本科生毕业论文（设计）

Undergraduate Graduation Thesis（Design）

题目 Title	控制自动驾驶汽车 驶过交叉路口方法的对比研究
院系 School (Department)	数据科学与计算机学院
专业 Major	计算机科学与技术
学生姓名 Student Name	王凯祺
学号 Student No.	16337233
指导教师（职称） Supervisor（Title）	张子臻 副教授

时间：2020 年 5 月 9 日

Date: May 9, 2020

说　明

1. 毕业论文（设计）的写作格式要求请参照《中山大学本科生毕业论文的有关规定》和《中山大学本科生毕业论文（设计）写作与印制规范》。

2. 除完成毕业论文（设计）外，还须填写三份表格：

（1）表一　毕业论文（设计）开题报告；

（2）表二　毕业论文（设计）过程检查情况记录表；

（3）表三　毕业论文（设计）答辩情况登记表。

3. 上述表格均可从教务部主页的“下载中心”处下载，如表格篇幅不够，可另附纸。每份毕业论文（设计）定稿装订时应随同附上这三份表格。

4. 封三是毕业论文（设计）成绩评定的主要依据，请认真填写。

Instruction

1. Please refer to ‘*The Guidelines to Undergraduate Graduation Thesis (Design) at Sun Yat-sen University*’and ‘*The Writing and Printing Format of Undergraduate Graduation Thesis(Design) at Sun Yat-sen University* ’ for anything about the thesis format.

2. Three forms should be filled up before the submission of the thesis (design):

(1) Form 1: Research Proposal of Graduation Thesis.

(2) Form 2: Process Check-up Form.

(3) Form 3: Thesis Defense Performance Form.

3. All the above forms could be downloaded on the website of the Office of Education Administration. If there is not enough space in the form, please add extra sheets. Each thesis (design) should be submitted together with the three forms.

4. The form on the inside back cover is the grading sheet. Please fill it up before submission.

表一　毕业论文（设计）开题报告

Form 1: Research Proposal of Graduation Thesis (Design)

论文 (设计) 题目:
Thesis (Design) Title: 控制自动驾驶汽车驶过交叉路口方法的对比研究

(简述选题的目的、思路、方法、相关支持条件及进度安排等)
(Please briefly state the research objective, research methodology, research procedure and research schedule in this part.)

选题目的:

自动驾驶技术正处于高速发展阶段，并将在未来给人类带来出行上的便利。自动驾驶是未来的趋势，我想在毕业课题上对它有一些新的研究，为自动驾驶技术献上自己的一份力。

思路:

我计划使用 Eclipse SUMO 软件对道路交通进行仿真，并用 TraCI API 对仿真中的自动驾驶车辆进行控制，后续用 Flow 框架进行深度强化学习。

方法:

我先使用目前流行的 Time-to-collision(TTC) 算法来控制车辆通过路口，并以此作为标准。之后我会尝试使用深度强化学习的方法来实现控制，与前款标准作对比分析。

相关支持条件:

1) 道路交通仿真软件：Eclipse SUMO；
2) 车辆控制接口：TraCI API；
3) 强化学习框架：Flow （底层为 Ray + TensorFlow）。

进度安排:

1) 2019 年 12 月 6 日前熟练掌握 Eclipse SUMO 软件和 TraCI API 的使用方法；
2) 2019 年 12 月 20 日前完成 TTC 算法实验；
3) 2020 年 1 月 17 日前同步进行 DQN 算法实验及毕业论文摘要、引言、方法、TTC 算法实验部分的撰写；
4) 2020 年 2 月 7 日前完成毕业论文初稿。

Student Signature:　　Date:

指导教师意见:

Comments from Supervisor:

1. 同意开题　　2. 修改后开题　　3. 重新开题

1. Approved(　)　2. Approved after Revision(　)　3. Disapproved(　)

Supervisor Signature:　　Date:

表二　毕业论文（设计）过程检查记录表

Form 2: Process Check-up Form

指导教师分阶段检查论文的进展情况（要求过程检查记录不少于 3 次）
The supervisor should check up the working process for the thesis（design）and fill up the following check-up log. At least three times of the check-up should be done and kept on the log.

第一次检查（First Check-up）：

学生总结
Student Self-Summary:

在这一阶段，毕业项目开始前的准备工作基本完成，主要在如下几个方面：

1) 确定了做“控制自动驾驶汽车通过无信号灯路口”的课题；
2) 查找了与自动驾驶仿真、强化学习的有关资料，为课题的实现打下基础；
3) 熟悉使用 SUMO 自动驾驶仿真平台、控制接口 TraCI API 。

指导教师意见
Comments of Supervisor:

第二次检查（Second Check-up）：

学生总结
Student Self-Summary:

在这一阶段，毕业项目的实验已经基本完成，主要在如下方面：

1) TTC 基准方法的论文已经找到，并已复现完毕；
2) DQN 方法的实验已经完成，并按照计划测量出 4 个指标。

指导教师意见
Comments of Supervisor:

第三次检查（Third Check-up）：

学生总结

Student Self-Summary:

在这一阶段，毕业项目的初稿已经基本完成，主要在如下方面：

1) 查阅了 MDP、Q-Learning 和 DQN 的论文，完成了论文的研究方法部分；
2) 根据源代码，完成了论文的实验设置部分；
3) 根据实验结果，完成了论文的结果部分。

指导教师意见

Comments of Supervisor:

学生签名（Student Signature）：　　日期（Date）：

指导教师签名（Supervisor Signature）：　　日期（Date）：

总体完成情况 **(Overall Assessment)**	指导教师意见 Comments from Supervisor: 本论文通过开源平台 SUMO 研究自动驾驶汽车的控制策略，以模拟汽车通过交叉路口。论文采用深度强化学习框架，针对问题环境设计了 DQN 和 PPO 算法，并与经典的 TTC 与 PRM 算法进行了对比，实验效果良好。论文结构合理，描述清晰，提出的算法具有一定新颖性，实验能可视化。论文达到本科生毕业论文要求。 1、按计划完成，完成情况优（Excellent）：（　） 2、按计划完成，完成情况良（Good）：（　） 3、基本按计划完成，完成情况合格（Fair）：（　） 4、完成情况不合格（Poor）：（　） 指导教师签名（Supervisor Signature）： 日期（Date）：

表三　毕业论文（设计）答辩情况登记表

Form 3: Thesis Defense Performance Form

答辩人 **Student Name**	王凯祺 Wang Kaiqi	**专业** Major	计算机科学与技术 Computer Science
论文 (设计) 题目 **Thesis（Design）Title**	控制自动驾驶汽车驶过交叉路口方法的对比研究 Comparative Study of Controlling Autonomous Vehicles through Intersections		
答辩小组成员 **Committee Members**			

答辩记录

Records of Defense Performance:

记录人签名（Clerk Signature）：　　日期（Date）：

学术诚信声明

本人所呈交的毕业论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料均真实可靠。除文中已经注明引用的内容外，本论文不包含任何其他人或集体已经发表或撰写过的作品或成果。对本论文的研究作出重要贡献的个人和集体，均已在文中以明确的方式标明。本毕业论文的知识产权归属于培养单位。本人完全意识到本声明的法律结果由本人承担。

本人签名：　　日期：

Statement of Academic Integrity

I hereby acknowledge that the thesis submitted is a product of my own independent research under the supervision of my supervisor, and that all the data, statistics, pictures and materials are reliable and trustworthy, and that all the previous research and sources are appropriately marked in the thesis, and that the intellectual property of the thesis belongs to the school. I am fully aware of the legal effect of this statement.

Student Signature:　　Date:

摘 要

在自动驾驶领域，控制汽车通过路口是一件非常有挑战的事情，它需要兼顾安全和效率。我们的目标是让自动驾驶汽车不与其他汽车碰撞，同时让汽车通过路口的时间尽量短。在本文中，我们用传统方法和强化学习的方法来控制自动驾驶汽车通过无红绿灯路口，测算了它们的成功率、碰撞率、平均通过时间、车流车辆平均制动时间，并将它们做了比较。实验结果表明，传统方法的安全性远远超过强化学习方法，可保证不碰撞，但行驶方式过于保守；而强化学习方法能帮助我们在低碰撞率的条件下更快速地通过路口，大幅缩短了平均通行时间。我们目前实现的传统算法及强化学习方法虽然都并不够完美，但它们提供的解决方案为我们分别指出了这两种方法的优缺点，也为未来的研究指明了方向。

关键词： 自动驾驶；强化学习；安全；导航

ABSTRACT

In the field of autonomous driving, controlling vehicles through intersections is an extremely challenging task. It needs to balance safety and efficiency. Our goal is to keep autonomous vehicles from colliding with other cars while letting vehicles pass through the intersections as fast as possible. This paper uses traditional methods and deep reinforcement learning methods to control autonomous vehicles to pass through traffic-free intersections, measures their success rate, collision rate, average transit time, and average brake time. We compare the metrics above, and the experimental results show that the safety of the traditional method far exceeds the reinforcement learning method, which can prevent autonomous vehicles from colliding with others, but its driving method is too conservative. The deep reinforcement learning method can help us dramatically reduce the average transit time with a low collision rate. Although the traditional method and the reinforcement learning method we currently implement are not perfect enough, the solutions provided by the methods above point out some advantages and disadvantages of the two methods, respectively, and also illustrate the direction for future research.

