广东工业大学
GUANGDONG UNIVERSITY OF TECHNOLOGY

博士学位论文

车载边缘计算中的资源分配与协同感知技术研究

作　　者　　姓　　名：薛 拯

导　　师　　姓　　名：韩国军

学科（专业）或领域名称：信息与通信工程

论　文　答　辩　年　月：2024 年 5 月

中图分类号：　　学校代号：11845

UDC：　　密级：　　学　　号：1112103009

广东工业大学博士学位论文

（工学博士）

车载边缘计算中的资源分配与协同感知技术研究

薛拯

指导教师姓名、职称：	韩国军　教授
校外指导教师姓名、职称：	无
学科（专业）或领域名称：	信息与通信工程
学生所属学院：	信息工程学院
答辩委员会主席：	黄双萍　教授
论文答辩日期：	2024 年 5 月 26 日

A Dissertation Submitted to Guangdong University of Technology
in Partial Fulfillment of the Requirement
for the Degree of Doctor of Engineering Science

Research on Resource Allocation and Collaborative Perception Technologies in Vehicular Edge Computing

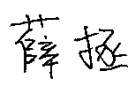
Candidate: XUE Zheng

Supervisor: HAN Guojun

May 26, 2024
School of Information Engineering
Guangdong University of Technology
Guangzhou, Guangdong, P. R. China, 510006

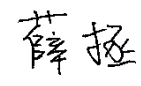
学位论文独创性声明

本人郑重声明：所呈交的学位论文是我个人在导师的指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明，并表示了谢意。本人依法享有和承担由此论文所产生的权利和责任。

论文作者签名：  日期：2024 年 6 月 11 日

学位论文版权使用授权声明

本学位论文作者完全了解学校有关保存、使用学位论文的规定：“研究生在广东工业大学学习和工作期间参与广东工业大学研究项目或承担广东工业大学安排的任务所完成的发明创造及其他技术成果，除另有协议外，归广东工业大学享有或特有”。同意授权广东工业大学保留并向国家有关部门或机构送交该论文的印刷本和电子版本，允许该论文被查阅和借阅。同意授权广东工业大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印、扫描或数字化等其他复制手段保存和汇编本学位论文。保密论文在解密后遵守此规定。

论文作者签名：  日期：2024 年 6 月 11 日

指导教师签名：  日期：2024 年 6 月 11 日

摘要

为解决单车智能面临的安全长尾问题，我国提出了车路云一体化协同创新发展模式，这一模式利用蜂窝车联网、边缘计算、云计算技术，将人、车、路、云的物理空间和信息空间融为一体，形成高效运行的信息物理系统，即车路云一体化系统，但随着计算密集型和时延敏感型应用的不断增加，传统的资源调度方案难以满足车载终端对计算资源日益增高的需求，系统引入车载边缘计算范式，它通过提供灵活的计算、存储和通信资源，以及实施有效的任务卸载和资源分配策略，帮助降低任务处理的时延。然而，作为服务提供方，车载边缘计算面临资源不确定性和动态变化性的挑战，这使得其难以灵活匹配多样化的车辆计算任务需求。为解决以上问题，本文从车路云一体化系统整体到局部开展任务计算环境的资源配置优化研究，确保计算环境的高效性和可靠性。在资源配置优化研究的基础上，进一步针对网联自动驾驶中的协同任务应用进行了深入研究，具体包括协同位姿优化及目标级协同感知系统应用。通过从资源配置优化到协同任务应用的全面研究，本文为车载边缘计算系统中的任务计算环境优化和协同感知任务的高效执行提供了重要的理论支持和实践参考。主要研究内容及创新如下:

（1）面向程序关联性任务的车载边缘计算系统资源配置优化。针对多车辆单任务动态请求环境下不合理的程序缓存和任务卸载决策导致系统时延和能耗成本过高的问题，首先，提出了一种面向程序关联性任务的服务缓存与计算卸载框架，基于计算、通信、能耗模型分析不同缓存情况下的任务完成时延和能耗。然后，考虑存储和计算约束，建立了最小化长期累计任务完成时延的混合整数非线性规划问题，并将其建模为马尔可夫决策过程，提出基于深度确定性策略梯度的强化学习算法来进行动态缓存和卸载决策。最后，构建了仿真实验模型并进行了性能评估，算法具有良好的收敛性，所提策略避免了最大时延情况下的缓存决策，显著改善系统在时延和能耗方面的综合性能。

（2）面向网联自动驾驶的协同感知任务资源配置优化。为解决网联自动驾驶中协同感知任务的计算延迟和资源分配问题，首先，提出协同感知感兴趣域的任务计算需求与服务场景，引出任务阶段选择、卸载以及卸载后的资源分配问题。基于计算和传输模型，考虑容忍时延、卸载和分配约束，建立最小化感兴趣域任务完成时延的混合整数非线性规划问题。然后，提出双层最优任务卸载与资源分配算法进行决策。最后，构建仿真实验模型并进行性能评估，证明了所提方案能有效降低协同感知任务的计算

延迟。

（3）面向网联自动驾驶的协同位姿优化及应用。协同感知任务依赖准确的位姿关系，但实际场景中存在非理想通信及各种信息源误差的问题，这会影响协作质量。针对此问题，建立了关于车-车、车-特征、车-动态障碍物以及高精地图和导航定位信息的因子图约束模型，形式化定义了最小化残差的状态估计问题，并采用列文伯格-马夸尔特法求解优化问题。由仿真分析可知，所提方案在不同观测误差情况下，能够显著提升协同定位性能。为推动自动驾驶协同感知研究向实际应用发展，通过协同感知仿真框架评估了不同融合策略的性能优劣，由于目标级协同感知具备良好的兼容性和可拓展性，搭建了基于车载终端和计算单元的硬件在环试验平台，并开发了配套的软件环境进行检测模型和融合算法部署。目标级协同感知系统原型具备可行性和有效性，能够满足单车感知能力拓展需求，实现提醒预警等功能。

关键词：车联网；车载边缘计算；资源分配；协同感知；任务卸载

ABSTRACT

To address the long-tail safety issues in autonomous driving, China has proposed an innovative development model for vehicle-road-cloud integration. This model leverages cellular vehicle-to-everything (C-V2X), edge computing, and cloud computing technologies to unify the physical and information spaces of people, vehicles, roads, and clouds, forming an efficiently operating cyber-physical system known as the vehicle-road-cloud integrated system (VRCIS). However, with the increasing demand for computationally intensive and latency-sensitive applications, traditional resource scheduling schemes struggle to meet the growing computational resource needs of in-vehicle terminals. To address this challenge, the system introduces the paradigm of vehicular edge computing (VEC), which provides flexible computing, storage, and communication resources, and implements effective task offloading and resource allocation strategies to reduce task processing latency. Nevertheless, as a service provider, VEC faces challenges related to resource uncertainty and dynamic variability, making it difficult to flexibly match the diverse computational task requirements of vehicles. To address these issues, this thesis conducts a resource allocation optimization study for task computing environments from the overall VRCIS down to individual components, ensuring the efficiency and reliability of the computing environment. Based on the resource allocation optimization study, further in-depth research on collaborative task applications in connected autonomous driving is conducted, specifically including collaborative pose optimization and object-level collaborative perception system applications. Through comprehensive research from resource allocation optimization to collaborative task applications, this thesis provides essential theoretical support and practical reference for optimizing task computing environments and efficiently executing collaborative perception tasks in VEC systems. The main research content and innovations are as follows:

(1) Optimization of resource allocation for VEC systems targeting program-associated tasks. To address the issues of high system latency and energy cost caused by unreasonable program caching and task offloading decisions in a dynamic multi-vehicle single-task request environment, this part first proposes a service caching and computation offloading framework

for program correlation tasks. It analyzes task completion latency and energy consumption under different caching conditions based on computation, communication, and energy models. Next, considering storage and computation constraints, a mixed-integer nonlinear programming problem is established to minimize long-term cumulative task completion latency, modeled as a Markov decision process. A reinforcement learning algorithm based on deep deterministic policy gradient is then proposed for dynamic caching and offloading decisions. Finally, a simulation model is constructed and performance evaluations are carried out. The algorithm demonstrates good convergence. The proposed strategy avoids caching decisions under maximum delay conditions, significantly enhancing the system's overall performance in terms of latency and energy consumption.

(2) Resource configuration optimization for collaborative perception tasks in connected autonomous driving. First, the computational requirements and service scenarios for tasks within the collaborative perception region of interest are proposed, leading to the issues of task phase selection, offloading, and resource allocation after offloading. Based on computation and transmission models, and considering delay tolerance, offloading, and allocation constraints, a mixed-integer nonlinear programming problem is established to minimize the task completion latency for the region of interest. A two-layer optimal task offloading and resource allocation algorithm for decision-making is then proposed. Finally, a simulation experiment model is constructed and performance evaluations are conducted, demonstrating that the proposed strategy effectively reduces the computational delay in collaborative perception tasks.

(3) Collaborative pose optimization and applications for connected autonomous driving. Collaborative perception tasks rely on accurate pose relationships. In actual scenarios, non-ideal communication and various information source errors can affect collaboration quality. To address this issue, a factor graph constraint model about vehicle-vehicle, vehicle-feature, vehicle-dynamic obstacles, as well as high-precision maps and navigation positioning information is established. The state estimation problem minimizing residuals is formally defined, and the Levenberg-Marquardt method is employed to solve the optimization problem. Simulation

analysis indicates that the proposed solution significantly improves collaborative positioning performance under various observation error conditions. To advance the research on collaborative perception in autonomous driving towards practical application, a collaborative perception simulation framework is used to evaluate the performance of different fusion strategies. Due to the compatibility and extensibility of target-level collaborative perception, a hardware-in-the-loop test platform based on vehicle terminals and computing units is built, and a supporting software environment is developed to deploy detection models and fusion algorithms. The prototype system for object-level collaborative perception is feasible and effective, meeting the needs of expanding standalone vehicle perception capabilities and realizing functions such as alerting and warning.

Key words: Internet of Vehicles; vehicular edge computing; resource allocation; cooperative perception; task offloading

目录

摘要 I
ABSTRACT III
目录 VI
CONTENTS IX
缩略词对照表 XII
第一章 绪论 1
1.1 研究背景及研究意义 1
1.2 国内外相关研究现状 6
1.2.1 车载边缘计算服务架构 6
1.2.2 车联网资源分配与计算卸载策略 7
1.2.3 车联网协同感知 10
1.2.4 主要问题总结 12
1.3 研究目标和主要研究内容 13
1.3.1 研究目标 13
1.3.2 主要研究内容 13
1.4 论文组织结构 15
第二章 车载边缘计算系统关键技术 17
2.1 引言 17
2.2 车载边缘计算系统 17
2.2.1 系统架构 17
2.2.2 计算卸载技术 21
2.3 深度强化学习 23
2.3.1 强化学习 23
2.3.2 基于价值迭代的深度强化学习 25
2.3.3 基于策略迭代的深度强化学习 28
2.4 协同感知 28
2.5 本章小结 31

第三章 面向程序关联性任务的车载边缘系统资源配置优化 32
3.1 引言 32
3.2 车载边缘计算系统服务缓存与计算卸载模型 33
3.2.1 系统模型概述 33
3.2.2 通信模型 34
3.2.3 服务缓存与计算卸载 35
3.2.4 计算延迟和能耗分析 36
3.2.5 优化问题描述 43
3.3 基于深度强化学习的服务缓存与计算卸载策略 44
3.3.1 马尔可夫决策过程 44
3.3.2 基于深度确定性策略算法的缓存与卸载方案 46
3.4 实验仿真及结果分析 49
3.4.1 实验参数设置 49
3.4.2 结果分析 51
3.5 本章小结 54
第四章 面向网联自动驾驶的协同感知任务资源配置优化 55
4.1 引言 55
4.2 协同感知任务卸载模型 57
4.2.1 系统模型 57
4.2.2 计算卸载模型 58
4.2.3 优化问题描述 60
4.3 双层最优任务卸载与资源分配算法设计 61
4.4 实验仿真及结果分析 65
4.4.1 实验参数设置 65
4.4.2 结果分析 66
4.5 本章小结 70
第五章 面向网联自动驾驶的协同位姿优化及应用 71
5.1 引言 71

5.2 网联协同自动驾驶协同位姿优化框架 72
5.2.1 系统状态描述 72
5.2.2 状态估计器 75
5.2.3 实验仿真及结果分析 77
5.3 目标级协同感知应用 82
5.3.1 仿真实验 82
5.3.2 硬件在环实测平台 85
5.3.3 性能评估 90
5.4 本章小结 94
结论与展望 95
参考文献 98
攻读学位期间取得与学位论文相关的成果 117
致谢 119

