值，存在过优化的问题，为了解决值函数过估计的问题，Hasselt 提出了 Double DQN 方法，将行为选择和行为评估采用不同的值函数实现，降低了过估计的误差[15]。2016 年，Ziyu Wang 提出 Dueling DQN 方法，把 Q 网络的结构显式地约束成跟动作无关的状态值函数 $V(s)$ 与在状态 s 下各个动作的优势函数 $A(s,a)$ 之和，使得 DQN 训练更容易，收敛速度更快，避免因为 Q 值的量级大而引起的结果不稳定问题[16]。2017 年，Zong 等人使用深度确定性策略梯度 DDPG 算法对智能体的加速度和转向控制进行训练，以实现自主避障，并在开源赛车模拟器 (TORCS) 环境中进行了测试，结果表明通过一段时间自主学习，无人驾驶汽车能学会复杂的避障换道行为[17]。该算法是将 DQN 和 Actor-Critic 算法相结合，使用深度神经网络来逼近值函数和策略函数，虽然，DDPG 算法实现了连续状态、动作空间下的强化学习，然而算法参数并不易确定，其超参数往往必须针对不同的问题进行仔细设置才能获得良好的训练结果。

随着研究的不断深入，深度强化学习算法在自动驾驶控制策略的应用范围越来越广，解决的问题也越来越复杂。这充分说明了深度强化学习算法应用于自动驾驶决策控制领域的可行性和有效性。本文在调研和分析大量文献的基础上基于深度强化学习算法设计自动驾驶决策控制算法，通过对比多种深度强化学习算法及其改进算法，验证在抽象环境与高维环境下深度强化学习算法的有效性和准确度。

1.3 本文主要研究内容及章节安排

本文以 DQN 算法作为基础，探究 DQN 及其改进算法 Double DQN 和 Dueling DQN 在两种不同仿真环境（Highway-Env、Metadrive）中对自动驾驶决策及控制产生的效果。通过在仿真环境下的反复训练和学习，训练的模型在多种仿真环境下均具有较好的鲁棒性，证明了 DQN 及其改进算法在自动驾驶决策及控制方面的有效性

和可行性。

第 1 章绪论。介绍了该课题的研究背景和意义，对强化学习在自动驾驶决策方面的发展现状进行了介绍，对深度强化学习特别是 DQN 算法的研究现状进行了详细的分析说明，在章节的最后通过分析总结参考文献得出了本文的研究方向和研究内容。

第 2 章强化学习相关算法理论分析。通过对强化学习算法原理的分析，引出了适用于无模型的 Q-learning 算法和 DQN 算法，同时，针对于 DQN 算法的不足和缺陷，介绍了 Double DQN 算法和 Dueling DQN 算法的原理，并对几种算法的适用场景进行了分析。

第 3 章基于 DQN 算法的自动驾驶避障算法设计。通过对两种仿真环境（Highway-Env、Metadrive）的分析和对比，介绍二者的状态值、动作值与奖励算法的详细情况。将 DQN 及其改进算法作用于两种仿真环境，同时改进 Q 网络的网络结构及超参数，使其完成自动驾驶决策和控制两方面的任务要求。

第 4 章实验验证。为了验证算法的可行性和有效性，进行了两种仿真环境中的仿真实验，并对具体的实验内容进行了分析与设计，对最后的实验结果进行了详细的分析与改进。

文章的最后进行了总结，对本文完成的主要工作和取得的主要成就进行了概述，对本文的创新点进行了阐述，同时指出了本文研究的不足和后续研究的重点内容。

第 2 章 强化学习相关算法理论分析

如图1-1所示，强化学习的基本原理在 1.1 节中已有提及，在此不再赘述。强化学习强调智能体与环境不断地交互，从而产生更多数据（状态和回报），并利用新的数据进一步改善自身的行为。智能体不会被告知在当前状态下，应该采取哪一个动作，只能通过不断尝试，依靠环境对动作的反馈改善自己的行为。经过数次迭代后，智能体最终能学到完成相应任务的最优动作（策略）。