Keywords: Autonomous Driving; Deep Reinforcement Learning; Safety; Navigation

目录

插图目录

表格目录

第一章 引言

自动驾驶是人工智能新时代的一个重要课题。目前，自动驾驶汽车已经开始走出实验室，与普通汽车一起行驶在马路上。它们的出现减少了交通事故及交通拥堵的发生，并改善了我们在拥挤不堪的城市中的流动性。根据预测，到 2035 年，大多数汽车都将具备完全自动驾驶功能[1]。

在驾驶的过程中，在交叉路口发生碰撞的概率远比在道路中间发生碰撞的概率高。比起有信号灯的交叉路口，无信号灯的交叉路口面临更多的问题，比如需要判断对方司机的行为意图、需要尽可能少地妨碍其他车辆等。因此如何能让自动驾驶汽车安全地、无碰撞地通过无信号灯交叉路口，还能同时保证通行效率，是本文要讨论的问题。

目前已经有一些团队在通过交叉路口的问题上设计了通行策略。这些方法中有传统的方法，例如基于碰撞时间的方法 (Time-to-collision，简称 TTC)[2] 以及基于概率风险指标的算法（Probabilistic Risk Metrics，简称 PRM)[3]。TTC 算法根据两车的距离与速度之差的比值大小来决定通行还是停车，而 PRM 算法则是根据当前对当前路口情况的风险评估来决定是否通过。还有一些团队使用了迁移学习的方法[4]，他们的策略是从人类驾驶员的行为中学得的。但是这种方法的缺陷是，如果自动驾驶汽车发现自己处在一个从未遇到过的情况，即不在训练数据中，就有可能无法正确处理它们。除此之外，也有一些团队研究了基于强化学习的方法[5]，即是在各种选择（加速、减速）中随机地尝试，并将成功或失败的方法存储起来，用于更新策略。

本文做出的贡献是重新实现了传统方法、研究了不同的强化学习方法，并对这些方法进行了对比分析。其中的强化学习方法我们采用了深度 Q 网络（Deep Q Network，简称 DQN)[6] 和邻近策略优化（Proximal Policy Optimization，简称 PPO)[7]。结果显示，车流量小的情况下，传统方法能做到 100 % 无碰撞通过路口；在车流量大且容许少量碰撞的情况下，强化学习方法比传统方法用时更少。

本文剩余的章节和内容安排如下：第二章重点介绍了本文所使用的方法，包括传统方法 TTC 和 PRM 算法，以及强化学习 DQN 和 PPO 算法，并详细介绍状态表示、动作集合等强化学习模型细节；第三章介绍了实验内容以及实验环境，并引入评估标准对上述 4 种算法进行考量；第四章深入分析了实验结果以及各个算法的特点；第五章对全文进行总结，对传统算法及强化学习算法的优势对比作了深入的分析，并展望未来可能的发展方向。

第二章 研究方法

自动驾驶汽车通过交叉路口的问题，可以用传统的 TTC 和 PRM 算法来解决，还可以建模成强化学习任务，进而用 DQN 和 PPO 的算法来解决。本章重点介绍了本文使用的研究方法，解释各算法的基本概念，并阐述建模的过程。

2.1 TTC 算法

TTC 最早是由 Hayward J.C. 提出的[8]。TTC 方法反映了刹车过程中人们先是感觉到危险，后危险消退的主观感受。其将 TTC 方法定义为“两辆车以当前的速度和路径继续行驶，发生碰撞所需的时间”。它是随时间变化的度量值，也就是说，我们可以在任意一小段时间片段中计算出 TTC 的值。该方法后来用于确定驾驶员辅助系统激活紧急刹车的标准，以减少高速公路上的追尾事故数量[2]。TTC 作为一个指标，可以系统性地观察碰撞之前的过程，这有助于我们分析、诊断和解决交通安全问题。在交通事故的技术研究中，TTC 已被证明是评估碰撞严重性的有效方法。

在本问题中，我们引用了 Bouton M. *et al.* 在交叉路口中 TTC 的定义[9]：考虑一条由本车中心为起点、往前进方向的一条射线，本车与另一辆车 i 的 TTC 为车辆 i 以当前速度沿当前方向与射线交叉所需要的时间。例如，在图2-1中，本车与车辆 i 的 TTC 为 d_i 与它速度 v_i 的比值。

我们计算本车与其它所有车辆的 TTC，并取它们的最小值，记为 TTC_{safe}。如果在仿真时连续两帧中的 TTC_{safe} 值均大于 TTC 阈值，则令本车启动。一旦决定启动，本车将由 IDM 控制器接管，驶入路口并控制本车通过路口。

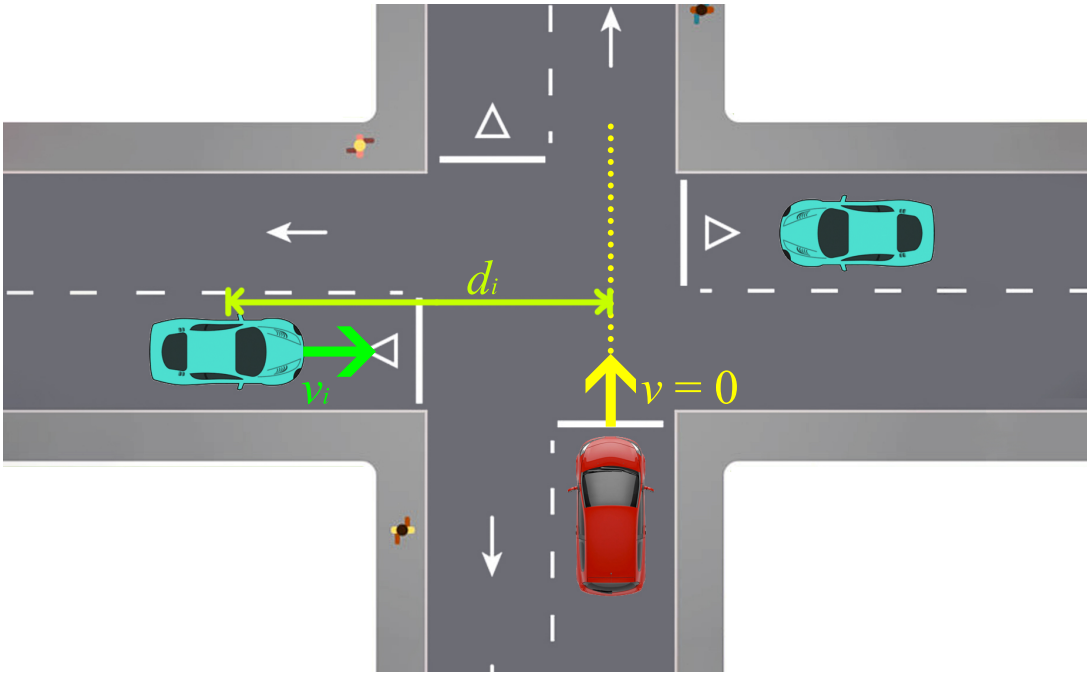

图 2-1 TTC 计算方法示意图

2.2 PRM 算法

人类驾驶员经常由于疏忽大意或者错误判断而发生碰撞。对于自动驾驶汽车来说也是一样，自动驾驶汽车很难通过十字路口或者与人类驾驶的汽车进行互动。PRM 算法是 McGill S.G. *et al.* 在 2019 年提出的新方法，他们为了解决这一挑战，对交叉路口发生碰撞的风险进行建模，并部署完整和共享的自动驾驶系统来提高车辆的安全性。他们提出了一个风险模型，该模型考虑了交叉路口的车流、遮挡、传感器的误差以及驾驶员注意力的集中程度，来估计本车通过路口的风险。

在我们的实验环境中，所有车辆都是无遮挡的，可以互相看见对方；传感器也是无误差的。因此我们的建模会比上面的建模方法简单很多。下面介绍我们的建模：令 t_c 为本车从现在开始以当前速度 v_e 完全通过交叉路口所需要的时间。考虑一辆速度为 v_i 、距离交叉路口 d_i 的车流车辆，我们定义这辆车的风险为

$$\begin{cases} 0 & d_i > v_i \cdot t_c \\ 1 & d_i < d_s \\ \exp(-\lambda_a(d_i - d_s)) & \text{其它} \end{cases}, \tag{2-1}$$

其中 $\lambda_a > 0$ 是模拟车流车辆注意力的参数，λ_a 越大表示注意力越高，看得更远；d_s 是车流车辆的刹车距离。若这辆车需要超过 t_c 的时间来通过路口，则风险为 0；相反，如果车流车辆离路口的距离小于刹车距离，则风险为 1；其它情况则根据车流车辆离路口

的距离与刹车距离的差指数衰减。

在计算出每辆车流车辆的风险后，取这些风险的最大值 r 作为整个路口的风险。在进入路口之前，我们通过改变本车的速度

$$v'_e = \begin{cases} \frac{d_e}{d_{\text{nudge}}} v_e & (r > r_{go}) \text{且} (d_e < d_{\text{nudge}}) \\ v_e & \text{其它} \end{cases} \tag{2-2}$$

来控制车辆，其中 r_{go} 表示风险阈值，d_e 表示本车离路口的距离，d_{nudge} 表示本车的起点离路口的距离。当路口风险 r 超出阈值 r_{go} 时，我们控制速度让本车慢慢停车；其它情况维持原速度。进入路口之后，则尽快加速到最大速度。