CONTENTS

ABSTRACT(IN CHINESE) I
ABSTRACT(IN ENGLISH) III
CONTENTS(IN CHINESE) VI
CONTENTS(IN ENGLISH) IX
Table of physical quantity names and symbols XII
Chapter 1 Introduction 1
1.1 Background and significance of research 1
1.2 Analysis of the research status at home and abroad 6
1.2.1 Architecture of vehicular edge computing services 6
1.2.2 Resource allocation and computation offloading in vehicular networks 7
1.2.3 Cooperative perception in vehicular networks 10
1.2.4 Summary of primary challenges 12
1.3 Research objectives and main contents 13
1.3.1 Research objectives 13
1.3.2 Research contents 13
1.4 Organization and content arrangement 15
Chapter 2 Key technologies in vehicular edge computing systems 17
2.1 Perface 17
2.2 Vehicular edge computing 17
2.2.1 The system architecture 17
2.2.2 Computation offloading technology 21
2.3 Deep reinforcement learning 23
2.3.1 Reinforcement learning 23
2.3.2 Deep reinforcement learning based on value iteration 25
2.3.3 Deep reinforcement learning based on policy iteration 28
2.4 Cooperative perception 28
2.5 Chapter summary 31

Chapter 3 Optimization of resource allocation for task offloading of program-centric tasks in vehicular edge computing systems 32
3.1 Perface 32
3.2 Service caching and computation offloading scenario in vehicular edge computing systems 33
3.2.1 Overview of system model 33
3.2.2 Communication model 34
3.2.3 Service Caching and Task Offloading 35
3.2.4 Analysis of computation delay and energy consumption 36
3.2.5 Problem formulation 43
3.3 Service caching and computation offloading scheme based on deep reinforcement learning 44
3.3.1 Markov decision processes 44
3.3.2 DDPG-Based edge caching and offloading scheme 46
3.4 Simulation experimentation and result analysis 49
3.4.1 Experimental parameter settings 49
3.4.2 Result analysis 51
3.5 Chapter summary 54
Chapter 4 Optimization of resource allocation for cooperative perception tasks in connected autonomous driving 55
4.1 Perface 55
4.2 Model for cooperative perception offloading 57
4.2.1 System model 57
4.2.2 Computation offloading model 58
4.2.3 Problem formulation 60
4.3 Design of two-level optimal task offloading and resource allocation algorithm 61
4.4 Simulation experimentation and result analysis 65
4.4.1 Experimental parameter settings 65

4.4.2 Result analysis 66
4.5 Chapter summary 70
Chapter 5 Collaborative pose optimization and applications for connected autonomous driving 71
5.1 Perface 71
5.2 Collaborative pose optimization framework for connected cooperative autonomous driving 72
5.2.1 Description of system states 72
5.2.2 State Estimator 75
5.2.3 Simulation experimentation and result analysis 77
5.3 Application of object-level cooperative perception 82
5.3.1 Simulation experiment 82
5.3.2 Hardware-in-loop testing platform 85
5.3.3 Performance evaluation 90
5.4 Chapter summary 94
Conclusion and prospect 95
References 98
Publication and patents during study 117
Acknowlegements 119

缩略词对照表

缩略词	英文全称	中文名称
3GPP	The 3rd Generation Partnership Project	第三代合作伙伴计划
5G	The 5th Generation Mobile Communication	第五代移动通信
A2C	Advantage Actor Critic	优势演员评论家
AC	Actor Critic	演员评论家
AI	Artificial Intelligence	人工智能
AP	Average Precision	平均精度
AR	Augmented Reality	增强现实
AV	Autonomous Driving	自动驾驶
BEV	Bird Eye View	鸟瞰图
CACIV	China Industry Innovation Alliance for the Intelligent and Connected Vehicles	中国智能网联汽车产业创新联盟
CCSA	China Communications Standards Association	中国通信标准化协会
CDN	Content Delivery Network	内容分发网络
C-ITS	China ITS Industry Aliance	中国智能交通产业联盟
CNN	Convolutional Neural Networks	卷积神经网络
CPS	Cyber Physical Systems	信息物理系统
CSAE	China Society of Automotive Engineers	中国汽车工程学会
C-V2X	Cellular Vehicle-to-Everything	蜂窝车联网
DDPG	Deep Deterministic Policy Gradient	深度确定性策略梯度
DPG	Deterministic Policy Gradient	确定性策略梯度
DQN	Deep Q-network	深度 Q 网络
DRL	Deep Reinforcement Learning	深度强化学习
DSRC	Dedicated Short Range Communication	专用短程通信
FCC	Federal Communications Commission	联邦通信委员会
GNSS	Global Navigation Satellite System	全球导航卫星系统
ICV	Intelligent Connected Vehicle	智能网联汽车
IEEE	Institute of Electrical and Electronics Engineers	电气与电子工程师协会
IMT-2020	International Mobile Telecommunications-2020	国际移动通信 2020 推进组
IoT	Internet of Things	物联网

IoU	Intersection over Union	交并比
IoV	Internet of Vehicles	车联网
ITS	Intelligent Transportation System	智能交通系统
LTE	Long Term Evolution	长期演进
MAC	Medium Access Control	媒体接入层
MDP	Markov Decision Process	马尔可夫决策过程
MEC	Mobile Edge Computing	移动边缘计算
MINLP	Mixed Integer Nonlinear Programming	混合整数非线性规划
MLP	Muiltiplayer Perceptions	多层感知机
NFV	Network Functions Virtualization	网络功能虚拟化
NOMA	Non-orthogonal Multiple Access	非正交多址
NR	New Radio	新空口
OBU	On Board Unit	车载终端
QoS	Quality of Service	服务质量
RL	Reinforcement Learning	强化学习
RMSE	Root Mean Square Error	均方根误差
RoI	Regions of Interest	感兴趣域
ROS	Robot Operating System	机器人操作系统
RPN	Region Proposal Network	区域提案网络
RSU	Road Side Unit	路侧单元
SDN	Software Defined Network	软件定义网络
SINR	Signal-to-Interference-plus-Noise-Ratio	信号加干扰噪声比
SVM	Support Vector Machines	支持向量机
SLAM	Simultaneous Localization and Mapping	同步定位与建图
TD3	Twin Delayed Deep Deterministic Policy Gradient	双延迟深度确定性策略梯度
TD	Temporal Difference	时间差分
TDMA	Time Division Multiple Access	时分多址
UDP	User Datagram Protocol	用户数据报协议
URLLC	Ultra Reliable and Low Latency Communications	超高可靠低时延通信
V2I	Vehicle-to-Infrastructure	车与基础设施通信

V2N	Vehicle-to-Network	车与网络通信
V2P	Vehicle-to-Pedestrian	车与人通信
V2V	Vehicle-to-Vehicle	车与车通信
V2X	Vehicle-to-Everything	车联万物
VEC	Vehicular Edge Computing	车载边缘计算
VR	Virtual Reality	虚拟现实
VRCIS	Vehicle-Road-Cloud Integrated System	车路云一体化系统
WAVE	Wireless Access in Vehicular Environments	无线接入车载环境

第一章 绪论

1.1 研究背景及研究意义

车联网（Internet of Vehicles，IoV）技术由于在降低单车智能成本、加快自动驾驶实现、提升道路交通安全和效率等方面的优越性，车联网及其赋能的智能网联汽车（Intelligent Connected Vehicle，ICV）和智能交通系统（Intelligent Transportation System，ITS）已上升为国家层面的重要战略。2018 年工信部、交通部、公安部、国标委等多部委先后发布《国家车联网产业标准体系建设指南》系列顶层设计文件，明确智能网联汽车、信息通信、智能交通、车辆智能管理、电子产品与服务标准体系建设目标，汽车、智能交通、通信及交通管理四方标准化技术委员联合推进蜂窝车联网（Cellular Vehicle-to-Everything，C-V2X）相关标准体系建设，如图1-1所示。同时联盟、学会等社会组织也在积极推动相关标准研制，包括国际移动通信 2020 推进组（International Mobile Telecommunications-2020，IMT-2020）成立的 C-V2X 工作组、中国通信标准化协会（China Communications Standards Association，CCSA）、中国汽车工程学会（China Society of Automotive Engineers，CSAE）、中国智能网联汽车产业创新联盟（China Industry Innovation Alliance for the Intelligent and Connected Vehicles，CAICV）、中国智能交通产业联盟（China ITS Industry Aliance，C-ITS）等。单车智能发展面临感知局限性等长尾难题短期内难以得到有效解决，制约高等级自动驾驶落地，因此各国已经开始探索智能化网联化融合发展路径，为抓住全球汽车产业革命和我国交通产业变革的重大战略机遇期，我国提出了基于 C-V2X 的“聪明的车 + 智慧的路 + 协同的云”的车路云一体化协同创新发展模式[1, 2]。

智能交通的车路协同和汽车产业变革的自动驾驶等都离不开车联网。车联网是指根据规定的通信协议和数据交互标准，在“人-车-路-云”之间进行信息交换的通信网络。车联网无线通信技术是实现车辆与周围的车辆、行人、交通基础设施和云（平台）等全方位连接和通信的新一代信息通信技术，车辆通过装备车载终端（On Board Unit，OBU）实现 V2X 通信，包括车与车之间（Vehicle-to-Vehicle，V2V）通信、车与路之间（Vehicle-to-Infrastructure，V2I）通信、车与人之间（Vehicle-to-Pedestrian，V2P）通信、车与网络/云（平台）之间（Vehicle-to-Network）通信等[3]（如图1-2所示），其中 V2V、V2I 和 V2P 具有低时延、高可靠等特殊严苛的通信要求，而 V2N 没有严格的时延与可

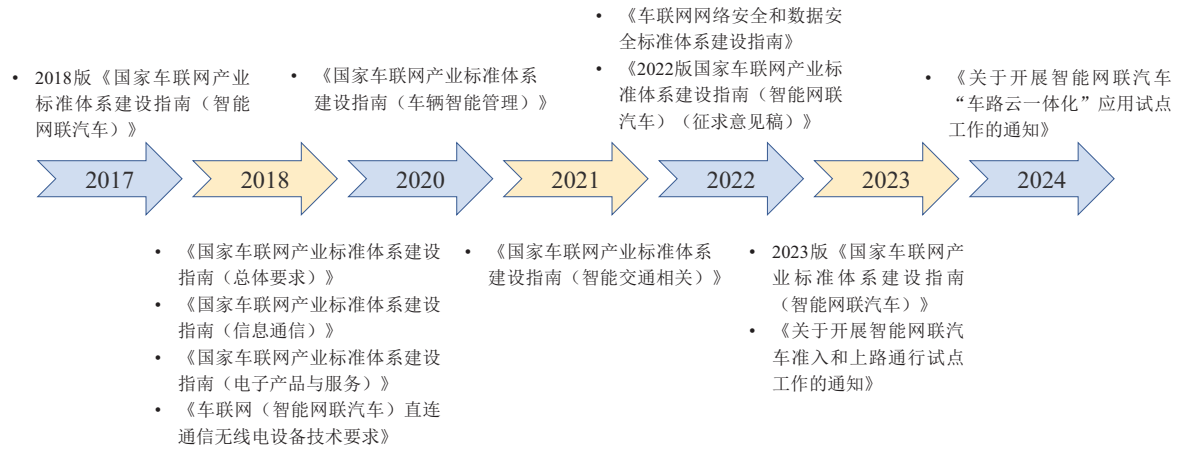


图 1-1 国家顶层政策出台时间线

Figure 1-1 Timeline of national top-level policy implementation

靠性要求。车联网赋能汽车和交通行业进而提高交通安全和出行效率，车路协同能够带动汽车、通信、交通、半导体等产业跨越式升级，加速构建自动化、智能化的交通出行体系。

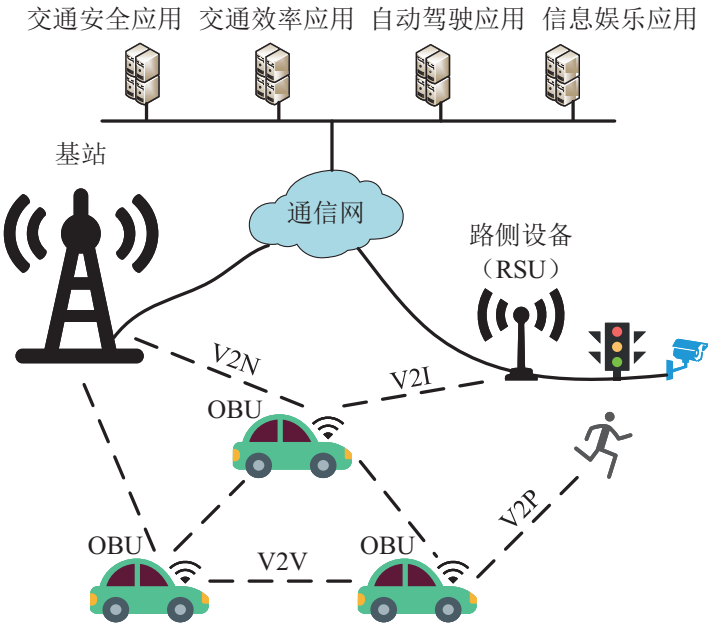


图 1-2 V2X 通信类型

Figure 1-2 Types of V2X communication

车联网技术标准分为电气和电子工程师协会（Institute of Electrical and Electronics Engineers, IEEE）提出的 IEEE 802.11p 和国际标准组织第三代合作伙伴计划（The 3rd Generation Partnership Project, 3GPP）提出的 C-V2X。IEEE 802.11p 协议定义了物理层