本章通过对强化学习以及相关算法的理论分析，对基于值函数（Value Based）的 Q-learning 算法及 DQN 算法进行详细介绍，对 DQN 算法及其改进算法的优劣进行比较，选择最适合解决自动驾驶决策控制的方法。

2.1 强化学习算法

强化学习包括智能体和环境两大对象。智能体又称为学习者或玩家，环境是指与智能体交互的内部。智能体由策略、值函数、模型三个组成部分中的一个或多个组成。下文将介绍强化学习智能体的各个组成部分与强化学习问题求解的目标，由此引出基于值函数的强化学习方法。

2.1.1 强化学习算法的组成

（1）策略：策略是决定智能体行为的机制，是状态到行为的映射，用 $\pi(a|s)$ 表示，它定义了智能体在各个状态下的各种可能的行为及概率。

$$\pi(a|s) = P(A_t = a|S_t = s) \tag{2-1}$$

策略分为两种，确定性策略和随机性策略。确定性策略根据智能体具体状态输出一个确切的动作，而随机性策略根据状态输出智能体每个动作的概率，输出值为一个概率分布。一个策略完整定义了智能体在各个状态下的各种可能的动作及其概率大小。策略仅和当前状态有关，与历史信息无关。策略就是用来描述各个不同状态下执行各个不同行为的概率，同一时刻某一确定的策略是静态的，与时间无关，但是智能体可以随着时间更新策略。

（2）值函数：值函数代表智能体在给定状态下采取某个行为的好坏程度。这里的好坏用未来的期望回报表示，而回报和采取的策略有关，所有值函数的估计都是基

于给定的策略进行的。值函数（或称为回报）用 G_t 表示，也称为“收益”或“奖励”。

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{2-2}$$

其中折扣因子 γ（衰减系数）体现了未来的回报在当前时刻的价值比例，在 $k+1$ 时刻获得的回报 R 在 t 时刻体现出的价值是 $\gamma^k R$。γ 接近 0 表示趋向于当前利益；γ 接近 1 表示偏向于长远期的利益。

值函数分为状态值函数与状态行为值函数，二者都与回报有关。

状态值函数 $V_\pi(s)$ 表示从状态 s 开始，遵循当前策略 π 所获得的期望回报。

$$\begin{aligned} V_\pi(s) &= E_\pi[G_t|S_t = s] \\ &= E_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots|S_t = s] \end{aligned} \tag{2-3}$$

值函数的另一个类别是状态行为值函数 $Q_\pi(s,a)$，也称为行为值函数。该函数表示针对当前状态 s 执行某一具体行为 a 后，继续执行策略 π 所获得的期望回报；也表示遵循策略 π 时，对当前状态 s 执行行为 a 的价值大小。

$$\begin{aligned} Q_\pi(s,a) &= E_\pi[G_t|S_t = s, A_t = a] \\ &= E_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots|S_t = s, A_t = a] \end{aligned} \tag{2-4}$$

（3）模型：在强化学习任务中，模型是智能体对环境的一个建模。环境模型至少要解决两个问题，一是预测状态转移概率 $P^a_{ss'}$，即预测在状态 s 上采取行为 a 后，下一个状态 s' 的概率分布；二是预测在状态 s 上采取行为 a 后可能获得的立即回报 R^a_s。

$$\begin{aligned} P^a_{ss'} &= P(S_{t+1} = s'|S_t = s, A_t = a) \\ R^a_s &= E[R_{t+1}|S_t = s, A_t = a] \end{aligned} \tag{2-5}$$

根据智能体在与环境交互的过程中是否建立环境的模型，强化学习可以分为两个大类，即有模型方法和无模型方法。一般的模型已知问题，就是智能体获得了确切的状态转移概率 $P^a_{ss'}$ 和回报 R^a_s。