2.3 强化学习

强化学习是机器学习中的一个分支。它能采取适当的行动，使得在特定的环境下最大化收益。强化学习与有监督学习的不同之处在于，在有监督学习中，训练数据有标记正确答案，因此模型可以使用正确答案来进行训练；而强化学习中没有答案，由强化智能体自主决定怎么样完成给定的任务。在没有训练数据的情况下，它必然会从经验学习。目前，强化学习方法已经被各种软件所使用，例如在计算机集群中进行资源分配[10]，研究人员使用了强化学习算法自动学习为挂起的进程分配与调度计算机资源，目的是最大限度地降低运行速度的损失；再如在机器人技术应用中，研究人员让机器人通过训练学习策略，将视频图像映射到机器人的动作[11]，将输入为 RGB 的图像通过卷积神经网络输出到电机扭矩。强化学习组件通过生成来自自身状态分布的训练数据，来指导机器探索策略。

强化学习任务[12][13]通常用马尔可夫决策过程（Markov Decision Process，简称 MDP）来表示：环境 E 的状态空间为 X，其中每个状态 $x \in X$ 为环境的描述；机器可采取的动作集合为 A；转移函数 $P : X \times A \times X \mapsto \mathbb{R}$ 表示从原状态 x 通过采取动作 a，能转移到新状态 x' 的概率；奖赏函数 $R : X \times A \times X \mapsto \mathbb{R}$ 表示在状态转移中反馈给机器的奖赏；总的来说，一个强化学习任务就对应了一个四元组 $E = \langle X, A, P, R \rangle$。

机器需要在环境 E 中不断尝试，学到一个策略 π。根据这个策略，在状态 s_t 下给定动作 a_t 就能得知选择这个动作的概率 $\pi(a_t|s_t)$。

我们将使用状态-动作值函数 Q_π、值函数 V_π 和优势函数 A_π 的标准定义：

$$Q_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}, a_{t+1}, \cdots} \left[\sum_{l=0}^{\infty} \gamma^l r(s_{t+l}) \right] \tag{2-3}$$

表示从状态 s_t 出发，执行动作 a_t 后再使用策略 π 带来的期望累计奖赏；

$$V_\pi(s_t) = \mathbb{E}_{a_t, s_{t+1}, \cdots}\left[\sum_{l=0}^{\infty} \gamma^l r(s_{t+l})\right] \tag{2-4}$$

表示从状态 s_t 出发，使用策略 π 带来的期望累计奖赏；

$$A_\pi(s_t, a_t) = Q_\pi(s_t, a_t) - V_\pi(s_t) \tag{2-5}$$

表示从状态 s_t 出发，执行动作 a_t 后使用策略 π 带来的期望累计奖赏和直接使用策略 π 带来的期望累计奖赏的差；其中 $a_t \sim \pi(a_t|s_t), s_{t+1} \sim P(s_{t+1}|s_t, a_t)$ 。

在接下来的 2.4、2.5、2.6和2.7 节中，我们会定义本任务的状态空间 X 、动作集合 A 、转移函数 P 和奖赏函数 R 。

2.4 状态的表示

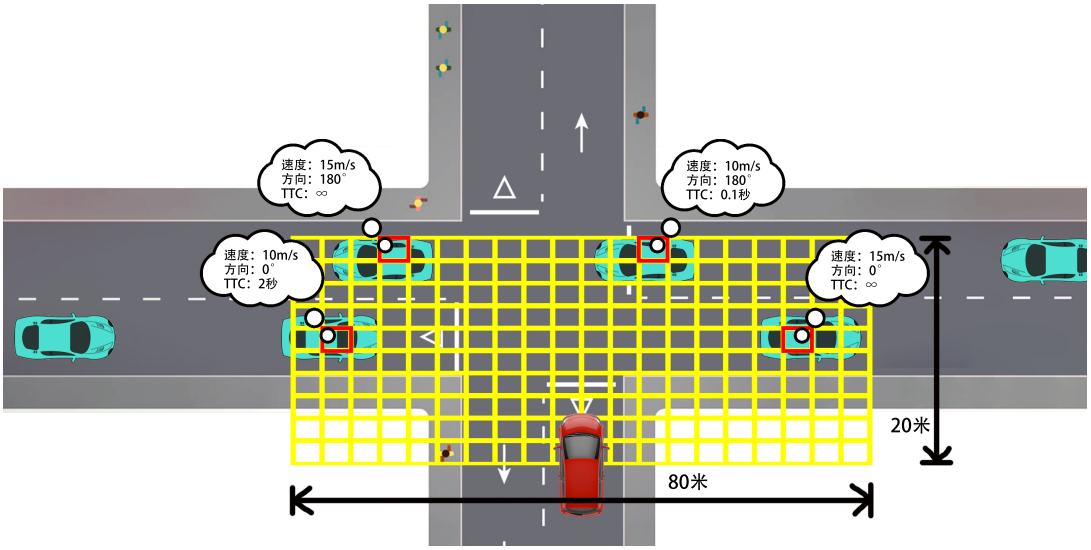

图 2-2 状态表示示意图

我们将以汽车为中心，左右各 40 米、前方 20 米的矩形区域（如图2-2）划分为 20×10 的网格。当某辆车的中心落在网格内时，则用一个五维向量 $\mathbb{R}^5$ 表示：这辆车的速度、速度在水平方向上的分量、速度在垂直方向上的分量、方向和与自动驾驶汽车的 TTC；缺省情况下，速度、方向和 TTC 分别记为 $0, 0, +\infty$；这样，我们就构造了一个 $\mathbb{R}^{20\times10\times5}$ 的向量。向量中的每个元素均需要被标准化到 $[0, 1]$ 区间内的实数。

2.5 动作集合

本实验的动作集合为 $\{-4\,\mathrm{m\,s^{-2}}, -2\,\mathrm{m\,s^{-2}}, 0\,\mathrm{m\,s^{-2}}, 2\,\mathrm{m\,s^{-2}}\}$，其中 $-4\,\mathrm{m\,s^{-2}}$ 代表急刹车，$-2\,\mathrm{m\,s^{-2}}$ 代表普通刹车，$0\,\mathrm{m\,s^{-2}}$ 代表维持原速度，$2\,\mathrm{m\,s^{-2}}$ 代表加速。

2.6 转移函数

强化学习控制器会为自动驾驶车辆在动作集合中选择一个加速度，车流的车辆由它自身的控制器选择加速度。设某车原坐标为 (x, y)、速度大小为 v、方向为 θ，选择的加速度为 a。

设每一步时间为 t，则它下一时刻的坐标、速度大小为：

$$\begin{aligned} x' &= x + vt\cos\theta + \frac{1}{2}at^2\cos\theta \\ y' &= y + vt\sin\theta + \frac{1}{2}at^2\sin\theta \\ v' &= v + at \end{aligned} \tag{2-6}$$

给定原状态 x 和动作 a，即可用新的坐标根据 2.4 节的方法重新计算新的状态 x'。

2.7 奖赏函数

在强化学习中，奖赏函数的设置尤为重要，它直接决定了学习目标。如果自动驾驶汽车能顺利通过路口，我们就给它一个正的奖赏（奖赏值 2000）；如果自动驾驶汽车与其它车辆发生碰撞，我们会给它一个很重的惩罚（奖赏值 $-20\,000$）。另外，我们鼓励自动驾驶汽车尽可能快地通过路口，而不希望它一直停在路口前，因此在正常行驶每一步的过程中都会规定一个步数代价，如车速不低于 $1\,\mathrm{m\,s^{-1}}$，记该步的奖赏值为 -1；否则令 $step$ 为车速低于 $1\,\mathrm{m\,s^{-1}}$ 的连续步数，记该步的奖赏值为 -1.005^{step}。

2.8 DQN 算法

DQN 算法是基于 Q 学习[14] 的改进算法。Q 学习是一种无模型的强化学习，它为智能体提供了一种能力，让智能体能体验到它的行为带来的后果，并从中学习到最合适的行为。

在 Q 学习中，给定最优的状态-动作值函数 $Q^*(x, a)$ 的情况下，我们可以找到一种最优策略 $\pi^*(x) = \arg\max_{a\in A} Q^*(x, a)$。状态-动作值函数 $Q(x, a)$ 根据著名的贝尔曼等式[15]

（Bellman Equation）

$$Q_\pi(x,a)=\sum_{x'}P(x,a,x')\left[R(x,a,x')+\gamma\sum_{a'}\pi(a'|x')Q_\pi(s',a')\right]$$

进行迭代更新，最后输出策略 π 的过程即为 Q 学习算法。

在本问题中，我们面临的状态空间 $\mathbb{R}^{20\times10\times5}$ 是连续的，有无穷多个状态，这在存储和计算上均无法实现，DQN 的出现为我们解决了这个问题。在 Q 学习的基础上，DQN 引入神经网络来近似 Q 值，即，使用值函数近似的方法，引入参数向量 θ ，将 $Q^*(s,a)$ 用 $Q(s,a;\theta)$ 近似，并最小化损失函数

$$L_i(\theta_i)=\mathbb{E}_{(s,a,r,s')\sim U(D)}\left[(r+\gamma\max_{a'}Q(s',a';\theta_i^-)-Q(s,a;\theta_i))^2\right] \tag{2-7}$$

从而达到支持高维状态输入的能力。其中，γ 为 Q 学习中决定机器更新视野的折扣因子，θ_i 为 Q 网络在第 i 次迭代时的参数，θ_i^- 为 Q 网络在第 i 次迭代时计算 Q 值增量所使用的参数。参数 θ_i^- 只会随着参数 θ_i 每 C 步刷新一次。

本文中的 DQN 算法在实现上使用了 RLlib[16] 的实现，其神经网络的构建方法如图2-3、图2-4及图2-5所示。图2-3中的神经网络用于将2.4节的 1000 维状态空间标准化为 256 维状态空间；图2-4的神经网络用于将标准化后的 256 维状态空间 s 映射为 $Q^*(s,a)$ ；图2-5的神经网络则用于将标准化后的 256 维状态空间 s 映射为 $V^*(s)$。