和媒体接入层（Medium Access Control，MAC），车载环境的无线接入（Wireless Access in Vehicular Environments，WAVE）包括 IEEE 802.11p 以及之上的 IEEE 1609 系列标准，以便支持应用层和性能标准。专用短程通信（Dedicated Short Range Communication，DSRC）在美国专指与 WAVE 有关的技术，即基于 IEEE 802.11p 和 IEEE 1609 系列标准的车联网技术。1999 年 10 月美国联邦通信委员会（Federal Communications Commission，FCC）将 5.850~5.925 GHz 共 75 MHz 频段分配给运输服务领域的短程通信[1]。IEEE 802.11p 标准在 2010 年制定完成，有测试验证和部分应用，但由于可扩展性差以及无法很好地应对高速移动的车联网通信提出的挑战，虽然已在美国、欧洲进行了十余年的研究和测试评估，但由于通信性能、部署成本等方面的原因，其商用进展一直不理想[4]。

蜂窝移动通信具有覆盖广、容量大、可靠性高的优点、具有产业规模优势，将蜂窝通信技术和短程直通通信技术有机结合，解决车车和车路间的低时延、高可靠通信难题，是兼顾技术和成本优势的选择，该构想最早由无线移动通信国家重点实验室的陈山枝博士提出，积极推动 C-V2X 的国际标准化及其演进[4–6]。C-V2X 技术在 3GPP 的标准化分为两个阶段，包括长期演进（Long Term Evolution，LTE）和基于第五代移动通信（The 5th Generation Mobile Communication，5G）新空口（New Radio，NR）的 V2X 标准，二者互相补充，标准设计充分考虑了前后向兼容性。作为比 IEEE 802.11p 后发起研究的车联网技术，C-V2X 尽量重用蜂窝系统已有的上层协议，重点聚焦于接入层的物理层和媒体接入层技术。2015 年初，3GPP 正式启动基于 C-V2X 的技术需求和标准化研究，于 2017 年 3 月发布 R14 版本 LTE-V2X 标准，面向基本的道路安全业务通信需求，引入了工作在 5.9GHz 频段的直通链路（PC5 接口）通信方式，并对公众移动蜂窝网通信接口（Uu 接口）进行了优化。3GPP R15 版本于 2018 年 6 月完成对 LTE-V2X 的增强标准化工作，在 PC5 接口引入了载波聚合、高阶调制、时延降低和传输分集等技术。2020 年 6 月，首个支持 NR-V2X 的 3GPP R16 版本冻结，面向增强车联网应用，完成支持 5G NR 的直连通信能力技术方案。2022 年 6 月，发布 R17 版本，针对直通链路特性增强资源分配机制，支持车辆间协调、功耗节约机制等。同年启动 R18 研究，展开进一步增强研究，预计于 2024 年 6 月冻结。具体的演进时间表如图1-3所示。LTE-V2X 提供基于 PC5 接口的 V2V、V2I 和 V2P 的通信能力，同时也支持基于 Uu 接口的网络

转发的 V2X 通信方式，为了满足低时延传输需求，在直通链路上，支持集中式调度（模式 3）和分布式调度（模式 4），其中模式 3 由网络集中调度 PC5 接口资源，模式 4 由各通信节点在资源池中分布式调度 PC5 接口资源。NR-V2X 用来支持面向自动驾驶的各类增强车联网业务，提供更可靠、时延更短以及数据传输速率更高的通信服务。

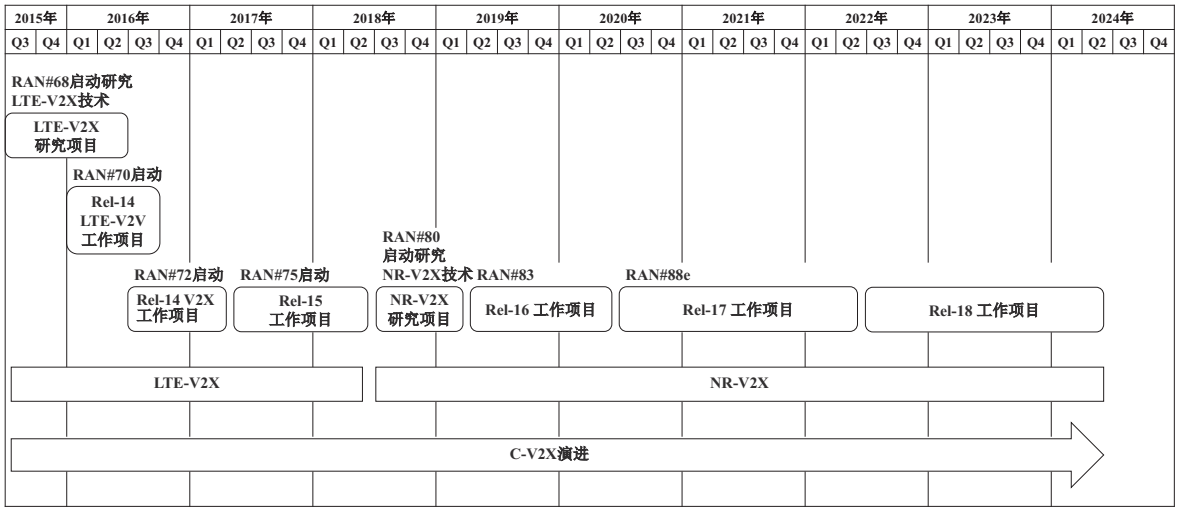


图 1-3 3GPP C-V2X 标准演进时间表

Figure 1-3 3GPP C-V2X standard evolution timeline

IEEE 802.11p 和 C-V2X 从物理层设计、MAC 层调度等无线通信角度对比分析，C-V2X 在低时延、高可靠性、资源利用率等方面具有技术优势。2018 年 11 月，工业和信息化部规划 5905-5925 MHz 频段作为基于 LTE-V2X 技术的车联网（智能网联汽车）直连通信的工作频段[7]。2020 年 11 月，美国 FCC 发布声明，决定放弃 DSRC，并将 30 MHz（5.895~5.925 GHz）分配给 C-V2X 技术，以加速 ITS 服务的实际部署。目前，C-V2X 已得到中国、美国等国家采用，成为全球车联网唯一事实标准。我国已经初步形成较为完备的车联网 LTE-V2X 全协议栈标准体系，依托 C-V2X 技术，我国车联网产业链条日渐完善，一汽、上汽、广汽、北汽、长城、蔚来等多家车企已实现基于 LTE-V2X 的 V2V、V2I 等提醒预警应用的前装量产。C-V2X“三跨”、“四跨”、“新四跨”系列先导应用实践活动不断推动 C-V2X 融合技术应用落地[8–11]。

自动驾驶是人工智能领域未来五年最复杂的任务之一，得益于 C-V2X、边缘计算、云计算等技术的发展和应用，车路协同有效补充单车智能面临的安全长尾，兼顾设计运行范围和经济性。车路协同与单车智能相辅相成，是自动驾驶的高阶发展形态和必然趋势[12]。车路云一体化系统网联化技术分为辅助信息交互、协同感知、协同决策、协

同控制。其中，辅助信息交互以及部分协同感知级别的产品与功能目前已经在量产和示范中得到了广泛应用。

国内外众多高校、科研机构围绕车联网、自动驾驶、车路协同、智能交通等领域进行深入探索和研究。清华大学汽车安全与节能国家重点实验室主任李克强院士的团队在智能网联汽车“中国方案”技术体系的提出和发展做出重要贡献[13–15]。无线移动通信国家重点实验室主任陈山枝博士团队长期致力于推动我国 C-V2X 车联网技术演进、标准制定与产业落地[6]。北京大学电子学院程翔教授团队推动复杂高速移动车联网信道建模和系统设计，探索通信与多模态感知联觉理论[16–18]。西安电子科技大学智慧交通研究院院长毛国强教授团队在车联网数据融合、高精度无线定位等领域取得重要进展[19–21]。深圳大学物联网研究中心主任 Victor C. M. Leung（梁中明）教授团队专注于车联网网络资源优化、缓存与卸载策略研究，取得系列重要研究成果[22–25]。清华大学国家研究中心智慧网联重点实验室牛志升教授团队专注智能和绿色的互联和移动智能，在边缘智能移动通信与计算领域做出系列贡献[26–29]。重庆大学刘凯教授团队提出异构融合车联网架构，完善了车联网系统架构、通信协议和理论体系[30–32]。

国际上，加拿大滑铁卢大学 Sherman Shen 教授团队在车联网资源管理、网络切片、边缘计算[33, 34] 领域做出了重要贡献。特拉华大学 Weisong Shi 教授团队一直倡导边缘计算范式[35]，在车载边缘计算、自动驾驶（Autonomous Driving，AV）领域取得一系列重要成果[36–38]。瑞典奥斯陆大学 Yan Zhang 教授团队在移动边缘计算（Mobile Edge Computing, MEC）、边缘智能、车联网安全领域产出不少具有影响力的成果[39–42]。美国休斯敦大学 Zhu Han 教授团队在车联网资源分配、安全以及博弈论应用领域展开深入研究，取得一系列重要研究成果[43–46]。加拿大卡尔顿大学 F.Richard Yu 教授团队在智能网联汽车安全、资源分配、信息物理融合系统方面做出重要贡献[47–49]。日本东北大学 Nei Kato 在车联网边缘智能、资源分配、智能反射面等课题取得一系列研究成果[50, 51]。

车路云一体化架构在行业内已初步形成共识，5G 等新一代的移动通信基础设施、智能化路侧交通基础设施、多级云控平台等共同构成了车路云一体化的新型信息基础设施。信息基础设施的建设中，运营商的 5G、移动边缘计算和 C-V2X 是重要组成部分，全面赋能智能网联应用实现在本地化的数据快速上传、边缘存储计算、决策结果下发执行的闭环，并通过边缘-区域-中心云的三级架构，为整个车城网平台系统，提供全面

的“通信-算力-应用-运营-安全”的支撑。基于 V2X 的车路协同自动驾驶通过信息交互协同、协同感知与协同决策控制，可以极大地拓展单车的感知范围，提升感知的能力，引入高维数据为代表的新的智能要素，实现群体智能。可以从本质上解决单车智能自动驾驶遇到的技术瓶颈。然而，在数据处理方面，车载计算系统难以完成对大量异构传感器数据的实时处理，其中仅摄像头每秒就产生 1.8 GB 的数据，将边缘计算应用到自动驾驶领域将有助于解决自动驾驶汽车在环境数据获取和处理上所面临的问题[52]。同时，车联网中存在大量异构节点，智能网联汽车行为受到边缘端、云端的决策和需求的影响，大量感知计算任务需要卸载到边缘计算服务节点处理。此外，由于车辆节点高速移动，单车感知范围有限及遮挡问题，智能网联汽车面临感知任务信息不准确和不完整等难题，在宝贵的车载边缘计算、存储资源下，如何低时延完成更多计算任务，同时降低能量损耗，提升感知任务完成质量是车载边缘计算网络面临的重要挑战。

车载边缘计算中的资源分配与协同感知的关系是相辅相成的，资源分配在车载边缘计算中指的是如何有效地分配和管理边缘计算节点（例如车载计算单元、路侧单元等）的计算资源、存储资源和网络带宽等。由于车载边缘计算环境的动态性和资源的有限性，合理的资源分配策略是确保系统高效运行的关键。协同感知是指通过多个车辆和基础设施之间的合作共享感知数据，以提升整体感知的准确性和可靠性。例如，车辆可以共享摄像头、雷达等传感器数据，形成一个更全面的环境感知视图，从而支持更安全和智能的驾驶决策。协同感知任务通常需要大量的计算资源和网络带宽来处理和传输感知数据。合理的资源分配策略可以确保这些需求得到满足，从而保证协同感知任务的高效执行。资源分配和协同感知的共同目标是提升车载边缘计算系统的整体性能。资源分配通过高效利用计算和网络资源，支持协同感知任务的执行。而协同感知通过共享和融合多源感知数据，提高感知精度和系统可靠性。深入研究车载边缘计算在资源分配和协同感知方面的问题，对于提升自动驾驶的安全性和效率、实现车联网的安全通信和智能交通具有至关重要的实际意义。

1.2 国内外相关研究现状

1.2.1 车载边缘计算服务架构

随着电信技术的飞速发展和网络应用的激增，传统的云网络架构由于回程链路负担过重、时延过长而无法很好的满足各种应用需求。边缘计算通过在网络边缘提供缓

存、计算和通信资源，使网络功能更接近最终用户，成为一种有前途的范例。此外，结合软件定义网络（Software Defined Network，SDN）[53] 分离数据平面和控制平面，实现高度灵活的数据调度策略和网络功能虚拟化（Network Functions Virtualization，NFV）技术，边缘计算可以从可扩展性和降低成本方面为网络生态系统提供各种优势[54]。车载边缘计算（Vehicular Edge Computing，VEC）基于移动边缘计算的动机和基础，提供和管理更接近车辆和最终用户的存储和计算资源，以较低的延迟提供对服务的访问，并满足服务类型的最低执行要求[55]。以 VEC 为基础的架构可以分为多层，仅考虑固定层和车辆边缘层的两层结构[56, 57]，固定层负责控制和在车辆边缘层之间建立通信、收集信息并管理可用资源，这也会导致更大的开销和更少的可用资源。三层的 VEC 架构使用不同的控制器来支持聚合和管理功能，增加了控制器层（也称为云层），该层具备更强大的计算能力，为用户提供服务、管理所有可用资源，给系统带来更大的灵活性[58, 59]。四层 VEC 架构进一步增加了应用层，遵循 SDN 范式，但将资源分配、服务提供、应用支持交给了应用层，控制器层专注构成 VEC 组件之间的流管理和数据转发[60]。两层的架构在可用资源方面存在限制，从而影响用户可用的服务数量，而更高层数则具有更高的复杂性，带来额外的架构组件。尽管数据传输有更多的资源和灵活性，但控制消息的开销很大。三层架构比四层更简单，需要优化机制来管理整个系统以及计算存储资源的分配和管理。