在后续章节的仿真环境介绍中，由于车辆的决策控制较难采用已知模型（如动力学模型、运动学模型）进行刻画，故均使用“无模型方法”的假设，即智能体在整

个训练过程中不需要对环境模型进行建模，直接使用学习得到的经验进行策略的优化。

根据以上三点概念，可以通过建立状态值估计的方法或建立策略估计的方法来解决强化学习问题。基于值函数的方法在求解强化学习的目标时只估计状态值函数，不估计策略函数，最优策略函数在对值函数进行迭代求解时，通过状态值函数间接得到。针对自动驾驶避障问题，由于我们较难估计车辆状态与行为之间的映射（即策略函数），但可以估计车辆在采取某个动作时，车辆状态发生的变化和环境回报（即状态值函数），所以采取基于值函数的方法更加符合解决自动驾驶避障问题所需要达成的目标。

2.1.2 强化学习算法的求解

求解强化学习问题的目标是求解每个状态下的最优策略，即在运行过程中接收的累计回报最大。为了获取更高的回报，智能体在进行决策时要考虑立即回报，也要考虑后续状态的回报。解决强化学习问题一般需要两步，将实际场景抽象成一个数学模型，然后去求解这个数学模型，找到使得累计回报最大的解。

第一步：构建强化学习的数学模型——马尔科夫决策（Markov Decision Process, MDP）模型。

不论涉及的智能体结构、环境和交互细节多么复杂，此类交互问题都能简化为三个信号：智能体的行为、环境的状态、环境反馈的回报。具体到实验中，便是仿真环境根据智能体做出的行为产生的回报和改变的状态。马尔科夫决策模型可以有效表示实际的强化学习问题，这样解决强化学习问题的问题就转化为求解马尔科夫决策模型的最优解。

第二步：求解马尔科夫决策模型的最优解。

求解马尔科夫决策问题，是指求解每个状态下的行为，使得累计回报最大。对于环境已知的情况可以选用基于模型的方法如动态规划法；基于未知的情况选择无模型方法如时序差分法；对于状态空间、动作空间连续的场景可以采用值函数逼近法等等。根据 2.1.1 节的分析，如何求解马尔科夫决策模型便是具体的算法与网络结构，下文将对基于值函数（Value Based）的 Q-learning 算法及 DQN 算法进行详细介绍。

2.2 Q-learning 算法

在介绍 Q-learning 算法之前，首先对时序差分方法进行简单的介绍。时序差分学习最早由 A.Summuel 在跳棋算法中提出，1988 年，Sutton 证明了时序差分方法在最小均方误差（MSE）上的收敛性[18]，之后时序差分方法被广泛应用在无法产生完整轨迹的无模型强化学习问题上。时序差分（TD）方法是无模型方法，无法获得当前状态的所有后续状态及回报，仅能通过采样学习轨迹片段，用下一状态的预估状态价值更新当前的状态价值。

Q-learning 算法属于离线策略时序差分（TD）问题，最早由 Watkins 和 Dayan 在 1992 年提出[19]，其任务是通过不断地学习，不断的更新状态-动作值函数 $Q(s,a)$，从而得出最优策略。根据 2.1.2 节中所提到的，求解强化学习问题的目标是求解每个状态下的最优策略，Q-learning 算法在更新一个状态-动作值函数（以下简称 Q 值）时，采用的不是遵循当前策略（行为策略 μ）的下一个状态-动作对的 Q 值，而是待评估策略（目标策略 π）产生的下一个状态-动作对的 Q 值。更新公式如下：

$$Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha(R_{t+1} + \gamma Q(S_t,A') - Q(S_t,A_t)) \tag{2-6}$$

其中 α 称为学习率，γ 称为折扣因子，TD 目标 $R_{t+1} + \gamma Q(S_t,A')$ 是基于目标策略 π 产生的行为 A' 得到的 Q 值和一个立即回报的和。在 Q-learning 算法中，行为策略 μ 是基于原始策略的 ϵ-贪心策略，保证取得经历足够丰富的新状态。目标函数 π 是单纯的贪心策略，通过最大化 TD 目标来保证策略最终收敛到最佳策略。