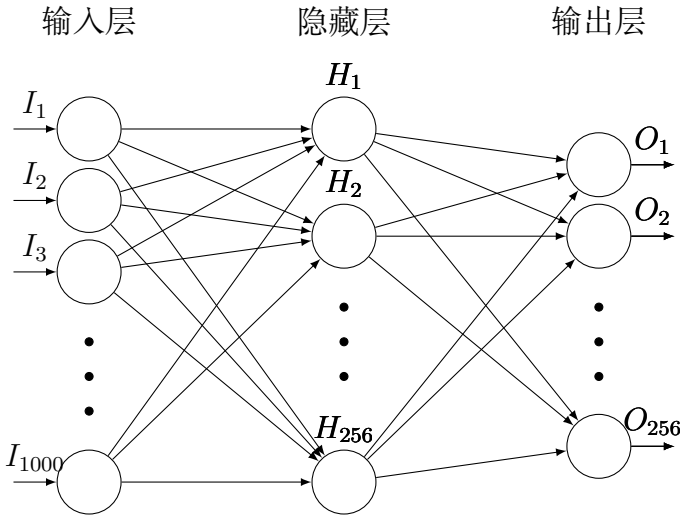

图 2-3 用于标准化状态空间的神经网络

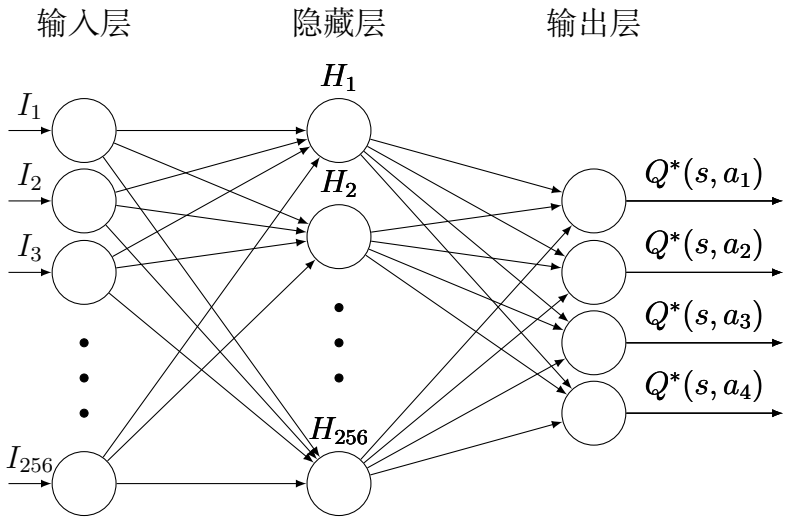

图 2-4 将标准化状态空间映射为 Q 值的神经网络

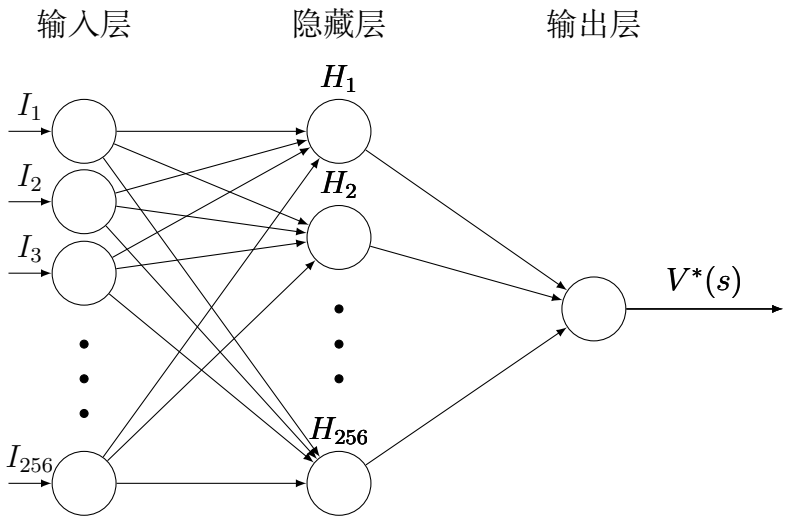

图 2-5 将标准化状态空间映射为 V 值的神经网络

2.9 PPO 算法

PPO 算法是强化学习中的梯度下降算法，通过与环境交互采样数据以及使用随机梯度上升方法优化“替代的”目标函数，二者交替运行，最终学到最佳策略。

PPO 算法是由受信域策略优化方法（Trust Region Policy Optimization，简称 TRPO[17]）改进而来。TRPO 定义了新的目标函数，并使用了约束和优势函数来完成策略的更新。

形式化地，参数更新的描述如下：

$$\begin{aligned} \text{maximize}_{\theta} \quad & \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right] \\ \text{subject to} \quad & \hat{\mathbb{E}}_t \left[\text{KL}[\pi_{\theta_{\text{old}}}(\cdot|s_t), \pi_\theta(\cdot|s_t)] \right] \leq \delta \end{aligned} \tag{2-8}$$

其中 $\hat{A}_t$ 表示时刻 t 的优势函数的估计值、$\hat{\mathbb{E}}_t$ 表示采样估计的期望、θ_{old} 表示更新前的策略参数、KL 为相对熵（Kullback-Leibler Divergence）。

PPO 在 TRPO 的基础上，增加了截断，从而使更新幅度限制在一定范围内。记 $r_t(\theta)$ 为概率比例 $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ ，那么有 $r(\theta_{old}) = 1$ 。我们可以把 TRPO 的目标函数替换为

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right] = \hat{\mathbb{E}}_t \left[r_t(\theta) \hat{A}_t \right] \tag{2-9}$$

上标 CPI 表示保守的策略迭代（Conservative Policy Iteration，简称 CPI）[18]。如果没有约束，最大化 L_{CPI} 将导致大幅度策略更新。因此，PPO 加入了截断机制：

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right] \tag{2-10}$$

在第二项中，$\text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)$ 通过截断概率比例来修改目标函数，使 r_t 不能超出区间 $[1-\epsilon, 1+\epsilon]$ 。

本文中的 PPO 算法在实现上同样使用了 RLlib 的实现，其神经网络的构建方法如图2-6所示。输入层为2.4节的 1000 维状态空间，分别经过两层全连接的 256 维隐藏层后，映射到输出层得到 $Q^*(s,a)$ 和 $V^*(s)$ 。

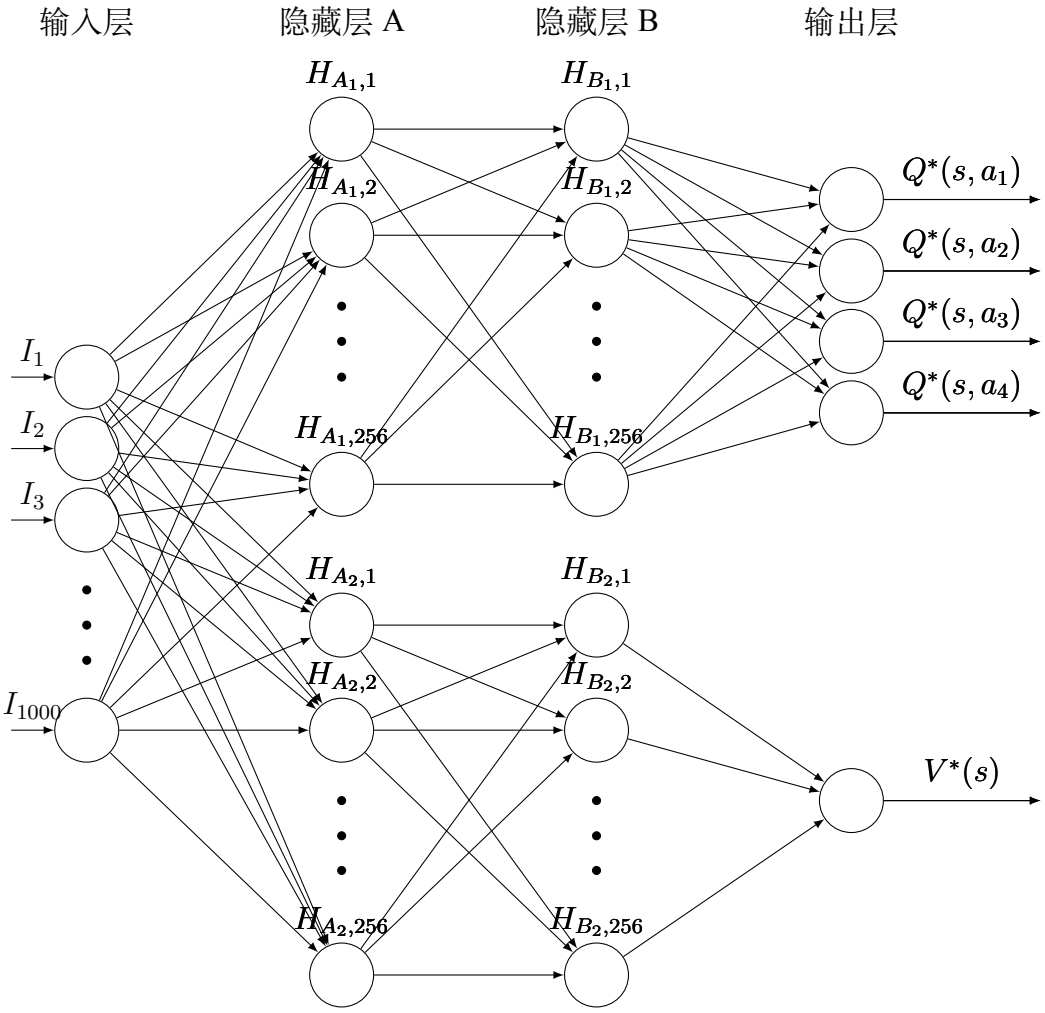

图 2-6 PPO 神经网络模型

第三章 实验

本章首先介绍了实验内容，介绍了本实验所使用的软件及各组件之间的关系，并且引入了 4 个安全性和效率性的评估标准来对这些算法进行综合考量，最后列出了本实验各算法的详细参数。