1.2.2 车联网资源分配与计算卸载策略

车联网中与资源分配相关的研究一直是学术热点，包括通信、计算、存储的单一及联合优化，与车联网中的应用业务息息相关，如视频流内容的放置、传输，感知任务、增强现实（Augmented Reality，AR）/虚拟现实（Virtual Reality，VR）相关的计算任务需求等[61, 62]。LIANG L 等人将图论、多智能体强化学习引入用于解决车联网资源分配问题，针对大尺度衰落的信道状态信息和延迟信道状态信息分别进行设计，提高通信可靠性[63, 64]。WU W 等人考虑不完美信道信息开发了联合信道训练和资源分配算法以满足异构应用的服务质量（Quality of Service，QoS）需求[65]。YANG H 等人提出一种帧设计算法和半持久调度算法，在合理复杂度下实现最优帧设计和资源分配，满足 V2V 通信对超可靠和低时延（Ultra Reliable and Low Latency Communications，URLLC）的要求[66]。CHEN J 等人提出两级自适应资源分配框架对无线资源和感知资源进行联合

分配以满足不同的 QoS 要求[67]。HE Y 等人将动态车辆环境建模为一系列相关的马尔可夫决策过程，提出基于元分层强化学习的动态资源管理框架，通过对上层主网络进行微调适应新环境，增强了学习模型的泛化能力[68]。上述研究集中在单一的通信资源分配，未涉及多种资源的联合分配及优化，仍需探索复杂多应用场景下的分配策略。

车载边缘计算（Vehicular Edge Computing，VEC）系统/移动边缘计算系统中的缓存、计算卸载策略一直是学术界的研究重点，边缘内容缓存主要缓存的是数据，解决缓存什么内容，缓存在哪以及怎么缓存的问题[45, 69–73]。比如内容分发网络将热门媒体数据缓存在距离用户较近的边缘节点，用户请求获取内容时就从边缘服务器获取而不是远程服务器，能够有效减轻回程链路上的移动流量并减少交付延迟。边缘服务缓存则主要缓存任务执行所需的服务（应用程序或依赖数据库），任务计算与服务是互相耦合的。如果没有缓存对应的服务，任务需要进行卸载或等待缓存更新。如在目标检测中，输入数据由相机、激光雷达等一种或多种传感器数据组成，任务执行需要将目标检测服务程序缓存在车辆或边缘服务器中，其他类型任务则无法在此执行计算。计算任务卸载的一般流程为计算任务数据上传、任务计算执行和任务执行结果反馈，通信、存储、计算资源的管理和优化贯穿整个计算任务卸载过程，对计算任务卸载研究的本质是对资源的管理和优化。TIAN H 等人提出基于多智能体的强化学习算法进行任务卸载和边缘缓存决策，优化车辆的服务延迟和能耗[74, 75]。而车载边缘计算中的服务缓存与计算卸载少有研究，TANG C 等人将服务缓存应用于 VEC，采用李雅普诺夫优化技术优化服务程序的响应时间，将两种贪婪启发式算法合并到基于漂移加惩罚的算法中，以帮助找到近似最优解[76]。LAN D 等人提出了一种基于雾计算的服务缓存与计算卸载架构，并设计基于深度强化学习（Deep Reinforcement Learning，DRL）算法的卸载策略，以减少任务计算延迟和能量消耗[77]。LI Z 等人提出基于吉布斯采样和深度强化学习的协同任务卸载和服务缓存替换方案来最小化车辆执行任务的总系统成本[78]。上述研究没有考虑车辆或边缘服务器中存储和计算资源的限制，车载边缘计算系统中动态任务请求的服务缓存与计算卸载还需深入研究。

车载边缘计算中的任务卸载与资源分配受到学者广泛的关注和研究，围绕计算卸载的步骤思考，首先是发现边缘服务器节点，随后是任务是否能切割，即任务只能完全卸载到一个节点上执行计算或者任务可以按比例实施部分卸载在两个节点分别计算，

然后是节点执行任务计算，最后将计算结果返回。这个过程中需要研究的问题涉及卸载什么任务，是否执行卸载，卸载在哪以及怎样卸载，不同论文研究侧重的方向不同，对部分研究条件取舍进而简化问题。由于车辆的高移动性，计算卸载往往需要根据当前的网络环境实时做出决策，联合卸载和资源分配方案的设计将会变得更加复杂，使得传统的优化方法难以适应复杂且动态的 VEC 网络，基于深度强化学习的算法由于其强大的学习和决策能力而被用来解决卸载决策此类复杂的优化问题[75, 79–85]。FAN W 等人充分利用车辆多种通信模式，设计多种方案优化调度任务、分配无线信道和计算资源，最小化车辆总任务处理时延[86, 87] 。随后针对路边单元（Road Side Unit，RSU）的负载差异设计基于双延迟深度确定性策略梯度的深度强化学习算法来减少总任务处理成本[88]。NAN Z 等人研究了带有结果反馈延迟的车载边缘计算的联合任务卸载和资源分配问题，该问题被描述为非凸混合整数非线性规划问题，并由开发的低复杂度算法求解[89]。FAN X 等人提出基于异构蜂窝车联网的 VEC 架构，车辆根据最大容忍延迟对任务分类并选择合适的通信网络，使用线性松弛的分支节点算法求解任务卸载决策和计算资源分配问题[90]。ZHANG J 等人为 VEC 卸载决策策略开发了一个两阶段博弈论模型，其中考虑了运营商的收入、能耗和延迟，然后设计神经网络模型来学习已建立的博弈论模型的预测行为，以便以更有效的方式卸载决策，随后提出了一种基于特征的迁移学习算法，用于在未见过的 VEC 环境下实现可扩展的卸载优化，能够显著提高准确性和效率[91]。LIN Z 等人提出了支持非正交多址（Non-orthogonal Multiple Access，NOMA）的多雾计算接入点协作卸载方案，通过连续凸近似的内点法和博弈论找到良好的任务分割比和用户关联进而减少能耗[92]。YIN L 等人引入了一种基于多臂老虎机理论的共享资源车辆自适应类型选择算法，将交互过程建模为涉及计算资源租赁合约的多阶段斯塔克尔伯格博弈，集成 RSU 和闲置车辆资源能够提供更好的服务[93]。此外，物理层技术相关的移动边缘计算系统也值得关注，ZHU H 等人考虑到随机的任务到达和不确定的信道条件对每个用户的任务的功耗和延迟影响很大，设计深度确定性策略梯度（Deep Deterministic Policy Gradient，DDPG）算法来学习基于去中心化深度强化学习框架的最优功率分配方案[94]。XU X 等人提出基于 NOMA 的 VEC 架构，并建立协作资源优化问题，通过博弈论、分布式深度确定性策略梯度和基于梯度的迭代方法求解联合问题[95]。WANG Z 等人提出了一种 NOMA 辅助的联合通信、感知和多

层计算框架，提高计算速率并减少功能间干扰[96]。然而现有研究忽略了协同计算的需求，即单车的计算任务需求往往由与其协作车辆的感知任务组合而成，从而解决单车遭遇遮挡问题或扩大感知范围的需求。

1.2.3 车联网协同感知

随着自动驾驶技术的成熟和商业化的加速，感知技术成为其中最关键的一环。车辆的感知系统能够感知其周围环境并做出决策，检测道路、障碍物、行人、车辆以及其他关键的交通元素，并对这些信息做出反应。自动驾驶车辆的感知系统常见的传感器类型有摄像头、激光雷达、毫米波雷达等。摄像头应用广泛、成本低廉，能够提供高分辨率图像，进而执行目标检测和分类、交通标志识别和车道线检测等任务，但容易受到强光、恶劣天气等影响，对距离和速度的测量精度低。激光雷达通过发射激光脉冲并测量其反射回来时间确定目标的距离，进而构建环境的三维地图，高精度的深度信息和全天候工作的特性被广泛应用于智能车上。毫米波雷达通过发射毫米波频率的无线电波，并接收反射回来的信号测量目标的位置、速度和角度等信息，可测距离远、抗环境干扰能力强。不同传感器组合从而提供车辆周围环境的准确可靠信息，保障车辆安全驾驶。

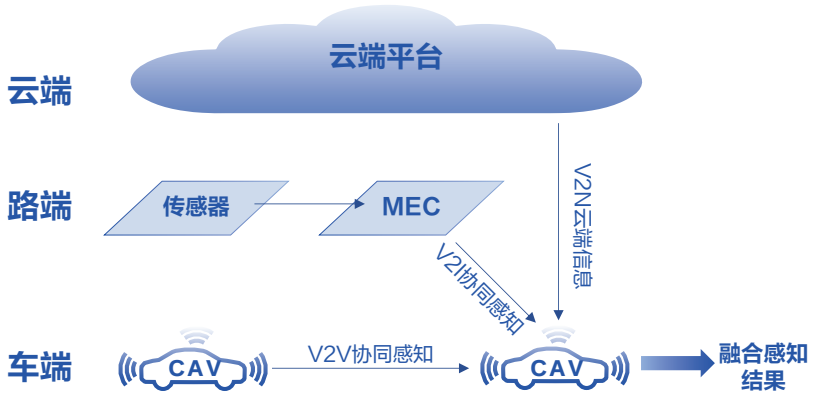


图 1-4 协同感知示意图

Figure 1-4 Diagram of cooperative perception

车联网协同感知是车路协同向协同决策控制演进的关键核心，实现自动驾驶感知互补、强化和冗余，通过发挥车端、路端、云端感知优势，与自车进行协同感知，解决超视距、盲区、遮挡等一系列感知长尾问题，协同感知示意图如图1-4所示，因此协同感知所需的基础支撑技术包括传感器技术、多传感器融合感知技术、通信技术和移动

边缘计算技术。协同感知根据融合数据类型不同可分为原始数据层级（前期融合）、特征数据层级（中期融合）、目标数据层级（后期融合）[97–101]，如图1-5所示，原始层级融合框架中直接融合协作终端共享的原始传感器数据，如图像/点云信息，其包含最全面的信息但传输数据量较大，而车联网无线通信带宽极为有限，造成较高传输延迟[102]。目标层级融合框架则对协作终端共享运行完目标检测算法得到的感知结果如目标形状、位置、朝向角等信息进行融合，通信数据量极低，依赖检测算法性能，会损失较多信息。特征层级融合框架对协作终端共享的经过特征提取算法压缩后的特征信息进行融合，是保留检测所需信息同时降低传输数据量的一种折中方案。车联网协同感知技术目前已开源仿真数据集 OPV2V[103]，V2XSet[104]，V2X-SIM[105]，DOLPHINS[106] 和实车采集数据集 V2V4Real[107]，DAIR-V2X[108]，V2X-Real[109]，上述数据集仅包含多模态感知数据，开源的 M^3SC 数据集包含了射频信道信息和多模态感知信息用于机器联觉研究[110]。车联网协同感知仿真框架目前已开源 OpenCOOD[103]，OpenCDA[111, 112]，HEAL[113] 等平台，基于数据集及仿真框架支撑协同感知、决策、控制等相关研究。

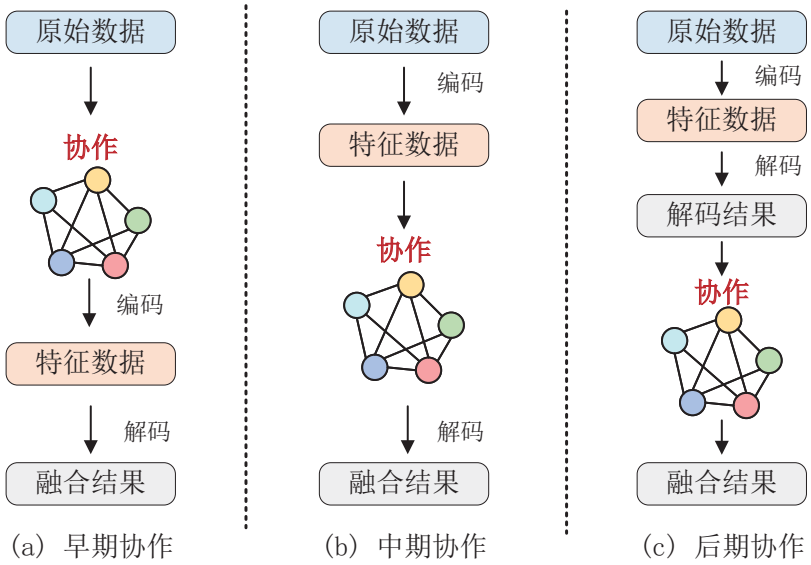


图 1-5 协同感知融合策略

Figure 1-5 Fusion strategies for collaborative perception

车联网赋能自动驾驶的协同感知技术面临无线通信带宽资源受限、随机时延、车辆高移动性导致位姿误差、传感器模态和感知模型异构、隐私及安全等多重挑战。CHEN Q 等人首先提出原始数据级协作感知系统 Cooper[114] 和基于点云数据的特征数据级协