Q-learning 算法处理有限的状态空间与有限的动作空间的问题，状态值和动作放在 Q 表中，值函数能够表示为一个数组。但在实际情况下，强化学习面临的问题的状态空间往往是连续的，无法用表格的方法准确列出每一种状态对应的 Q 值大小，故需要进行对 Q 值进行非线性的逼近。下文的 DQN 就属于这样的方法。无论是 Q-learning 或是 DQN，均采用了目标策略 π 产生的下一个状态-动作对的 Q 值对原有的 Q 值进行更新，这是时序差分法的一般思想。

Q-learning 的算法流程如下：

Algorithm 1: Q-learning 算法

Input: 环境 E，状态 S，动作 A，折扣因子 γ，学习率 α，初始化行为值函数 $Q(s,a)=0$

for *k = 0,1,2,...,m* **do**

初始化状态 s;

for *t=0,1,2,...* **do**

在 E 中通过 π 的 ϵ-贪心策略采取行为 a;

r,$s^{'}=$ 在 E 中执行动作 a 产生的回报和转移的状态;

$Q(S_t,A_t)\leftarrow Q(S_t,A_t)+\alpha(R_{t+1}+\gamma Q(S_t,A^{'})-Q(S_t,A_t))$;

$s\leftarrow s^{'}$;

end

end

$\pi^{*}(s)=\arg\max_{a\in A}Q(s,a)$;

Output: 最优策略 π^{*}

2.3 DQN 算法

DQN（Deep Q-Network）算法是建立传统强化学习算法 Q-learning 的基础上的时序差分算法，Q-learning 是离线策略时序差分法，使用 ϵ 贪心策略产生数据，利用查表法对行为值函数（Q 值）进行预测，TD 目标是 $R_{t+1}+\gamma Q(S_t,A^{'})$。DQN 算法在传统强化学习 Q-learning 的基础上，主要对其精确的查表法做了近似拟合，同时通过引入深度学习网络，对网络结构和参数更新做出了如下改进。

（1）DQN 使用深度神经网络从原始数据中提取特征，近似行为值函数（Q 值）。

当状态空间很大且连续时，无法使用查表法来求解每个状态的价值，此时可以考虑“离散”状态空间的方法来减少算力。在“离散”状态空间中，使用深度神经网络来表示行为值函数是常见的方法。对于深度神经网络，其参数是每层网络的权重及偏置，用 θ 表示，对值函数的更新等价于对参数 θ 的更新。DQN 神经网络结构如图2-1所示。

DQN 神经网络结构是三个卷积层和两个全连接层。输入为经过处理的 4 个连续的 84×84 的图像，经过卷积层和两个全连接层输出包含每一个动作的 Q 值向量。DQN 网络将高维的状态输入转换为低维的动作输出，即将图像输入转换为动作输出。

利用深度神经网络实现了数据的降维。

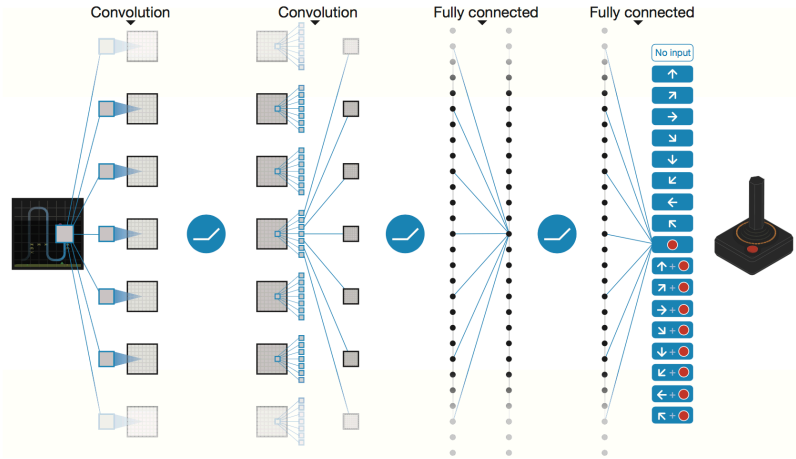

图 2-1 DQN 神经网络结构

（2）DQN 使用经历回放训练强化学习。

在使用深度神经网络进行行为值函数（Q 值）近似时，如果不对训练数据做处理，直接将当前时刻的信息进行学习训练，学习效果会出现较大偏差。由于使用神经网络的前提是数据之间独立同分布，而强化学习过程中，数据是通过与环境交互产生的，相邻数据之间高度相关。如果智能体在很长一段时间均学习相同环境下的数据，在接收到另一环境的数据后，参数会出现不稳定与大范围波动发散，求解无法收敛。针对这一问题，DQN 采用“经验回放”的方法进行解决。