3.1 实验内容

本文用传统方法和强化学习的方法，通过控制自动驾驶汽车的加速度，让它具备通过路口的能力，并测量相关指标。我们使用传统的 TTC 、PRM 算法以及强化学习的 DQN 、PPO 算法分别进行实验。

通过控制变量的方法，我们既要对比不同车流量情况下各种算法的性能，也要对比不同行驶路线下各种算法的能力。每种算法我们分别进行了两个子实验：

1) 直行通过不同车流量的路口；
2) 在车流量一定的条件下，以左转弯、直行、右转弯的行驶路线通过路口，如图3-1所示。

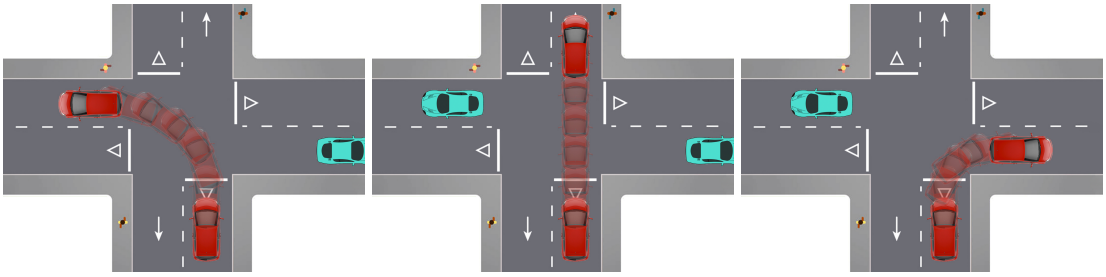

图 3-1 左转弯、直行、右转弯通过路口示意图

3.2 实验环境

在本实验中，我们使用了开放源代码、高度可移植的道路交通模拟软件 SUMO[19] 来对道路、车辆、路线进行建模和仿真。SUMO 提供了灵活的应用程序接口 TraCI 来获取道路、车辆的状态，用户可以控制车辆、道路、车流量和车辆的出发时间。在仿真结束后，SUMO 还会帮助我们记录车辆轨迹、交通流量数据、交通灯、尾气排放、能源消耗等量化指标，输出到文件中，并提供应用程序将输出转化为其它格式。

我们还使用了开源的 Python 库 Flow[20] 。Flow 是一款用于将 SUMO 应用程序和强

化学习算法库 RLlib 集成在一起的计算框架，帮助我们将深度强化学习算法应用于交通场景，并在千变万化的路况中对车辆进行控制。它能让我们更轻松地使用策略优化算法来训练控制器，以满足我们对高度自定义的交通指标的优化需求。

图3-2描述了各组件之间的数据流动关系。Flow 负责启动和停止 SUMO 模拟器、构建 OpenAI Gym[21] 环境。OpenAI Gym 环境向 SUMO 模拟器发送自动驾驶汽车的动作（加速/减速），接收车辆、道路的状态。环境与强化学习模型的交互为：强化学习模型从环境中观测到观测值 x，根据策略作出动作 a 发回环境，再从环境中获取奖赏值，不断重复此过程。

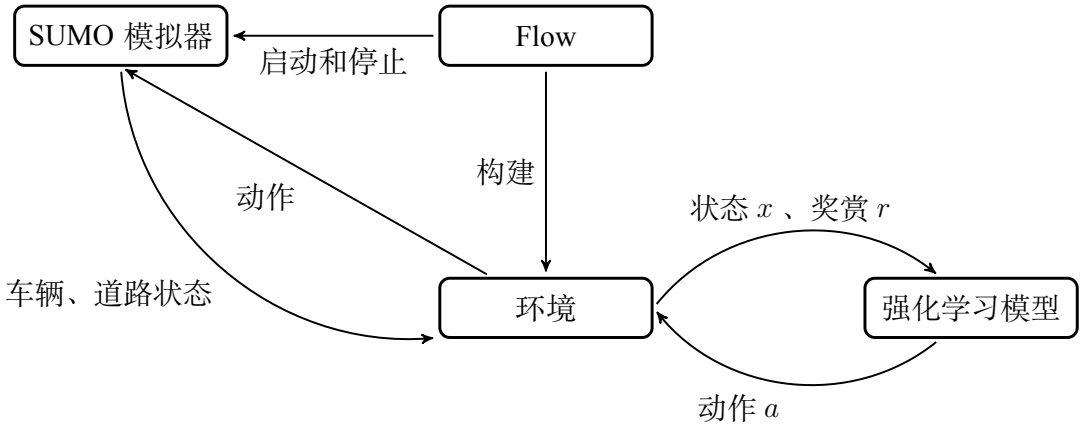

图 3-2 各组件之间的数据流动关系

3.3 评估标准

为了在安全性和效率性方面对该算法进行综合性地考量，我们特别设定了以下 4 个评估标准。成功率和碰撞率是用于衡量算法安全性的指标，平均通过时间是用于衡量算法效率性能的指标，而车流车辆平均制动时间则是衡量自动驾驶汽车的通行对其它车辆影响的指标。

1) **成功率**：小汽车在规定时间内无碰撞地通过路口并到达目标车道的试验次数与总试验次数的比值。注意，碰撞和超时未完成均不计为成功。
2) **碰撞率**：小汽车在通过路口时与车流车辆发生碰撞的试验次数与总试验次数的比值。
3) **平均通过时间**：小汽车通过路口的时间，并在多次成功的试验中取平均值。
4) **车流车辆平均制动时间**：统计车流车辆发生 $4\,\mathrm{m\,s^{-2}}$ 以上（含）制动的总时间（若同一时刻有多辆车制动，计多次）与车流车辆数量的比值，并在多次成功的试验中取平均值。

3.4 参数设置

3.4.1 SUMO 模拟器参数

我们的实验设置在一个十字路口中，参数如表3-1所示。模拟器按预先定义的时间间隔（即每一步的长度）运行，步长为 0.1 秒，即每 0.1 秒刷新一次仿真画面。由于每次仿真开始时道路上都没有车，我们特别设置了 15 秒的准备时间，让车流充满整条道路。准备完成后，自动驾驶车辆才会出现在停止线前，计时开始。为了防止自动驾驶车辆在停止线前无限期停留，我们限制最大仿真时长为 60 秒即 600 步（不含准备时间）。

每个车道最高限速 $20\,\mathrm{m\,s^{-1}}$ 。车流量为每秒钟驶入道路的车辆数的数学期望，例如车流量为 0.2 辆/s 意味着平均每 10 s 有一辆车驶入由西向东的道路，每 10 s 又有一辆车驶入由东向西的道路。

在我们的实验中，不仅有我们的自动驾驶车辆，还有一些别的车流也会通过路口。我们在车流的车辆中应用了智能驾驶员模型（Intelligent Driver Model，简称 IDM）[22]。它是根据前方车辆控制加减速的跟车模型，能准确地表示真实世界中的驾驶行为。

我们还通过 SUMO 软件内置的 Krauss 随机驾驶模型[23]，指定驾驶员的熟练度，从而改变车流车辆的速度分布，使得熟练度高的开得快一些、熟练度低的开得慢一些，模拟了随机性。

表 3-1 SUMO 参数表

	参数名	参数值
SUMO 模拟器参数	仿真步长	0.1 s
	准备时间	15.0 s
	最大仿真时长	60.0 s
道路参数	车道数	双向两车道
	车道宽	3.2 m
	全路段限速	$20\,\mathrm{m\,s^{-1}}$
车流参数	车长	5.0 m
	车宽	1.8 m
	最大速度	$30\,\mathrm{m\,s^{-1}}$
	最大加速度	$2.6\,\mathrm{m\,s^{-2}}$
	最大减速度	$4.5\,\mathrm{m\,s^{-2}}$
	行驶方向	由西向东、由东向西
	行驶路线	直行
	车流量	0.1 辆/s 至 0.8 辆/s
	控制器	智能驾驶员模型

3.4.2 TTC 阈值的选定

由于 TTC 阈值是超参数，在实验之前先确定此参数。我们在不同的 TTC 阈值下对 TTC 算法进行了评估。为了确保在车流量大的场景下 TTC 算法也能保证无碰撞，在试验中，车流量设定为 0.6 辆/s ，行驶路线在左转弯、直行、右转弯中等概率随机。我们先行对阈值为 $0.0, 0.1, \cdots, 4.9$ 分别进行了 1000 次试验。