作感知系统 F-Cooper[115]。WANG T H 等人提出 V2VNet 模型压缩协作方共享的特征数据[116]。LI Y 等人提出蒸馏协作网络 DiscoNet 取得更好的性能带宽权衡[117]。HU Y 等人提出的 Where2comm 考虑使用压缩通过动态调整通信涉及的空间区域来处理变化的通信带宽[118]。此外 When2comm[119]，What2comm[120]，How2comm[121] 均通过不同的技术路径实现感知性能和通信带宽之间的权衡。LU Y 等人提出 CoAlign 增强协作车辆之间的位姿一致性，显著降低相对定位误差[122]。基于边缘计算系统辅助的车路协同感知 EdgeCooper 系统使用多跳协作 5G V2X 通信安排车辆与边缘服务器共享互补性增强和冗余最小化的原始传感器数据，从而增强感知鲁棒性并扩大感知范围[123]。然而绝大部分工作严重依赖准确的位姿关系，未考虑实际环境下的非理想通信环境和定位误差等，且目前缺少实际在车联网硬件部署相关算法的实践案例。

1.2.4 主要问题总结

在车载边缘计算系统中，核心研究问题是计算卸载策略，其本质在于计算、存储和通信资源的精细化管理与优化。从资源供应端来看，车端、路端及云端各自拥有的计算和存储能力存在显著差异，且这些资源的可用性不是静态的，而是动态变化且不可预测的。从资源需求端分析，不同场景任务对资源的需求多样而复杂，任务执行与其计算环境紧密耦合，这种耦合性使得程序缓存和任务卸载决策相互依存，增加了决策的复杂度。因此，在多车辆单任务动态请求环境下，不合理的程序缓存和任务卸载决策会导致系统时延和能耗成本过高。基于车路云一体化系统，寻求一个有效的长期缓存与卸载联合决策方案，成为了一个紧迫且有价值的研究课题。

自动驾驶感知任务是程序关联性任务的典型代表，要求高计算和存储资源，同时对时延容忍度严格。由于车辆物理环境随时间变化，车联网的网络拓扑结构高度动态，可能造成车辆和边缘计算服务器之间的通信不稳定，降低任务卸载的可行性。网联自动驾驶中的协同感知任务旨在克服单车感知的局限性，如感知范围限制、盲区和遮挡等问题。然而，不同车辆的感知模型和计算能力存在差异，当前研究未充分考虑不同级别的数据协同融合和感知任务的结果反馈。此外，协同感知场景中的任务往往需要多个节点的感知任务组合以确保主车的自动驾驶安全。因此，如何在车载边缘计算系统的支持下，通过更精细化的资源管理和有效的任务卸载决策，确保自动驾驶的安全性，是一个亟需解决的问题。

协同任务的典型应用是协同感知。车联网协同感知任务依赖准确的位姿关系，但在非理想通信环境和存在定位误差的情况下，定位不准确可能导致姿态估计错误，从而显著降低协作性能。因此，深入研究网联自动驾驶中的协同位姿优化对于提升协作效果至关重要。为推动车载边缘计算系统实现车联网协同感知应用走向落地，实现协同感知原型系统对进一步开展实际性能测试具有一定的实用价值和意义，现阶段还缺乏实际原型系统以及相关的性能评估，基于已商用的车载通信单元和计算单元开发潜在落地的目标级协同感知系统是十分必要的，这将助推自动驾驶协同感知研究向实际应用迈进。

车载边缘计算系统面临的主要问题包括资源供应端的动态性和不确定性、任务执行环境的复杂性以及决策复杂度的提升。特别是在网联自动驾驶中，感知任务的高资源需求和严格时延要求增加了卸载决策的难度。此外，准确的位姿优化和协同感知原型系统的开发与性能评估也是亟需解决的关键问题。通过精细化的资源管理和优化决策，解决这些问题将显著提升自动驾驶系统的性能和安全性，促进其向实际应用的转化。

1.3 研究目标和主要研究内容

1.3.1 研究目标

本文面向车载边缘计算系统中多节点多类型协同任务需要实时高效完成计算的需求，应对有限通信、计算、存储资源条件下的供应不确定性、动态变化性，面向多场景任务需求多样性展开研究，旨在提升车载边缘计算系统中多类型任务场景资源分配合理性和自适应性，推动网联自动驾驶协同感知应用走向落地。本文的研究内容框图如图1-6所示。

1.3.2 主要研究内容

本文致力于车载边缘计算系统中任务计算环境的资源配置优化和协同任务应用研究，各研究内容的主要关系描述如下：任务计算环境的资源配置优化研究包括面向程序关联性任务的资源配置优化和协同感知任务资源配置优化，前者基于车路云一体化系统整体架构配置单车任务计算环境，为研究内容二探索协同任务计算环境下的资源配置优化提供技术支撑，两者共同保障协同任务应用研究的计算环境。研究内容三面向网联自动驾驶的协同位姿优化及应用聚焦在车载边缘计算中的系统业务，是协同感

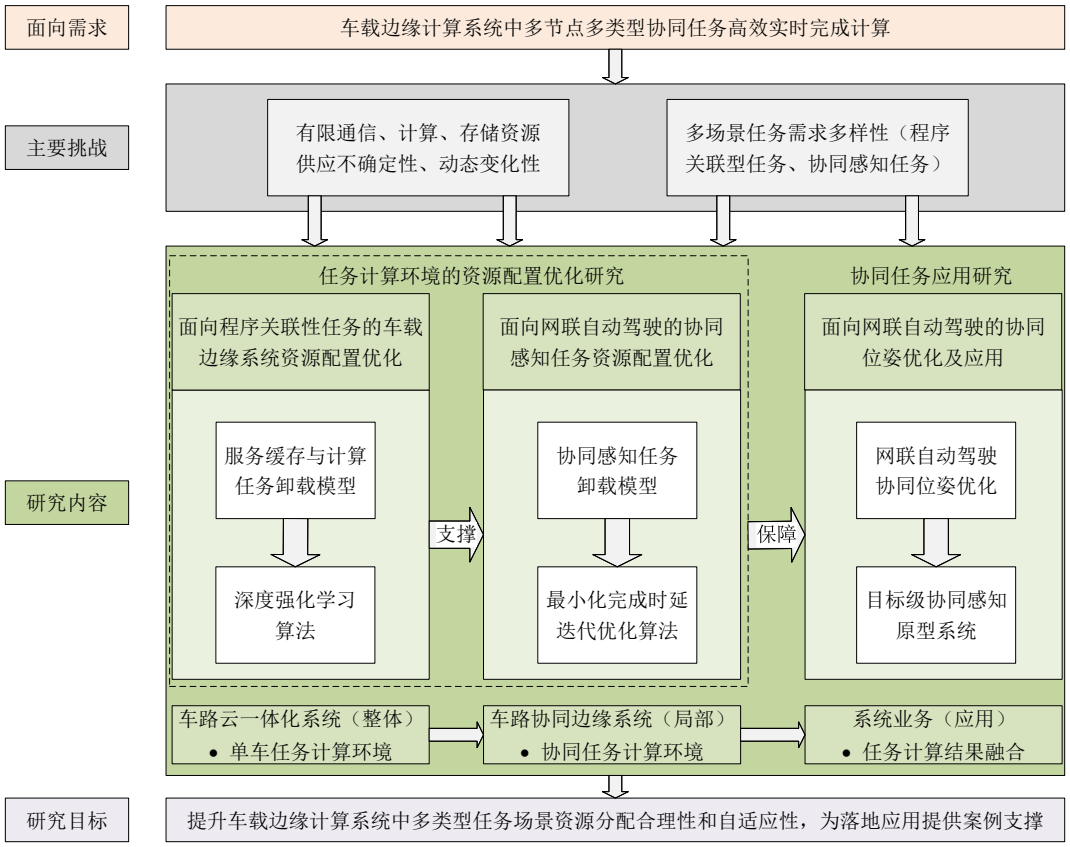


图 1-6 本文的研究内容框图

Figure 1-6 Block diagram of the research in this thesis

知任务执行计算后的结果融合，提升协同感知的准确性，设计并实现目标级协同感知原型系统进行融合算法验证。研究内容一和二分别从整体和局部层面对资源配置进行优化，共同保障协同任务应用研究的计算环境。研究内容三依赖于前两个研究内容提供的优化计算环境，进行更高层次的协同感知结果融合和优化。各研究内容既独立成章，又相互支撑，共同推进车载边缘计算系统中资源配置与协同任务应用的协同发展。通过这些研究内容的有机结合，本文将为车载边缘计算系统中的任务计算环境优化和协同感知任务的高效执行提供全面的理论基础和实践指导。

(1) 面向程序关联性任务的车载边缘计算系统资源配置优化

针对车边云系统架构，面向车载边缘计算系统网联汽车动态程序关联性任务计算请求，提出服务缓存与计算卸载框架，任务计算完成时延及边缘节点能量消耗分析推导，建立任务完成时延优化模型，提出基于确定性策略梯度算法执行缓存和卸载决策，保障动态时变联网环境下长期缓存与卸载决策最优，系统任务完成及时可靠。具体而言，考虑存储和计算资源受限情况下，车载计算任务需要缓存对应的服务程序才能执

行计算，缓存决策与卸载决策相互关联，任务请求动态变化，基于车辆移动模型、通信模型、服务缓存与卸载模型对任务计算时延和边缘节点能量消耗进行分析推导。由于缓存决策与卸载决策相互关联，对不同缓存情形任务完成时延和能耗极值分析推导，建立长期任务平均完成时延最小化问题，该问题可以表示为马尔可夫决策过程，使用深度强化学习方法即深度确定性策略梯度算法求解，最后进行性能仿真、结果分析。

(2) 面向网联自动驾驶的协同感知任务资源配置优化

面向协同感知任务计算场景，建立自动驾驶车辆的感兴趣域，感兴趣域由不同车辆的感知任务计算结果组合而成，对于算力不够、无法保证低时延完成的任务可选择V2V/V2I 卸载到服务域的节点完成计算，建立计算、卸载、传输模型，优化目标旨在最小化感兴趣域任务处理完成时延。此问题为混合整数非线性规划问题，对问题的可行域进行分析。求解方法先对其解耦，固定离散变量，采用非线性优化求解器对资源分配变量优化，遍历离散变量，得到最优解。对算法，场景多次仿真并进行结果分析。

(3) 面向网联自动驾驶的协同位姿优化及应用

面向协同感知任务计算场景，感知信息共享存在不准确的定位信息，导致位姿估计错误进而降低协作性能，为减轻该影响并充分利用协作的优势，建立关于车-车、车-特征、车-动态目标以及高精地图、全球导航卫星定位系统信息的因子图约束模型，提出针对自车及感知目标的状态估计算法进行位姿优化，实现自定位及感知目标定位增强，提升协同感知准确性。为了验证协同感知算法设计，对不同协同感知融合策略进行性能评估，并开发了目标级协同感知系统原型，基于 C-V2X 设备和自动驾驶边缘计算平台部署检测模型和融合算法，并评估了协同链路各功能模块时延，验证了实际部署系统的可行性。

1.4 论文组织结构

本文围绕车载边缘计算系统资源分配和协同感知相关问题展开了研究。具体地，本文面向多节点多类型协同任务高效及时完成计算的需求，针对有限通信、计算、存储资源供应不确定性、动态变化性以及多场景任务需求多样性等挑战，重点研究了面向程序关联性任务和网联自动驾驶协同感知任务的调度和资源分配问题，并对卸载结果的性能进行优化，以及系统原型实现方面进行了理论研究和技术创新。本文共分为六个章节，详细内容安排如下：

第一章，绪论。首先介绍了车联网的研究背景及意义，其中包括顶层政策、标准演进以及国内外研究团队介绍。然后对车载边缘计算服务架构、资源分配和协同感知的研究现状进行深入调研，分析并总结出目前存在的主要问题。最后，详细阐述了本文的研究目标、主要研究内容以及论文组织结构。

第二章，车载边缘计算系统关键技术。介绍本文研究内容涉及的关键技术，包括车载边缘计算系统架构、计算卸载、深度强化学习理论和协同感知技术。

第三章，面向程序关联性任务的车载边缘计算系统资源配置优化。首先，提出程序关联性任务的服务缓存与计算卸载场景，建立程序任务缓存与卸载传输模型，对不同缓存与卸载决策情形推导分析形式化定义了最小化长期任务完成计算时延问题。其次，提出基于深度强化学习的缓存与卸载决策方案，最后，构建实验仿真模型并验证了相关指标和算法的优越性。