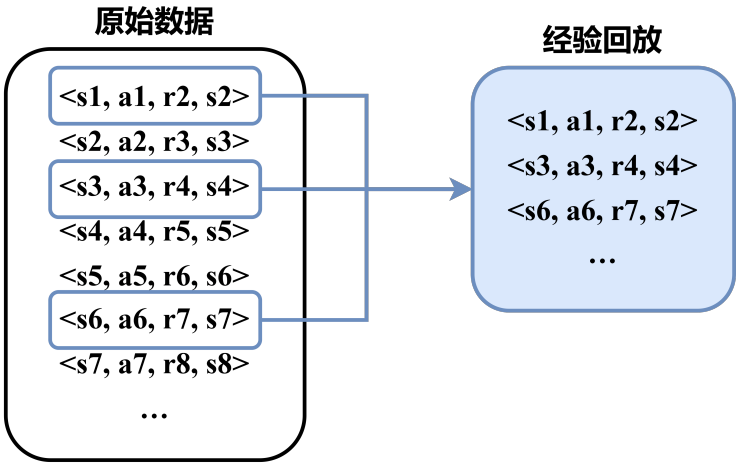

图 2-2 经验回放

经验回放最早由 Long Ji Lin 在 1993 年提出[20]，如图2-2，它在强化学习中是这

样实现的：智能体跟环境不断交互，将在环境中积累的数据存储到记忆库中。首先对环境做出探索并将 < 本时刻状态、行为、奖励、下一时刻状态 >（$< s_1, a_1, r_2, s_2 >$）作为一个事件对进行存储，每一次对神经网络中的参数进行更新时，利用均匀随机采样的方法从数据库中抽取数据，通过抽取的数据对神经网络进行训练。因为经验回放的样本是随机抽取，每次用于训练的样本不再是连续的数据，打破了数据的关联，由此可以满足神经网络的假设要求。

（3）DQN 使用单独的目标网络处理 TD 偏差。

与式2-6类似，DQN 更新神经网络的参数 θ 采用的是梯度下降法。更新公式如下：

$$\theta_{t+1} = \theta_t + \alpha(r + \gamma \max_{a'} Q(s', a'; \theta_t) - Q(s, a; \theta_t))\nabla Q(s, a; \theta_t)$$

但如（2）中提及，行为值函数（$Q(s, a; \theta_t)$）和 TD 目标值函数（$r + \gamma \max_{a'} Q(s', a'; \theta_t)$）产生的数据需要避免关联性。为解决上述问题，DQN 引入两个神经网络，一个网络固定参数专门用来产生 TD 目标，称为 TD 网络。另一个网络专门用来评估策略更新函数，逼近值函数，称为行为值函数（Q 值）逼近网络。两个网络参数的更新速率不一致，用于行为值函数（Q 值）逼近的网络参数每一步都更新；用于计算 TD 目标值的网络参数每隔固定的步数更新一次，期间保证不变。于是得到以下 DQN 网络的更新公式：

$$\theta_{t+1} = \theta_t + \alpha(r + \gamma \max_{a'} Q(s', a'; \theta_t^-) - Q(s, a; \theta_t))\nabla Q(s, a; \theta_t) \tag{2-7}$$

综合上述三点改进，DQN 算法将经验回放和设置单独的目标网络两个方面对 Q-learning 方法进行改善，使其对于更加复杂的问题和大规模神经网络更加稳定和容易收敛，DQN 的算法流程如下：

Algorithm 2: DQN 算法

Input: 环境 E，状态 S，动作 A，折扣因子 γ，学习率 α

初始化经验回放库 D 并定义容量 N;