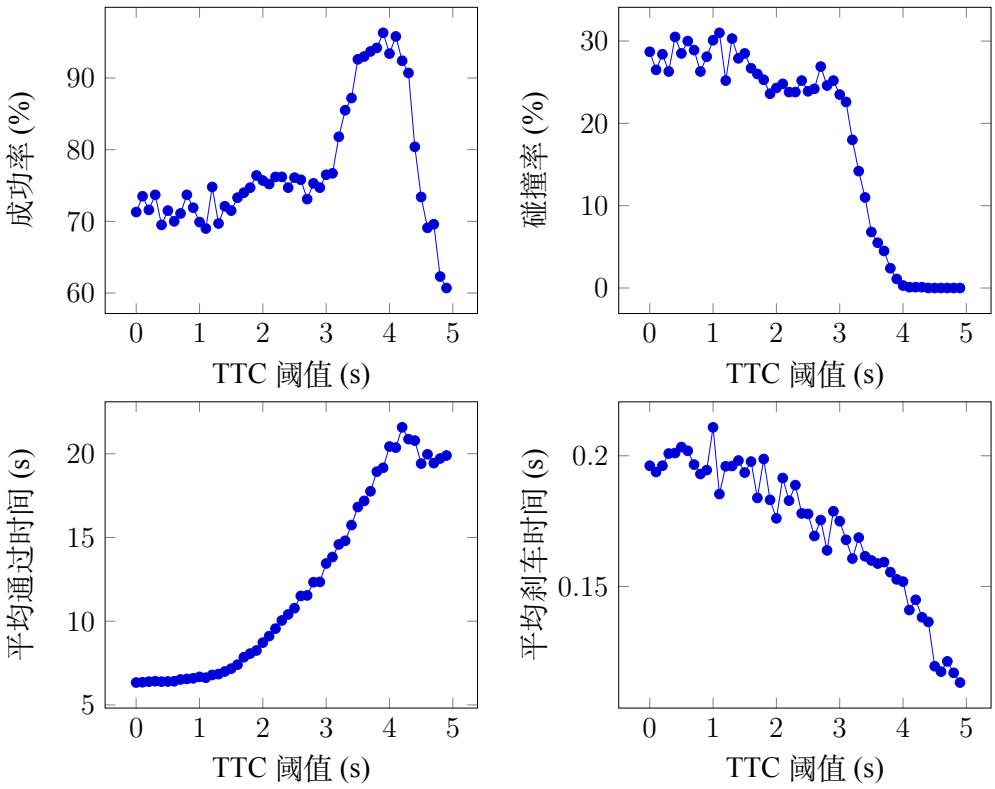

图 3-3 不同 TTC 阈值下的指标

图3-3为不同 TTC 阈值下的试验结果。随着 TTC 阈值的增加，碰撞率从 3 s 开始急剧减少，直到 4.4 s 时减至 0；成功率自 3 s 开始上升，到 4 s 时断崖式下降，其原因是 TTC 阈值越高，车辆在路口等待的时间就会越长，导致最终发生超时未完成的情况。综合各项指标的考虑，我们将 TTC 阈值定为 4.5 。

3.4.3 PRM 参数

PRM 算法的参数如表3-2 所示。r_{go} 根据交叉路口风险观测图[3] 选取，d_s 和 λ_a 则是根据对两变量的多个数值进行排列组合，选择数值尽可能小、碰撞率尽可能低的组合。

表 3-2 PRM 参数表

参数名	参数值	备注
r_{go}	0.1	风险阈值
λ_a	1.0	模拟车流车辆注意力的参数
d_s	55.0 m	车流车辆的刹车距离

3.4.4 训练参数

DQN 和 PPO 算法的训练参数分别如表3-3、3-4所示。由于我们的仿真时长为 600 步，而前面的决策对最终结果有决定性的影响，故折扣因子应取较大的值 $\gamma = 0.999$；学习率则是根据收敛速度和更新幅度综合确定的。

表 3-3 DQN 训练参数表

参数名	参数值	备注
gamma	0.999	折扣因子
learning rate	0.001	学习率
n_step	3	N-step Q 学习
buffer_size	500 000	经验回放存储大小
prioritized_replay_alpha	0.6	优先经验回放 α 参数
prioritized_replay_beta	0.4	优先经验回放 β 参数

表 3-4 PPO 训练参数表

参数名	参数值	备注
gamma	0.999	折扣因子
learning rate	5×10^{-5}	学习率（SGD 步长）
lambda	0.95	广义优势估计器[24] 参数
kl_coeff	0.5	相对熵（KL divergence）系数

第四章 结果

本章我们将展示“固定行驶方向不同车流量”以及“固定车流量不同行驶方向”这两组实验的实验结果，并具体分析。

4.1 车流量变化下的指标

对于每种算法，我们在车流量取值为 0.1 辆/s, 0.2 辆/s, $\cdots$, 0.8 辆/s 的条件下分别进行 10 000 次试验，并测算 3.3 节关于成功率、碰撞率、平均通过时间、车流车辆平均制动时间共计 4 项指标，结果如图4-1所示。

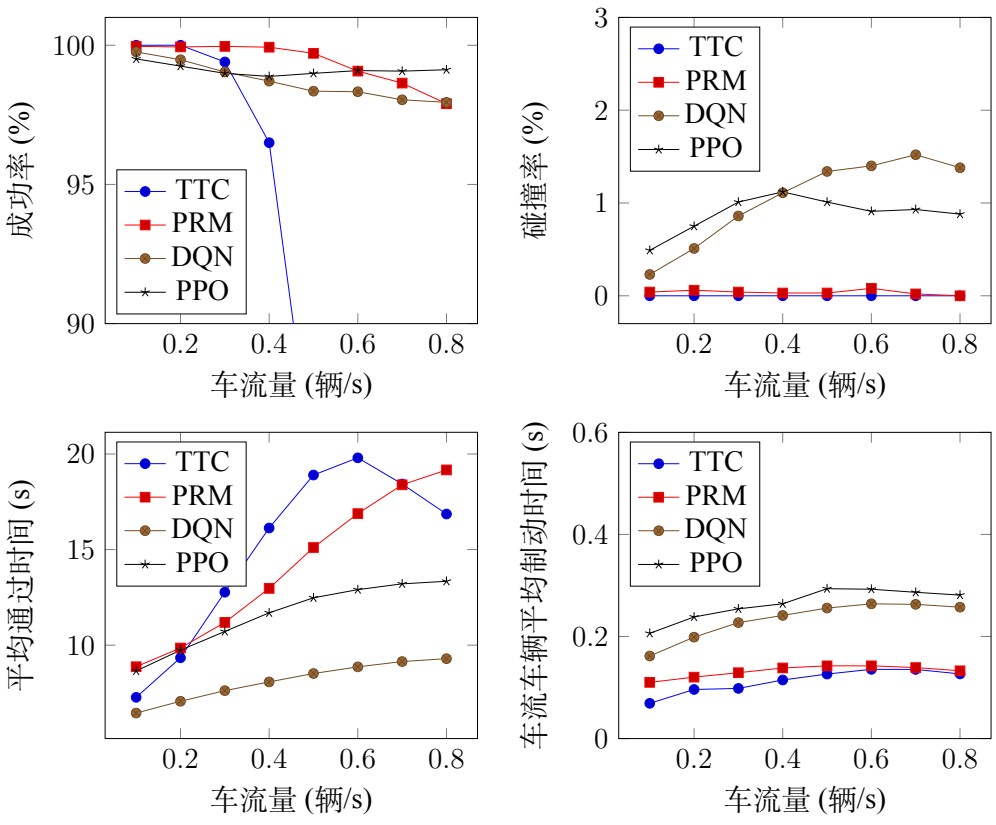

图 4-1 直行场景下，不同车流量的指标

TTC 算法无论在车流量小的情况下，还是在车流量大的情况下，都能保持无碰撞的成绩。在这 4 种算法中它是唯一一个能保持无碰撞成绩的算法。其它车辆的平均刹车时间在非常低的水平，说明 TTC 算法驱动的车辆在通行的过程中，别的车辆基本没有发生制动，这意味着它对交通的影响最小。在图中我们可以看到，TTC 算法在平均通过时间的曲线在最上方，由此可见它的缺点就是通过时间过长，在车流量大的情况下于规

定时间内无法通过路口。注意到 TTC 算法的平均通过时间在车流量大于 0.6 辆/s 时有回落，这是因为 TTC 算法在车流量大的情况下有大量超时未完成的案例，而在平均通过时间的计算中仅对完成的案例进行统计。故 TTC 算法在车流量大于 0.6 辆/s 时，实际的平均通过时间会比图中的高。

PRM 算法在成功率上较 TTC 算法有大幅提升，TTC 算法的成功率在车流量为 0.4 辆/s 以上时大幅下降，以至于在车流量为 0.8 辆/s 时成功率仅为 53.30 %；而 PRM 算法在同等情况下的成功率高达 97.90 %。PRM 算法的碰撞率还不能做到与 TTC 算法同样为 0，但碰撞率最高仅为 0.08 % ①。对于平均通过时间，PRM 算法在整体上超过了 TTC 算法，仅在车流量少的情况下用时稍比 TTC 算法多。其它车辆的平均刹车时间方面，PRM 算法的成绩仅次于 TTC 算法，在车流量小的情况下对车流车辆的影响比 TTC 稍高，随着车流量的增长，二者在对车流车辆的影响几乎一致。

DQN 算法的成功率比 PRM 算法稍低，但远高于 TTC 算法。在平均通过时间方面，DQN 算法的表现远比 TTC 和 PRM 两种传统算法好，无论车流量为多少，它都是最快的，用时最多可以比 TTC 算法和 PRM 算法节省 50 % ②。在用时上取得这么好的成绩是需要付出代价的。代价的其中之一是将产生了近 1.5 % 的碰撞率，大约是 PRM 算法的 20 倍，与完全无碰撞的 TTC 算法更是无法比较；代价的其中之二是它对其它车辆的影响高于 TTC 和 PRM 算法，在 DQN 算法下其它车辆的制动时间大约为 TTC 、PRM 算法的两倍。