第四章，面向网联自动驾驶的协同感知任务资源配置优化。首先，提出了协作感知任务感兴趣域的需求与服务场景，建立协作感知任务阶段选择与卸载决策传输模型，在此基础上形式化定义了最小化任务计算时延问题，随后提出了最优卸载决策与资源分配算法，建立实验仿真模型验证所提算法的优越性。

第五章，面向网联自动驾驶的协同位姿优化及应用。首先，建立车与环境目标之间的因子图，对约束关系建立状态描述的数学模型，建立最小化残差的状态估计问题。构建实验仿真模型，验证提出的估计求解算法有效性。随后评估不同融合策略下协同感知的仿真性能，搭建基于 C-V2X 和边缘计算平台的硬件在环试验平台，部署目标级协同感知算法，并对系统链路时延进行分析和评估，验证目标级协同感知的技术可行性与有效性。

第六章，总结与展望。对本文的研究工作归纳总结，对未来可能的研究方向进行展望。

第二章　车载边缘计算系统关键技术

2.1 引言

车联网技术在加快自动驾驶实现、降低单车智能成本、提升道路交通安全等方面具有优越性，且 C-V2X 已得到中国和美国等主要汽车与交通大国认可，成为全球事实车联网通信标准。我国率先提出依托 C-V2X 的“聪明的车 + 智慧的路 + 融合的云”的车路云一体化协同发展路线，支撑交通产业、汽车产业和智慧城市变革。车路云一体化系统基于路侧感知、边缘计算、云端信息融合以及 C-V2X 和 4G/5G 通信技术，实现通信技术，实现“车-路-云”之间的全方位协同配合（如协同感知、协同决策规划、协同控制等），从而满足不同等级自动驾驶车辆行驶安全、高效、节能与舒适需求的车路云一体化系统，以达到自动驾驶车辆性能和交通全局最优化发展目标[124]。随着智能网联汽车渗透率不断提高，由此产生的车载信息娱乐、传感器、车路协同、地图等数据传输、处理、存储的需求极大地增加了网络负荷，并对网络时延、带宽、可靠性提出了更高的需求。将移动边缘计算技术应用到车联网之后，将业务通过任务卸载到边缘节点的服务器，减少数据传输路由长度，从而降低端到端通信时延；边缘服务器作为本地托管环境，提供弹性的计算、存储资源，能够提供更具地理和区域特色、更高吞吐量的车联网服务。作为未来车联网场景中的应用需求，本文研究了车载边缘计算系统中的资源分配和协同感知技术。本章主要通过对车路云一体化系统、计算任务卸载、深度强化学习、自动驾驶协同感知等关键技术进行总结和梳理，为本论文后续研究工作奠定基础。

本章内容安排如下：2.1 节是本章的引言，对车联网及其赋能相关技术简介。2.2 节阐述了车载边缘计算系统及计算卸载技术。2.3 节介绍深度强化学习理论。2.4 节介绍自动驾驶协同感知技术。2.5 节总结本章内容。

2.2 车载边缘计算系统

2.2.1 系统架构

车路云一体化系统是通过新一代信息与通信技术将人、车、路、云的物理空间、信息空间融合为一体，基于系统协同感知、决策与控制，实现智能网联汽车交通系统安全、节能、舒适及高效运行的信息物理系统（Cyber Physical Systems，CPS）[9]。车路云

一体化系统（Vehicle-Road-Cloud Integrated System，VRCIS）由车辆及其他交通参与者、路侧基础设施、云控平台、相关支撑平台、通信网等部分组成的一个复杂大系统，如图2-1。智能网联汽车是系统的关键组成，是产生交通动态数据的来源，也是路侧基础设施和云控平台的服务对象。智能网联汽车的感知和决策信息通过 C-V2X 通信，上传到路侧和云端，同时也接收路侧和云端下发的感知信息、决策规划建议和控制指令。智能化路侧基础设施是增强智能网联汽车感知的有效补充手段。路侧基础设施主要包括路侧通信单元、路侧计算单元、路侧感知设备、交通管理设施等，以实现车路云互联互通、环境感知、局部辅助定位等功能。云控基础平台由边缘云、区域云与中心云三级基础平台组成，形成逻辑协同、物理分散的云计算中心。云控应用平台则基于基础平台之上，提供有效整合的车路云感知、决策、控制信息，实现驾驶安全和效率的综合提升[125]。

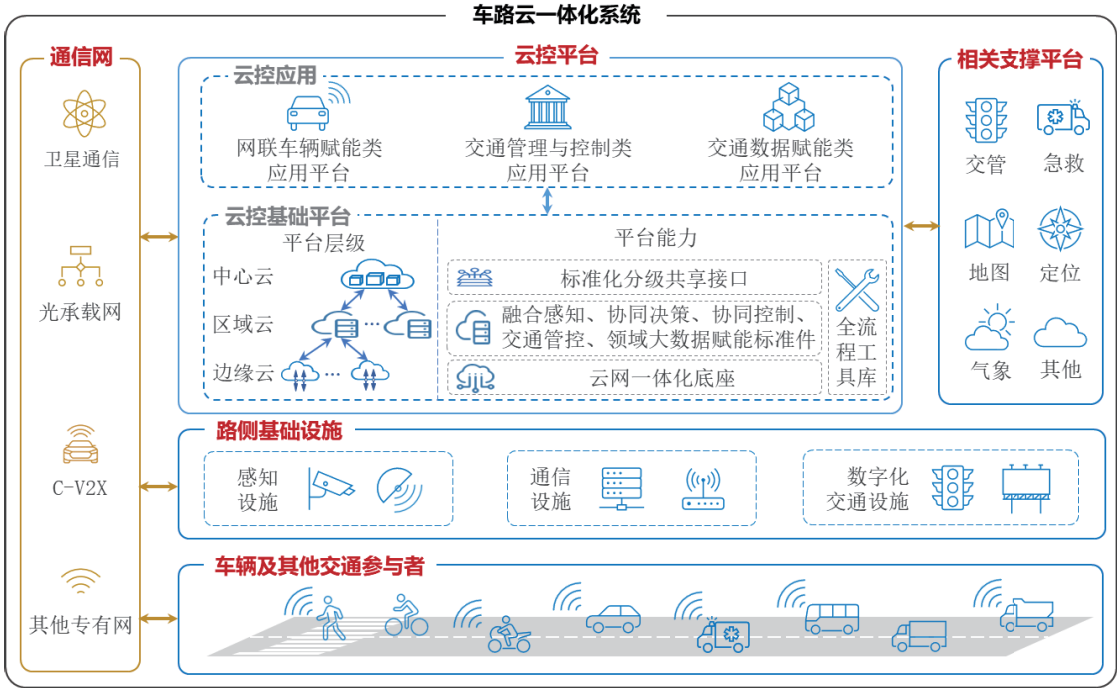


图 2-1 车路云一体化系统架构[9]

Figure 2-1 Architecture of Vehicle-Road-Cloud integrated system

云控基础平台的信息基础设施建设中，5G+MEC+C-V2X 构成的车载边缘计算系统是其重要的组成部分，为车路协同平台运行提供关键支撑，全面赋能智能网联应用实现本地化的数据快速上传、边缘存储计算、决策结果下发执行的闭环。车载边缘计算的融合通信网络架构如图2-2所示，包括“端-管-云”三层结构，实现环境感知、融合

计算、决策控制，从而提高安全、高效、便捷的智慧交通服务，端在广义上包括 OBU、RSU、具有感知功能的摄像头、雷达等，以及红绿灯、公告牌、电子站牌等路侧交通设备。RSU 是道路信息的汇集点，将收集到的道路交通信息通过 5G Uu 发送给云平台或通过 C-V2X PC5 发送给 OBU 终端，另外可以通过 5G 接收云平台下发的业务数据，再通过 C-V2X PC5 广播给附近的 OBU 终端。OBU 是用户获取车联网业务的入口，车联网交通参与者通过 5G 网络接入至云平台层，获取多样的车联网应用服务，同时通过 C-V2X PC5 获取近距离范围内其他 OBU 终端和 RSU 终端发送的实时消息。管指交通各实体互联互通的网络，5G 网络具有广覆盖、大带宽的特点，主要用于承载 RSU/OBU 至云端的业务。C-V2X PC5 网络则负责本地小范围内低时延通信，如车联网安全消息类业务。云指实现数据汇集、计算、分析、决策以及运维管理、性能监控功能的平台，根据业务时延需求可部署在边缘侧或中心。总体系统架构分为三层，分别为中心节点、边缘计算节点、路侧边缘计算单元。中心节点实现非实时业务的数据汇集、监管、宏观决策以及运维管理，汇聚广域范围的车辆、路侧、交通控制平台等交通单元的相关信息，实现各类信息数据的存储和分析，以及非实时信息的下发。

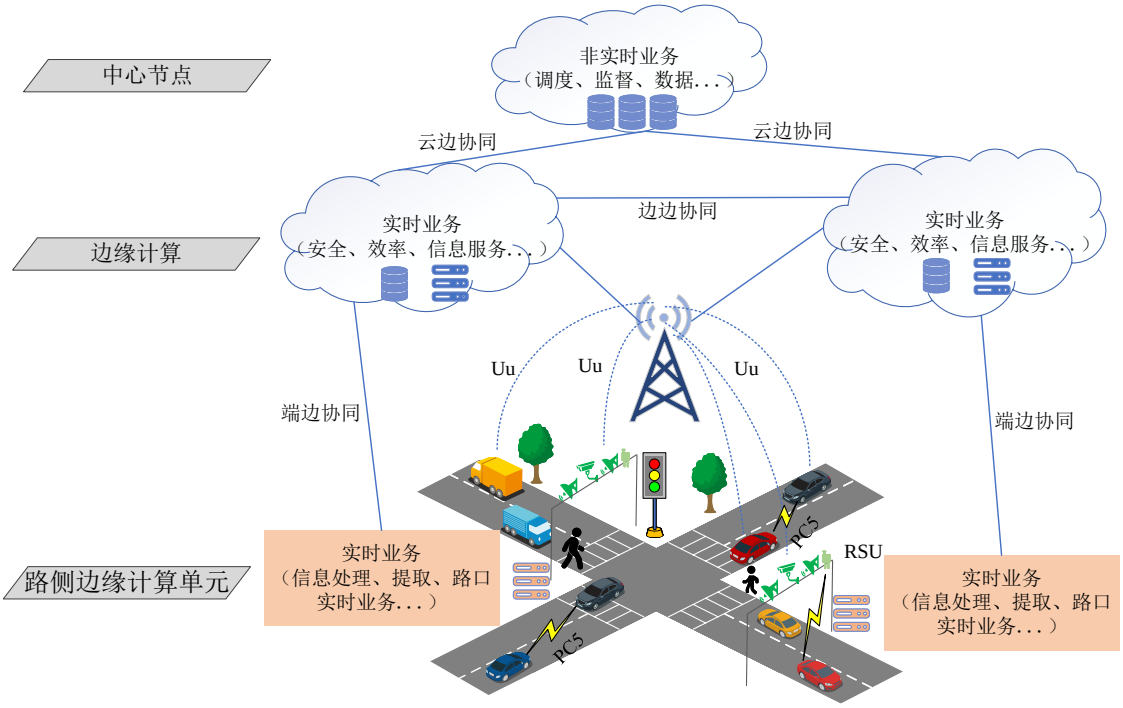


图 2-2 5G+C-V2X+MEC 融合通信网络架构

Figure 2-2 Integrated communication network architecture of 5G, C-V2X, and MEC

MEC 以边缘网络及计算资源为基础，满足路侧信息融合分析、安全提示、高精度

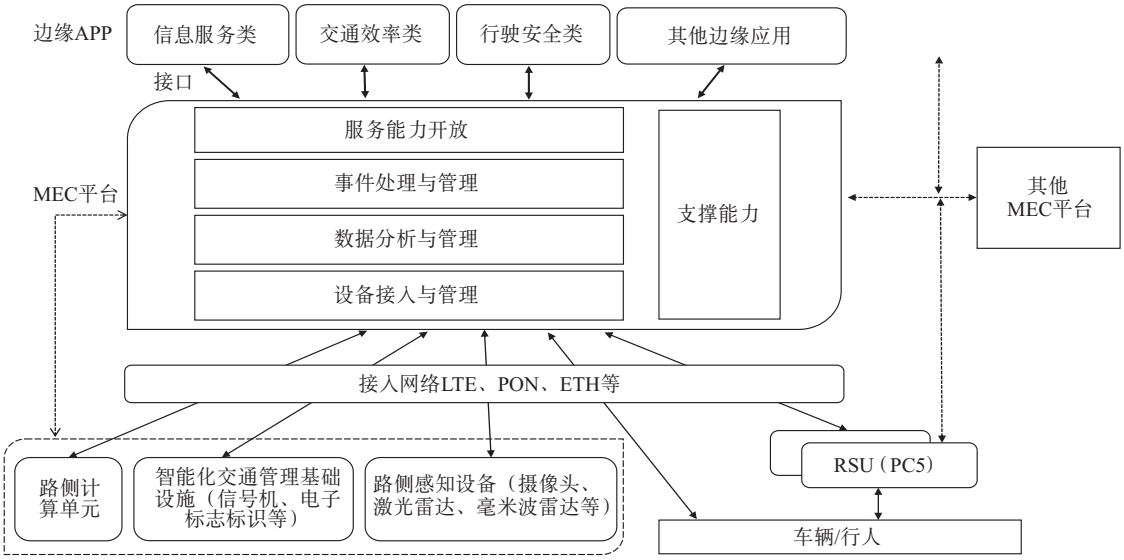


图 2-3 车联网 MEC 业务总体架构[126]