随机初始化网络参数 θ，用 θ 初始化主网络 $Q(;\theta)$;

随机初始化网络参数 $\theta^- = \theta$，用 θ^- 初始化 TD 网络 $Q(;\theta^-)$;

for *k = 0,1,2,...,m* **do**

初始化状态 s;

for *t=0,1,2,...* **do**

在 E 中通过主网络的 ϵ-贪心策略采取行为 a（以 ϵ 概率随机选择任一随机动作，以 $1-\epsilon$ 概率选择行为值函数最大的动作，即 $a = \arg\max_{a \in A} Q(s, a; \theta)$;

在 E 中执行动作 a，返回奖励 r, 和下一时刻状态 s';

将当前事件对 $< s, a, r, s' >$ 存入经验回放库 D 中;

从经验回放库 D 中随机采样 n 个数据，进行如下算法更新:

$$q_{target} = \begin{cases} r, & end \\ r + \gamma \max Q(s', a; \theta^-), & else \end{cases}$$

$q_{next} = Q(s, a; \theta)$

$Loss = (q_{target} - q_{next})^2$;

对于主网络参数 θ 使用 $Loss$ 进行梯度下降法更新网络参数 θ;

每隔 X 步更新一次 TD 网络，$\theta^- \leftarrow \theta$;

end

end

Output: 最优网络参数 θ

2.4 改进的 DQN 算法

由式2-6与式2-7可知，不论是 Q-learning 还是 DQN，值函数的更新公式中均有最大化操作，通过最大化值函数网络的操作来选择行为。通过最大化值函数网络的操作来选择行为并进行评估，整体上使得估计的值函数比真实的值函数大，并且误差会随着动作空间的增加而增加。此时产生的过估计量往往是非均匀的，故此时值函数的过估计就会影响到最优决策，导致最终选择一个次优的动作。

为了解决 DQN 的不足，Double DQN 和 Dueling DQN 分别从网络的更新策略和网络的结构上做出了改进。

2.4.1 Double DQN 算法

为了解决值函数过估计的问题，Hasselt 提出了 Double DQN 方法[15]。传统 DQN 中，选择行为指的是选择一个动作 a，使其满足 $a = \arg\max_a Q(s', a; \theta-)$；评估行为指的是利用 a 构建 TD 目标 q_{target}，$q_{target} = r + \gamma \max Q(s', a; \theta^-)$，选择行为和评估行为用的是同一个 Q 网络及其网络参数。

Double DQN 分别采用不同的值函数来实现动作选择和动作评估。针对于传统 DQN，由于传统 DQN 已经存在两个网络 (主网络和 TD 网络)，因此不需要改变 DQN 的网络结构，只需要改变 DQN 的参数更新策略。Double DQN 和 DQN 的区别在于：

（1）首先使用主网络选择动作。

$$a = \arg\max_a Q(s', a; \theta)$$

（2）其次使用 TD 网络找到该动作对应的 Q 值，构成 TD 目标。

$$\begin{aligned} q_{target} &= r + \gamma Q(s', a; \theta^-) \\ &= r + \gamma Q(s', \arg\max_a Q(s', a; \theta); \theta^-) \end{aligned}$$

此时构成的 q_{target} 在 TD 网络中不一定是最大的，但是该值是通过每步更新的主网络选取的最优动作，可以在一定程度上避免选到被高估的次优行为。Double DQN 的其余流程均与 DQN 相似，故算法流程不再赘述。

2.4.2 Dueling DQN 算法

在许多基于视觉感知的深度强化学习的任务中，不同的状态对应的值函数 Q(s,a) 是不同的，但是在某些状态下，值函数的大小与动作无关。Baird 在 1993 年提出将 Q 值分解为价值（Value）和优势（Advantage）[21]，即 $Q(s,a) = V(s) + A(s,a)$。$V(s)$ 是在 s 状态下所有行为值函数（Q 值）关于行为概率的期望，即所有可能行为对应的 Q 值乘以该行为所对应的概率之和。$A(s,a) = Q(s,a) - V(s)$，表示行为值函数相比于当前状态值函数的优势，即在这个状态下各个动作的优劣程度。