PPO 算法的成功率比 DQN 算法略高，甚至在车流量大的情况下比 PRM 算法还高。它的碰撞率和车流车辆平均制动时间与 DQN 算法差不多，均比 PRM 算法高出不少；平均通过时间比 DQN 算法高出大约 3 s。总的来说，PPO 算法与 DQN 算法的性能差距不大，DQN 算法更激进一些，而 PPO 算法更保守。

4.2 行驶路线变化下的指标

对于每种算法，分别取车流量为 0.2 辆/s 和 0.6 辆/s，再分别以左转弯、直行、右转弯的行驶路线行驶，测算 3.3 节关于成功率、碰撞率、平均通过时间、车流车辆平均制动时间共计 4 项指标。

在这三种行驶路线中，右转弯只需要考虑由西往东的车流，应是这三种行驶路线中最简单的；直行不但需要考虑由西往东的车流，还需要考虑由东往西的车流，在这三种行驶路线中难度排行第二；左转弯的难度最大，是因为它不但需要考虑两个方向的车

① 在车流量为 0.6 辆/s 的条件下取得
② 分别在车流量为 0.6 辆/s 及 0.8 辆/s 的条件下取得

流，它在路口中行驶的时间还是最长的，风险也是最大的。

4.2.1 车流量小的场景

我们用 0.2 辆/s 的车流量来代表车流量小的场景，其各项指标的实验结果如表 4-1 所示。

表 4-1 车流量为 0.2 辆/s 的条件下不同行驶路线的指标

行驶路线	指标	TTC	PRM	DQN	PPO
左转弯	成功率 (%)	**100.00**	99.62	98.45	98.21
	碰撞率 (%)	**0.00**	0.38	1.50	1.13
	平均通过时间 (s)	9.57	10.03	**7.46**	10.21
	车流车辆平均制动时间 (s)	**0.0871**	0.1118	0.1530	0.1515
直行	成功率 (%)	**100.00**	99.94	99.48	99.25
	碰撞率 (%)	**0.00**	0.06	0.51	0.75
	平均通过时间 (s)	9.34	9.85	**7.06**	9.74
	车流车辆平均制动时间 (s)	**0.0962**	0.1204	0.1987	0.2382
右转弯	成功率 (%)	**100.00**	**100.00**	99.51	98.56
	碰撞率 (%)	**0.00**	**0.00**	0.31	0.53
	平均通过时间 (s)	7.19	8.65	**6.12**	8.41
	车流车辆平均制动时间 (s)	**0.0480**	0.0596	0.0939	0.1068

TTC 算法无论是在左转弯、直行、右转弯的场景下，都能有 100 % 的成功率、零碰撞率以及最低的车流车辆平均制动时间，平均通过时间比最快的 DQN 算法慢了大约 2 s 。

PRM 算法在成功率和碰撞率上稍稍比 TTC 算法差，有极小概率碰撞，平均通过时间和车流车辆平均制动时间均比 TTC 算法稍长。在车流量小的情况下，PRM 算法相对于 TTC 算法并不占优势。

DQN 算法在直行、右转弯场景下均有大约 99.5 % 的成功率，略低于传统算法；但左转弯场景的性能不佳，仅有 98.45 % 的成功率。DQN 算法在所有场景下均有最快的通行时间，但它的碰撞率、车流车辆平均制动时间也比 TTC 、PRM 算法高。

PPO 算法在直行场景下表现优秀，成功率略低于 DQN 算法；但转弯性能不佳，在左转弯和右转弯场景下，成功率都比较低。PPO 算法的平均通过时间与其它算法相比较高，车流车辆平均制动时间与 DQN 算法接近，高于传统算法。

4.2.2 车流量大的场景

我们用 0.6 辆/s 的车流量来代表车流量大的场景，其各项指标的实验结果如表 4-2 所示。

表 4-2 车流量为 0.6 辆/s 的条件下不同行驶路线的指标

行驶路线	指标	TTC	PRM	DQN	PPO
左转弯	成功率 (%)	69.89	**98.82**	97.58	94.76
	碰撞率 (%)	**0.00**	0.48	1.52	3.15
	平均通过时间 (s)	19.22	16.64	**12.97**	17.13
	车流车辆平均制动时间 (s)	**0.1284**	0.1420	0.1848	0.2314
直行	成功率 (%)	70.80	99.07	98.33	**99.09**
	碰撞率 (%)	**0.00**	0.08	1.40	0.91
	平均通过时间 (s)	19.80	16.88	**8.86**	12.90
	车流车辆平均制动时间 (s)	**0.1357**	0.1425	0.2640	0.2927
右转弯	成功率 (%)	99.89	**99.96**	99.14	98.01
	碰撞率 (%)	**0.00**	0.04	0.27	1.93
	平均通过时间 (s)	12.30	11.17	**8.72**	10.25
	车流车辆平均制动时间 (s)	**0.1008**	0.1097	0.1320	0.1364

TTC 算法无论是在左转弯、直行、右转弯的场景下，都能有零碰撞率以及最低的车流车辆平均制动时间，平均通过时间比最快的 DQN 算法多出约 50％至 100％。TTC 算法在右转弯场景下几乎都能成功通过路口，但在左转弯和直行场景下，只有 70％的成功率。

PRM 算法在成功率较 TTC 算法有大幅提升，在左转弯和右转弯场景中位列第一，平均通过时间也有小幅降低。但在碰撞率方面，PRM 算法稍高，尤其在左转弯场景产生了 0.48％的碰撞率。

DQN 算法在左转弯、直行、右转弯的场景下，都有最快的通行时间。速度快带来的副作用则是成功率的降低以及碰撞率的提升，车流车辆平均制动时间也稍比传统的 TTC 、PRM 算法高。

PPO 算法在直行场景下表现优异，成功率最高，碰撞率比 DQN 算法低，平均通过时间介于 DQN 算法和 PRM 算法之间；但在转弯场景下表现最差，碰撞率最高，通行时间也较长。

第五章 总结

在本文中，我们分别对传统算法 TTC 、PRM 与强化学习算法 DQN 、PPO 进行了实验。在对这些控制自动驾驶汽车驶过交叉路口的方法进行对比后，我们对这两类算法分别总结出了它们相较于对方的几个优势。

对于传统算法，它的优势在于：

1) 碰撞率低。图4-1显示，TTC 算法的碰撞率为 0 ，PRM 算法的碰撞率小于 0.1 % ，均远低于 DQN 和 PPO 算法 1 % 至 2 % 的碰撞率。
2) 易于实现。传统算法只使用了数学库，没有额外再导入代码库；而强化学习算法不仅仅需要使用数学库，还需要使用强化学习库 RLlib 、机器学习框架 Tensorflow 或 PyTorch 。如果没有现成的框架，从零开始实现强化学习算法，并最终能控制自动驾驶汽车驶过交叉路口，这将是非常庞大的工程量。
3) 可解释性强。TTC 算法及 PRM 算法原理简单，TTC 是根据最小碰撞时间、PRM 则是根据最大车流车辆风险来决定是否通过路口。算法中的每个参数都有明确的物理意义，增大或减小而产生的效果是可解释的，如 TTC 阈值的选定，若它太小则会导致本车没有足够的时间通过路口而与车流车辆发生碰撞，若它太大则车辆启动需要路口很长一段距离以外都没有车辆，可能会导致超时。而强化学习算法是基于神经网络构建的，神经网络中的参数可能并没有实际的含义，可解释性较弱，这就导致了即使碰撞重现，也无法定位到问题，无法进一步降低碰撞率。
4) 可定位错误。正是因为前述的可解释性强，自动驾驶汽车在实验的过程中遇到的种种情况都很容易定位到错误，包括模型本身的问题、参数问题等。而强化学习算法要定位到问题就比较困难了，除了要考虑模型的选择、训练参数是否正确，还要考虑网络结构、传入的参数需要经过何种变换等等问题。
5) 调试时间短。模型简单，易于实现，编码完成即可运行测试并记录实验结果；而强化学习算法还需要经过训练才能运行测试，从编码完成到得到测试结果需要的时间特别长。若代码出现了问题，则需要不断地重复“修改代码——训练——测试”这一过程，其中训练的时间占了很大的比重。

而强化学习算法的优势则在于：

1) 成功率高。图4-1指出，强化学习 DQN 算法及 PPO 算法的成功率均远高于传统的 TTC 算法，其中 DQN 算法的成功率更是与 PRM 算法持平。这得益于良好的模型设计、网络结构以及 DQN 算法计算出的较为精确的 Q 值。

2) 通过路口的时间短。我们清楚地看到，强化学习 DQN 算法及 PPO 算法通过路口的时间比传统的 TTC 算法低，其中 DQN 算法通过路口的时间相对于传统的 PRM 算法更是减少了 50 % [①]，而成功率仍能与 PRM 算法持平。这是因为强化学习算法能学习到更多的知识帮助判断，而不是简单地根据预设的规则启动或停止。
3) 无需人工设置规则。设置规则是一件非常复杂的事情，它需要我们考虑所有可能的情况，包括靠近路口时需要减速、进入路口前需要观察、当可能发生碰撞时不能通过路口等等，缺少了哪一条都有可能导致碰撞。而强化学习算法拥有很好的泛化能力，能自动学习这些规则，从而控制自动驾驶汽车安全地驶过交通路口。使用强化学习算法，将减轻我们设计规则的工作量。在实际应用中，我们只需要设置前置规则或者是兜底规则，即能使强化学习算法的性能大幅提高。
4) 规则灵活。传统算法所设置的规则是死板的，一般为“当满足什么条件时，执行什么操作”这种格式。有些时候，这些规则没有办法做到非常细致，从而会存在一些场景被误分类，导致执行规则指定的操作会比执行其它操作更容易碰撞，或是通过时间更长。而强化学习算法能充分评估当前场景，并能根据 Q 值准确地预测哪个操作能更容易、更快地通过交叉路口。
5) 能完全理解当前场景。传统算法完全不理解当前的场景，只会根据预设的规则判断该执行什么样的操作；而强化学习算法的规则完全是因为理解了当前场景生成并不断更新的。当两类算法分别遇到完全没出现过的场景时，强化学习算法更有可能独自正确地做出正确的操作。在路口环境发生变化的情况下（如增加一个车道、来车方向发生改变等），传统算法需要人工修改代码，使之能适应新的场景；强化学习算法只需要在不同的路口环境下进行训练，就能理解不同的路口环境下的场景，进而能自如地在这些场景下操作，而无需修改代码。