Figure 2-3 Overall architecture of vehicular edge computing services

定位等场景能力部署，提供低时延、大带宽、高算力的运行环境。车联网 MEC 业务总体架构如图2-3所示，架构最底层是终端用户设备包括网联车辆、行人等能够接入网络的多类型交通参与者，智能网联汽车配备有各类感知周围环境的传感器、车载通信单元以及配套的计算单元，具有一定的计算资源，但算力有限。对于具有时延紧迫性的应用和大规模的数据信息，需要以卸载的方式将任务输入数据全部或部分地交给边缘计算平台或云平台，也可以通过 V2V 方式卸载到其他有闲置计算资源的智能网联汽车。路侧 MEC 平台包含了智能化交通管理基础设施和智能化路侧感知设备，智能化交通管理基础设施包括信号机、电子标志标识等。智能化路侧感知设备主要是摄像头、毫米波雷达和激光雷达三类常用的传感器以及专用管控类传感器。摄像头、雷达分别用于采集交通环境中的图像、视频和点云等原始数据，随后通过接入网传输到 MEC 平台，经过时空同步、多模态数据融合计算处理后，产生实时性较强、范围较广的结构化感知数据，包含各类交通参与者状态信息等。路侧 MEC 平台也能够根据交通参与者需求提供额外的感知信息，发挥其上帝视角优势。MEC 平台的部署位置根据应用完成时延、请求数量、数据传输的安全性和当前网络环境等多因素动态确定，可以选择部署在 RSU 后，或部署在基站节点后，或部署在其他的合理位置。网络中 MEC 平台可实施分层多级部署，下级 MEC 平台可作为上级 MEC 平台的接入端，多层级 MEC 平台上下级之间的网络连接方式不做限制，可以根据具体车联网应用场景的多方面需求实现在终端和

云端之间的多层级灵活部署。MEC 平台分为三层，包括虚拟化基础设施、微服务组件、接入管理、应用管理以及 V2X Server，其中 V2X Server 提供统一的 V2X 综合服务[127]。此外提供面向边缘应用如信息服务类、安全驾驶类、驾驶效率类等应用程序接口。

MEC 在网络边缘提供 IT 基础资源及虚拟化应用托管环境，多种车联网应用（信息服务类、安全驾驶类、驾驶效率类和其他边缘应用）可以部署在其中。MEC 可为车联网提供按需的资源规划和编排，同时也可为车联网业务提供计算和存储能力，包括数据缓存、预处理、流分析、函数计算、分布式人工智能及推理等能力。可提供感知数据处理能力，还可提供高精地图的存储和分析计算能力，并进一步为车联网应用的决策算法提供计算能力。MEC 将车、路、云等多源数据进行分析，转化为可理解的信息，进一步通过判断与权衡来形成最优决策，对车辆下发控制建议或直接进行控制。车载边缘计算系统中的边缘缓存、计算任务卸载以及感知数据处理等是其中的关键技术，研究其中的资源分配和协同感知技术对满足资源密集型车联网应用具有重要意义。

2.2.2 计算卸载技术

MEC 中的计算任务卸载是一种将计算任务从本地设备转移到远程服务器或云基础设施的过程。该技术旨在通过利用远程计算资源来优化资源利用、降低能耗并提高性能，其流程可分为可卸载节点感知、计算任务划分、卸载决策制定、任务上传、服务器执行计算、计算结果返回等六部分[128]，如图2-4所示。

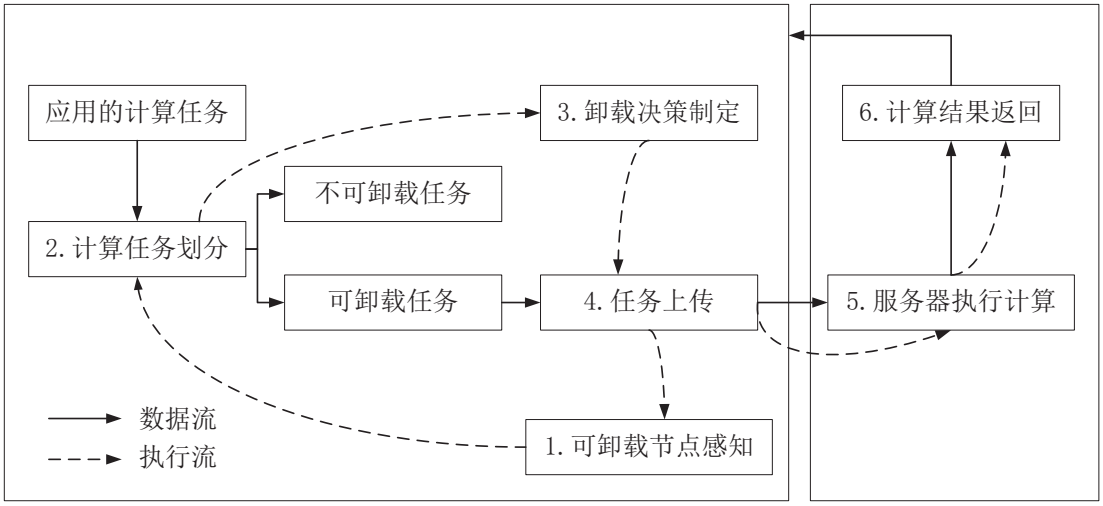


图 2-4 计算卸载流程

Figure 2-4 The process of computation offloading

（1）可卸载节点感知

车辆产生了计算任务是，需要识别出网络中可用的卸载节点。可卸载节点可能是部署在 RSU 的 MEC 服务器或云端的数据中心或者具备空闲计算资源的车辆。在这一步骤中，会收集有关这些节点的信息，包括它们的计算能力、存储容量、网络带宽、延迟以及费用等。只有找到可卸载节点，才执行后续步骤。

（2）计算任务划分

车辆确定卸载节点后，在这一阶段，需要确定哪些计算任务可以卸载，任务通常被划分为可拆卸型任务和不可拆卸型任务，这通常基于任务的特性和需求[129]。可拆卸型任务可将任务进行分割，车辆可以选择一部分任务在本地执行计算，一部分卸载到边缘节点执行计算，本地计算的任务与卸载的任务比重取决于制定的卸载策略。不可拆卸型任务即任务不可分割，任务只能全部在本地计算或全部上传至边缘节点进行计算。

（3）卸载决策制定

在这个环节，系统将基于可卸载节点的感知信息和计算任务的划分结果来做出卸载决策。对于可拆卸型任务，决定卸载到边缘服务器或协作车辆以及卸载任务的比例，此类卸载模式称为部分卸载。对于不可拆卸型任务，决定是在本地或远程服务器或协作车辆，此类卸载模式称为完全卸载，完全卸载由于车辆全程没有参与计算，一定程度上减少了能量消耗，MEC 由于计算资源丰富，缩短原本在本地进行计算产生的时延。决策制定通常涉及到多目标优化问题，需要权衡计算效率、能耗、成本和延迟等多个因素。卸载算法可能包括凸优化算法、启发式方法、博弈论、深度强化学习等。卸载策略制定是流程中关键的一环。

（4）任务上传

任务上传基于第三步的卸载决策，如若决定的是全部任务在本地计算则无需上传，反之则需要根据卸载决策上传至相应节点。任务上传过程中需要确保数据的安全传输，同时根据网络状况（如带宽和延迟）调整上传速率。

（5）服务器执行计算

MEC 接收到卸载的任务和数据后，会先对任务的计算量进行判断，若超出计算能力范围，如果有其他边缘服务器空闲，则可以进一步卸载或上传至核心网的中心云服务器执行计算。MEC 服务器依靠 NFV 技术，产生与每个任务对应的虚拟环境区执行

相应的计算。服务器执行计算可能涉及到大规模数据处理、复杂的数学模型运算等。

（6）计算结果返回

计算完成后，服务器需要将结果返回给请求任务计算车辆。服务器完成对数据的提取和计算后的结果，其数据量比原本任务的输入数据量小很多，其传输过程中产生的时延和能耗通常忽略不计[75]。计算结果返回标志着一次计算任务卸载的完成。

基于计算卸载流程，对计算任务卸载的时延进行分析，如果任务是卸载给 MEC 服务器，则包括计算任务通过 V2I 的通信方式上传至 RSU 的传输时延、RSU 传输给 MEC 服务器的时延、MEC 服务器执行计算的时延，计算结果返回给请求任务计算车辆的时延，其中 MEC 服务器与 RSU 之间通过光纤连接，时延忽略不计。如果是卸载给提供计算服务的协作车辆，则包括计算任务通过 V2V 的通信方式上传给协作车辆的传输时延，协作车辆执行计算任务的时延，再将结果返回的传输时延。

2.3 深度强化学习

2.3.1 强化学习

强化学习（Reinforcement Learning，RL）是研究智能体在复杂、不确定的环境中学习的一种机制，其核心是不断调整策略以最大化获得的奖励。强化学习中智能体和环境是两个关键组成部分，智能体是执行决策的主体，影响智能体决策的因素统称为环境。在强化学习过程中，智能体与环境一直在交互，智能体在环境里获取某个状态后，会根据该状态进行决策并输出一个动作。随后环境根据智能体执行的动作做出一些反馈，输出下一个状态以及当前这个动作带来的奖励，奖励是环境提供的反馈信号，指示智能体采取特定策略的效果，智能体通过不断调整策略来最大化自身利益。智能体通过与环境的不断交互，积累经验，学习最优策略，实现收益最大化。强化学习的目的就是最大化智能体可以获得的期望累积奖励，强化学习示意图如图2-5所示。

强化学习与监督学习的区别在于，监督学习的输入数据是没有关联的以及提供给学习器正确的标签，通常假设样本空间中的全部样本服从一个未知分布，每个样本独立地从这个分布采样获得即独立同分布。而强化学习不满足这两个条件，强化学习输入的样本是序列数据，学习器并没有告诉每一步正确的动作应该是什么，学习器需要自己发现哪些动作可以带来最多的奖励，通过不断尝试来发现最有利的动作。智能体获得能力的过程是通过不断试错探索来实现的，探索和利用是强化学习里非常核心的

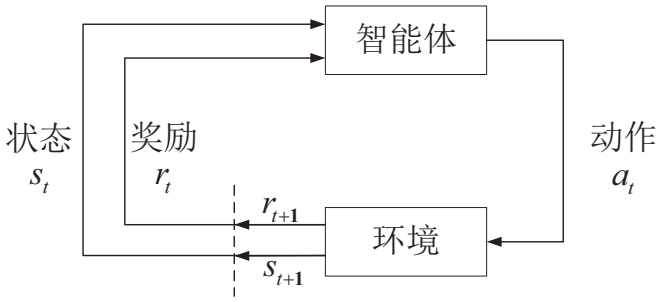


图 2-5 强化学习示意图

Figure 2-5 Illustration of reinforcement learning

问题[130]。探索即尝试新的动作，可能获得更多的奖励也可能相反。利用则可以采取已知的可以获得最多奖励的动作，重复执行该动作，因为该动作可以获得一定的奖励。因此需要在探索和利用之间进行权衡，这是监督学习里没有的情况。在强化学习的过程中，没有非常强的监督者，只有奖励信号，奖励信号是延迟的，即环境在很久以后告诉我们之前采取的动作到底是不是有效的。因为不是即时的反馈，智能体学习起来非常困难，如果采取了一个错误的动作，环境可能会告诉智能体采取了错误的动作，但并没有告诉正确的动作是什么，并且是在一段时候后告诉智能体动作是错误的。因此总结出来强化学习的特征：强化学习会试错探索，通过探索获取对环境的理解；从环境中获得延迟的奖励；训练过程中，时间非常重要，需要有时间关联的数据；智能体的动作会影响随后得到的数据，一个重要的问题是如何让智能体的动作一直稳定地提升。