基于 Baird 的思想，将 DQN 用于竞争网络，就有了 Dueling DQN 算法的原理，

图2-3清晰地给出了 Dueling DQN 和 DQN 的网络结构差异。

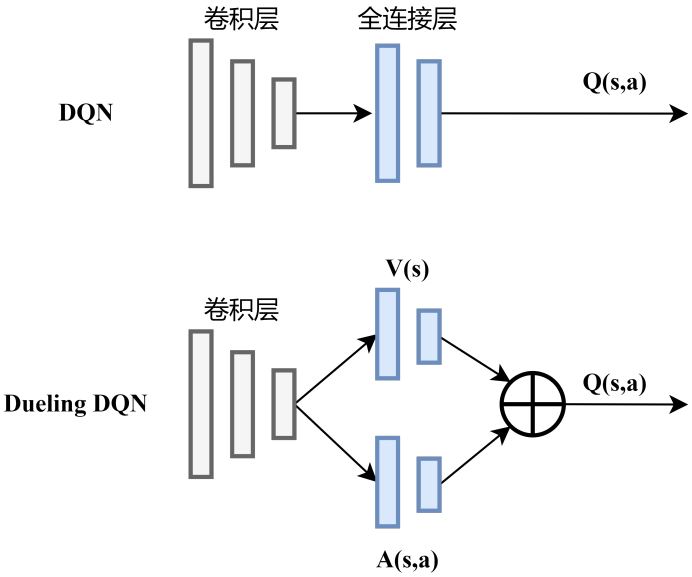

图 2-3 DuelingDQN 网络结构

Dueling DQN 将卷积层提取的抽象特征分流到两个支路上，两个支路所采用的全连接层结构相同。两个支路分别输出状态值函数 $V(s)$ 和优势值函数 $A(s,a)$，聚合函数将两个支路合并为 Q 函数：

$$Q(s,a;\alpha,\beta)=V(s;\beta)+A(s,a;\alpha)$$

但此式存在一个无法识别的问题，在确定的 Q 下，V 和 A 有无数可能，故需要对 A 值做出限定，强制令所有选择贪婪动作的优势函数为 0（认为没有比选择贪婪动作的方法更优的选项了）。添加约束条件，此时 Dueling DQN 的行为值函数为：

$$Q(s,a;\alpha,\beta)=V(s;\beta)+A(s,a;\alpha)-\max_{a'\in A}A(s,a';\alpha) \tag{2-8}$$

在实际中，一般使用优势函数的平均值代替上述最优值，虽然平均值改变了优势函数的值，但它可以保证缩小 Q 值的范围，去除多余的自由度，从而提高算法的稳定性。

$$Q(s,a;\alpha,\beta)=V(s;\beta)+A(s,a;\alpha)-\frac{1}{|A|}\sum_{a'\in A}A(s,a';\alpha) \tag{2-9}$$

由于 Dueling DQN 与传统 DQN 有相同的输入输出，只需要改变网络结构和前

向传播的公式，除此之外算法的处理流程和 DQN 是相同的。

2.5 本章小结

本章介绍了强化学习算法的组成及其马尔科夫决策模型的求解，通过对基于值函数（Value Based）的 Q-learning 算法及 DQN 算法的详细分析，由于自动驾驶决策控制任务无法进行精确的“查表”预知，需要进行神经网络的拟合，故采取基于 DQN 及其改进算法 Double DQN 和 Dueling DQN 算法完成自动驾驶决策控制任务。同时，根据输入数据的抽象程度，下一章节将对三种 DQN 算法进行决策与控制方面的设计与探究，从而得到 DQN 及其改进型的对比实验效果。