强化学习算法能学习到应在何时出发，也能学习到缓慢前行观察的行为，并且还能泛化样本以外的数据，但它偶尔还是会发生碰撞。因此，未来我们可以在以下几个方向上努力：

1) 传统算法与强化学习算法相结合。综合二者的优点，取其精华，去其糟粕，既能做到“零碰撞”，也能快速通过路口。
2) 提升算法的鲁棒性。强化学习算法从成功的案例中得到奖赏，从失败的案例中得到惩罚，然后把它们加在一起。有些碰撞事件是小概率发生的，在计算 Q 值时这种事件的分量较小，没有被强化学习算法留意到，最终导致的碰撞恰恰是这种小概率类型的碰撞。

[①] 取直行方向车流量为 0.8 辆/s 的数据：DQN 算法 9.29 s 、PRM 算法 19.17 s

3) 支持可视化策略和人工修改策略。目前强化学习算法学得的策略均保存在神经网络中的参数中，而无法像传统算法一样能让人理解。如能可视化策略，由人类观察到上述的小概率事件，修改策略并将修改后的参数应用到神经网络中，这种偶然的碰撞也将能够大幅减少。

参考文献:

[1] BIMBRAW K. Autonomous Cars: Past, Present and Future[C] // In Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO-2015). [S.l.]: SCITEPRESS (Science and Technology Publications, Lda.), 2015: 191 – 198.

[2] van der HORST R, HOGEMA J. TIME-TO-COLLISION AND COLLISION AVOIDANCE SYSTEMS[J], 1994.

[3] McGill S G, Rosman G, Ort T, et al. Probabilistic Risk Metrics for Navigating Occluded Intersections[J/OL]. IEEE Robotics and Automation Letters, 2019, 4(4): 4322 – 4329.
http://dx.doi.org/10.1109/LRA.2019.2931823.

[4] BOJARSKI M, TESTA D D, DWORAKOWSKI D, et al. End to End Learning for Self-Driving Cars.[J/OL]. CoRR, 2016, abs/1604.07316.
http://arxiv.org/abs/1604.07316.

[5] Isele D, Rahimi R, Cosgun A, et al. Navigating Occluded Intersections with Autonomous Vehicles Using Deep Reinforcement Learning[C/OL] // 2018 IEEE International Conference on Robotics and Automation (ICRA). 2018: 2034 – 2039.
http://dx.doi.org/10.1109/ICRA.2018.8461233.

[6] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning[J/OL]. Nature, 2015, 518(7540): 529 – 533.
https://doi.org/10.1038/nature14236.

[7] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal Policy Optimization Algorithms[J], 2017.

[8] HAYWARD J, TRANSPORTATION P, CENTER T S. Near Miss Determination Through Use of a Scale of Danger: Near Miss Determination Through Use of a Scale of Danger[M/OL]. [S.l.]: Pennsylvania Transportation and Traffic Safety Center, The Pennsylvania State University, 1972.
https://books.google.com/books?id=OxKWpwAACAAJ.

[9] Bouton M, Cosgun A, Kochenderfer M J. Belief state planning for autonomously navigating urban intersections[C/OL] // 2017 IEEE Intelligent Vehicles Symposium (IV). 2017: 825 – 830.
http://dx.doi.org/10.1109/IVS.2017.7995818.

[10] MAO H, ALIZADEH M, MENACHE I, et al. Resource Management with Deep Reinforcement Learning[C/OL] // HotNets '16: Proceedings of the 15th ACM Workshop on Hot Topics in Networks. New York, NY, USA: Association for Computing Machinery, 2016: 50–56.
https://doi.org/10.1145/3005745.3005750.

[11] LEVINE S, FINN C, DARRELL T, et al. End-to-End Training of Deep Visuomotor Policies[J]. J. Mach. Learn. Res., 2016, 17(1): 1334–1373.

[12] SUTTON R S, BARTO A G. Reinforcement Learning: An Introduction[M/OL]. Second. [S.l.]: The MIT Press, 2018.
http://incompleteideas.net/book/the-book-2nd.html.

[13] 周志华. 机器学习 [M/OL]. [S.l.] : 清华大学出版社, 2016 : 371 – 397.
https://books.google.com/books?id=j0G8nQAACAAJ.

[14] WATKINS C J C H, DAYAN P. Q-learning[J/OL]. Machine Learning, 1992, 8(3) : 279 – 292.
https://doi.org/10.1007/BF00992698.

[15] BELLMAN R. The theory of dynamic programming[J/OL]. Bull. Amer. Math. Soc., 1954, 60(6) : 503 – 515.
https://projecteuclid.org:443/euclid.bams/1183519147.

[16] LIANG E, LIAW R, NISHIHARA R, et al. RLlib: Abstractions for Distributed Reinforcement Learning[C/OL] // DY J, KRAUSE A. Proceedings of Machine Learning Research, Vol 80 : Proceedings of the 35th International Conference on Machine Learning. Stockholmsmässan, Stockholm Sweden : PMLR, 2018 : 3053 – 3062.
http://proceedings.mlr.press/v80/liang18b.html.

[17] SCHULMAN J, LEVINE S, ABBEEL P, et al. Trust Region Policy Optimization[C/OL] // BACH F, BLEI D. Proceedings of Machine Learning Research, Vol 37 : Proceedings of the 32nd International Conference on Machine Learning. Lille, France : PMLR, 2015 : 1889 – 1897.
http://proceedings.mlr.press/v37/schulman15.html.

[18] KAKADE S, LANGFORD J. Approximately Optimal Approximate Reinforcement Learning[C] // ICML ' 02 : Proceedings of the Nineteenth International Conference on Machine Learning. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2002 : 267–274.

[19] LOPEZ P A, BEHRISCH M, BIEKER-WALZ L, et al. Microscopic Traffic Simulation using SUMO[C/OL] // The 21st IEEE International Conference on Intelligent Transportation Systems. [S.l.] : IEEE, 2018.
https://elib.dlr.de/124092/.

[20] WU C, KREIDIEH A, PARVATE K, et al. Flow: A Modular Learning Framework for Autonomy in Traffic[C/OL] // . 2017.
https://arxiv.org/abs/1710.05465.

[21] BROCKMAN G, CHEUNG V, PETTERSSON L, et al. OpenAI Gym[C/OL] // . 2016.
http://arxiv.org/abs/1606.01540.

[22] TREIBER M, HENNECKE A, HELBING D. Congested Traffic States in Empirical Observations and Microscopic Simulations[J/OL]. Physical Review E, 2000, 62 : 1805 – 1824.
http://dx.doi.org/10.1103/PhysRevE.62.1805.

[23] KRAUSS S. Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics[C] // . 1998.

[24] SCHULMAN J, MORITZ P, LEVINE S, et al. High-Dimensional Continuous Control Using Generalized Advantage Estimation[J], 2015.

致谢

在整个毕业论文的实验、撰写过程中，我得到了指导老师张子臻教授、阿里云计算中心的帮助。在此我对他们给予我的帮助表达诚挚的谢意。

感谢指导老师张子臻教授。在选题阶段，张老师根据我未来的就业方向为我推荐了自动驾驶方面的课题，为我毕业后的工作着想；在实验阶段，张老师为我提供了一款开源软件 SUMO 以及一套开源的强化学习框架 Flow ，让我得以在其上创建一整套实验环境，进行训练和模拟仿真，最后完成实验。张老师对机器学习领域有很好的理解，以其渊博的知识给予我极具价值的技术性指导。在此，我向张老师致以崇高的敬意和衷心的感谢。

感谢阿里云计算中心为我提供有偿的计算资源。为节约训练时间，我在实验阶段使用了阿里云的服务器来进行训练。如果没有这种廉价的按量付费的计算服务，我要么就要以昂贵的价格购买高性能计算机，要么就得耗费大量的时间等待实验结果。阿里云的这项服务，为我提供了一个经济又高效的服务。

王凯祺

2020 年 6 月 8 日

毕业论文（设计）成绩评定记录

Grading Sheet of the Graduation Thesis (Design)

指导教师评语 Comments of Supervisor:

本论文通过开源平台 SUMO 研究自动驾驶汽车的控制策略，以模拟汽车通过交叉路口。论文采用深度强化学习框架，针对问题环境设计了 DQN 和 PPO 算法，并与经典的 TTC 与 PRM 算法进行了对比，实验效果良好。论文结构合理，描述清晰，提出的算法具有一定新颖性，实验能可视化。论文达到本科生毕业论文要求。

成绩评定 Grade:

指导教师签名 Supervisor Signature: Date:

答辩小组意见 Comments of the Defense Committee:

成绩评定 Grade:

签名 Signatures of Committee Members: Date:

院系负责人意见 Comments of the Academic Chief of School:

成绩评定 Grade:

签名 Signature: 院系盖章 Stamp: Date:

中山大學

SUN YAT-SEN UNIVERSITY