第 3 章 基于 DQN 算法的自动驾驶避障算法设计

3.1 仿真环境

3.1.1 Highway-Env

Highway-Env 的状态值设计

111

Highway-Env 的动作值设计

Highway-Env 的奖惩值函数设计

3.1.2 Metadrive

Metadrive 的状态值设计

Metadrive 的动作值设计

Metadrive 的奖惩值函数设计

3.2 基于 DQN 算法的控制策略设计

3.2.1 网络结构设计

3.3 基于 DQN 算法的控制器设计

第 4 章 实验验证

4.1 Highway-Env

4.2 Metadrive

4.3 改进的网络结构

4.4 结果对比

结　论

本文结论……。

结论作为毕业设计（论文）正文的最后部分单独排写，但不加章号。结论是对整个论文主要结果的总结。在结论中应明确指出本研究的创新点，对其应用前景和社会、经济价值等加以预测和评价，并指出今后进一步在本研究方向进行研究工作的展望与设想。结论部分的撰写应简明扼要，突出创新性。阅后删除此段。

结论正文样式与文章正文相同：宋体、小四；行距：22 磅；间距段前段后均为 0 行。阅后删除此段。

参考文献

[1] 唐振韬, 邵坤, 赵冬斌, 等. 深度强化学习进展: 从 AlphaGo 到 AlphaGo Zero[J]. 控制理论与应用, 2017, 34(12): 18.
[2] Li Y. Deep Reinforcement Learning: An Overview[J]., 2017.
[3] Babbar S. Review - Mastering the game of Go with deep neural networks and tree search[J]., 2017.
[4] Yurtsever E, Lambert J, Carballo A, et al. A Survey of Autonomous Driving: Common Practices and Emerging Technologies[J]., 2019.
[5] Chen C, Seff A, Kornhauser A, et al. DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving[J]. IEEE, 2015: 2722-2730.
[6] Pomerleau D A. Alvinn: An Autonomous Land Vehicle in a Neural Network[J]. Morgan Kaufmann Publishers Inc., 1989.
[7] Bojarski M, Testa D D, Dworakowski D, et al. End to End Learning for Self-Driving Cars[J]., 2016.
[8] Baluja S. Evolution of an artificial neural network based autonomous land vehicle controller[J]. Systems Man & Cybernetics Part B Cybernetics IEEE Transactions on, 1996, 26(3): 450-463.
[9] Sallab A, Abdou M, Perot E, et al. Deep Reinforcement Learning framework for Autonomous Driving[J]. Electronic Imaging, 2017, 2017(19): 70-76.
[10] Zhu Y, Mottaghi R, Kolve E, et al. Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning[J]., 2016.
[11] Chen D, Koltun V, Krhenbühl P. Learning to drive from a world on rails[J]., 2021.
[12] Ha D, Schmidhuber J. Recurrent World Models Facilitate Policy Evolution[J]., 2018.
[13] Lange S, Riedmiller M, Voigtlander A. Autonomous reinforcement learning on raw visual input data in a real world application[C]//International Joint Conference on Neural Networks. 2012.
[14] Chung J. Playing Atari with Deep Reinforcement Learning[J]. Computer Science, 2013.
[15] Hasselt H V, Guez A, Silver D. Deep Reinforcement Learning with Double Q-learning[J]. Computer ence, 2015.
[16] Freitas N D, Lanctot M, Hasselt H V, et al. Dueling network architectures for deep reinforcement learning[J]., 2016.
[17] Zong X, Xu G, Yu G, et al. Obstacle Avoidance for Self-Driving Vehicle with Reinforcement Learning[J]. SAE International Journal of Passenger Cars - Electronic and Electrical Systems, 2017, 11(1): 07-11-01-0003-.
[18] Sutton R, Barto A. Reinforcement Learning:An Introduction[M]. Reinforcement Learning:An Introduction, 1998.
[19] Watkins C, Dayan P. Technical Note: Q-Learning[J]. Machine Learning, 1992, 8(3-4): 279-292.
[20] Lin L J. Reinforcement Learning for Robots Using Neural Networks[J]. Ph.d.thesis Carnegie Mellon University, 1992.
[21] Iii L B. Advantage updating[J]. Advantage Updating, 1993.

致　谢

值此论文完成之际，首先向我的导师……

致谢正文样式与文章正文相同：宋体、小四；行距：22 磅；间距段前段后均为 0 行。阅后删除此段。