在强化学习环境里，智能体目的是选取一系列动作来最大化奖励，而这些动作必须具有长期的影响力。强化学习是基于序列化学习的过程，其数学基础和建模工具是马尔可夫决策过程（Markov Decision Process，MDP），一个 MDP 通常由状态空间、动作空间、状态转移函数、奖励函数、折扣因子等组成。状态空间指所有可能存在状态的集合，状态空间可以是离散的或连续的、有限的或无限的。动作空间即所有可能动作的集合，可以是离散的或连续的、有限的或无限的。状态转移是智能体从当前 t 时刻的状态 s 转移到下一个时刻状态为 s' 的过程，状态转移通常是随机的，随机性来源于环境，常用状态转移函数来描述状态转移，记作

$$p_t(s'|s,a) = \mathbb{P}(S'_{t+1} = s'|S_t = s, A_t = a) \tag{2.1}$$

表示这个事件的概率：在当前状态 s，智能体执行动作 a，环境的状态变成 s'。

2.3.2 基于价值迭代的深度强化学习

强化学习智能体的重要组成成分是策略和价值函数, 策略是智能体的动作模型, 即根据观测到的状态，如何做出决策，即如何从动作空间选取一个动作。价值函数是对当前状态做出评估，价值函数越大，智能体进入该状态越有利。模型表示智能体对环境的状态进行理解，决定环境中世界的运行方式。强化学习的目标就是得到一个策略函数，在每个时刻根据观测到的状态做出决策，分为随机性策略和确定性策略。随机性策略是 π 函数，即

$$\pi(a|s) = \mathbb{P}(A = a|S = s) \tag{2.2}$$

策略函数的输入是状态 s 和动作 a，输出是一个 0 到 1 之间的概率值。这个概率是智能体所有动作的概率，然后对这个概率分布进行采样，得到智能体将采取的动作。确定性策略是状态 s 作为输入，直接输出动作 $a = \mu(s)$，而不是输出概率值。对于给定的状态 s，做出的决策 a 是确定的，没有随机性。

价值函数是对未来奖励的预测，评估状态的好坏，累积奖励指当前时刻开始到本回合结束的所有奖励的总和，也叫回报，将 t 时刻的回报记作随机变量 U_t。如果一回合结束，已经观测到所有奖励，回报记作 u_t[131]。本回合在时刻 n 结束，则回报定义为

$$U_t = R_t + R_{t+1} + R_{t+2} + R_{t+3} + ... + R_n \tag{2.3}$$

回报是未来获得的奖励总和，强化学习的目标是寻找一个策略，使得回报的期望最大化，而不是最大化当前的奖励。由于奖励 r_t 与 r_{t+1} 所处时刻不同，存在不确定性，未来的奖励需要执行折扣计算，折扣回报定义为

$$U_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \gamma^3 R_{t+3} + ... \tag{2.4}$$

其中 $\gamma \in [0, 1]$ 为折扣率。越久远的未来，其折扣奖励越大。

智能体在 t 时刻观测到 s_t 及其之前的状态、动作、奖励为

$$s_1, a_1, r_1, s_2, a_2, r_2, \cdots, s_{t-1}, a_{t-1}, r_{t-1}, s_t \tag{2.5}$$

这些都是实际观测到的数值，一旦被观测便是确定的，而 s_t 之后未被观测到的随机变

量如下

$$A_t, R_t, S_{t+1}, R_{t+1}, \cdots, S_n, A_n, R_n \tag{2.6}$$

奖励 R_t 依赖于已观测到的状态 s_t 与动作 A_t，奖励 R_{t+1} 依赖于 S_{t+1} 和 A_{t+1}，以此类推，U_t 的随机性来源于这些动作和状态：

$$A_t, S_{t+1}, R_{t+1}, \cdots, S_n, A_n, R_n \tag{2.7}$$

动作的随机性来源于策略，状态的随机性来源于状态转移函数。

回报 U_t 是未来所有奖励加权和，是一个随机变量，对其求期望进而消除其随机性。假设已经观测到状态 s_t 并执行动作 a_t。U_t 的随机性来自于 $t+1$ 时刻起所有的状态和动作：

$$S_{t+1}, A_{t+1}, S_{t+2}, A_{t+2}, \cdots, S_n, A_n \tag{2.8}$$

U_t 关于变量 $S_{t+1}, A_{t+1}, S_{t+2}, A_{t+2}, \cdots, S_n, A_n$ 求条件期望得到得到动作价值函数 $Q_\pi(s_t, a_t)$ 如下

$$Q_\pi(s_t, a_t) = \mathbb{E}_{S_{t+1}, A_{t+1}, \cdots, S_n, A_n}[U_t | S_t = s_t, A_t = a_t] \tag{2.9}$$

动作价值函数依赖于 s_t 与 a_t，而不依赖于 $t+1$ 时刻及其之后的状态和动作。由于动作 $A_{t+1}, \cdots, A_n$ 的概率质量函数是 π，使用不同的 π 求期望的结果不同。

如果排除策略 π 的影响，则为最优动作价值函数即选择最好的策略函数，如下：

$$Q_*(s_t, a_t) = \max_\pi Q_\pi(s_t, a_t), \forall s_t \in \mathcal{S}, a_t \in \mathcal{A} \tag{2.10}$$

状态价值函数则只依赖于策略 π 和当前状态 s_t，如下：

$$V_\pi(s_t) = \mathbb{E}_{A_t \sim \pi(\cdot|s_t)}[Q_\pi(s_t, A_t)] = \sum_{a \in \mathcal{A}} \pi(a|s_t) \cdot Q_\pi(s_t, a) \tag{2.11}$$

公式对随机变量 A_t 求期望，得到状态价值函数 $V_\pi(s_t)$，其也是回报 U_t 的期望：

$$V_\pi(s_t) = \mathbb{E}_{A_t, S_{t+1}, A_{t+1}, \cdots, S_n, A_n}[U_t | S_t = s_t] \tag{2.12}$$

状态价值越大，意味着回报的期望越大，用状态价值函数衡量策略 π 与状态 s_t 的好坏。

强化学习基础的算法有 Q 学习算法[132] 和 SARSA 算法[133]，其目的分别是学到最

优动作价值函数和动作价值函数。这类算法适用的状态和动作空间是有限的，且通过列表存储近似函数值，不断更新，最终收敛，学习过程效率低下，现实中的强化学习任务面临的状态空间是连续的，存在无穷多个状态，于是通过深度 Q 网络（Deep Q-network，DQN）来对价值函数进行近似[134]。训练 DQN 使用的是时间差分（Temporal Difference，TD）算法，由回报的定义

$$U_t = R_t + \gamma U_{t+1} = R_t + \gamma \sum_{k=k+1}^{n} \gamma^{k-t-1} R_k \tag{2.13}$$

将上述公式(2.13)带入动作价值函数，可以进一步表示为

$$\begin{aligned} Q_\pi(s_t, a_t) &= \mathbb{E}[U_t | s_t, a_t] && (2.14)\\ &= \mathbb{E}[R_t + \gamma U_{t+1} | s_t, a_t] && (2.15)\\ &= \mathbb{E}[R_t | s_t, a_t] + \gamma \mathbb{E}[U_{t+1} | s_t, a_t] && (2.16)\\ &= \mathbb{E}[R_t | s_t, a_t] + \gamma \mathbb{E}[Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t] && (2.17) \end{aligned}$$

此公式即为贝尔曼方程，右边是期望，对其做蒙特卡洛近似可得

$$Q_\pi(s_t, a_t) \approx r_t + \gamma Q_\pi(s_{t+1}, a_{t+1}) \tag{2.18}$$

使用神经网络来近似动作价值函数：

$$Q(s_t, a_t; \boldsymbol{\omega}) \approx Q_\pi(s_t, a_t) \tag{2.19}$$

则公式(2.18)变为

$$Q(s_t, a_t; \boldsymbol{\omega}) \approx r_t + \gamma Q_\pi(s_{t+1}, a_{t+1}; \boldsymbol{\omega}) \tag{2.20}$$

公式(2.20)左边为 $\hat{q}_t \triangleq Q(s_t, a_t; \boldsymbol{\omega})$，是神经网络在 t 时刻做出的预测。右边记为 TD 目标 $\hat{y}_t = r_t + \gamma Q_\pi(s_{t+1}, a_{t+1}; \boldsymbol{\omega})$，是神经网络在 $t+1$ 时刻做出的预测，TD 目标中前一部分基于真实观测奖励 r_t。定义损失函数：

$$L(\boldsymbol{\omega}) = \frac{1}{2}[Q(s_t, a_t; \boldsymbol{\omega}) - \hat{y}_t]^2 \tag{2.21}$$

损失函数关于 $\boldsymbol{\omega}$ 的梯度为

$$\nabla_{\boldsymbol{\omega}} L(\boldsymbol{\omega}) = (\hat{q}_t - \hat{y}_t) \nabla_{\boldsymbol{\omega}} q(s_t, a_t; \boldsymbol{\omega}) \tag{2.22}$$

随后使用梯度下降法更新价值网络参数 $\boldsymbol{\omega}$。

智能体与环境交互，记录下观测到的状态、动作、奖励，用这些经验来学习一个策略函数。经验回放通过收集智能体与环境交互的记录存储到数组，事后反复利用这些

经验训练智能体，用于提升强化学习性能。然而由于高估问题的存在，使用目标网络、双 Q 学习算法[135] 来进行缓解。

2.3.3 基于策略迭代的深度强化学习

基于策略迭代的强化学习算法通过学习最优策略函数或近似函数（策略网络）来求解优化问题，可以表述为

$$\max_{\theta} \left\{ J(\theta) \triangleq \mathbb{E}_S \left[V_\pi(S) \right] \right\} \tag{2.23}$$

求解最大化问题的方法即梯度上升：

$$\theta \leftarrow \theta + \beta \nabla_\theta J(\theta) \tag{2.24}$$

其中 $\nabla_\theta J(\theta)$ 为策略梯度。策略梯度定理证明：

$$\nabla_\theta J(\boldsymbol{\theta}) = \mathbb{E}_S \left[\mathbb{E}_{A \sim \pi(\cdot|S;\boldsymbol{\theta})} \left[Q_\pi(S, A) \cdot \nabla_\theta \ln \pi(A|S;\boldsymbol{\theta}) \right] \right] \tag{2.25}$$

通过蒙特卡洛近似和随机抽样得出一个动作，计算随机梯度：

$$\boldsymbol{g}(s, a;\boldsymbol{\theta}) \triangleq Q_\pi(s, a) \cdot \nabla_{\boldsymbol{\theta}} \ln \pi(a|s;\boldsymbol{\theta}) \tag{2.26}$$

其为策略梯度的无偏估计。可以使用随机梯度上升更新 $\boldsymbol{\theta}$。由于其动作价值函数 Q_π 未知, 因此需要采用 REINFORCE[136] 或演员评论家（Actor Critic, AC）或优势演员评论家（Advantage Actor Critic, A2C）[137] 方法进行近似, 这种策略网络带有随机性。而确定性策略梯度的策略网络则直接输出确定性的动作，常见的有确定性策略梯度（Deterministic Policy Gradient, DPG）算法[138]、深度确定性策略梯度（Deep Deterministic Policy Gradient, DDPG）算法[139]、双延迟深度确定性策略梯度（Twin Delayed Deep Deterministic Policy Gradient，TD3）算法等[140]。

以上算法在单智能体系统中运行良好，还可以扩展到多智能体系统中。在多智能体系统中，每个智能体的决策不仅与其所处的环境有关，还与其他智能体的决策密切相关。因此，多智能体系统的稳定性面临严峻挑战。只有当所有智能体之间达到均衡状态时，多智能体系统才能达到收敛状态。

2.4 协同感知

协同感知技术相较于单车自主感知具有明显优势，能够有效弥补单车智能局限性，如感知盲区、遮挡等，提升信息连贯性和准确性，为确保交通安全和提高交通效率发

挥了重要作用[141]。协同感知的基础支撑技术包括传感器技术、多传感器融合技术、通信技术与边缘计算技术，通信技术与边缘计算技术在前面的内容已进行介绍，本节着重介绍前两个技术。

传感器技术广泛应用的主要有视觉感知技术、激光雷达感知技术、毫米波雷达感知技术、多传感器融合感知技术。视觉感知技术应用广泛、成本低廉。在视觉特征提取方面，早期主要基于人为设计特征、像素值、颜色直方图、梯度信息等来实现感知信息识别和提取，聚焦目标的颜色信息和属性信息，而深度学习方法则自动从大量原始数据中学习、抽取高维且抽象的特征，需要通过大量数据投喂训练网络。典型的图像目标检测算法包括基于锚框的目标检测算法和无锚框的目标检测算法，基于锚框的检测算法可以分为侧重于精度的二阶段检测算法（如 R-CNN[142]、Fast R-CNN[143]）和侧重于速度的一阶段检测算法（SSD[144]、YOLO[145]）。无锚框的检测算法大致可分为两类，一类是直接预测边框（Foveabox[146]），另一类是基于关键点实现目标检测（CornerNet[147]、ExtremeNet[148]、CenterNet[149]）。环境因素会对视觉感知技术效果产生重要影响，如光线强度、大气能见度、高动态范围变化等。

激光雷达感知技术是一种重要的车辆感知技术，它通过测量激光脉冲的往返时间来计算车辆与物体之间的距离。在激光雷达感知过程中，三维点云是其对物体表面进行扫描得到一组无序点的集合，相比图像，它能够提供物体在三维空间中精确的位置信息。采集到的点云数据需要经过特征提取的多个阶段，首先是数据预处理阶段，包括点云的降采样、坐标转换以及点云的滤波处理，以提高后续特征提取的效率和准确性。接着是目标特征提取阶段，这一阶段主要包括点云区域分割和特征提取两个步骤。点云区域分割阶段涉及点云的聚类、特征模式分类以及确定特征模式区域，从而将点云数据分割成不同的目标区域。在特征提取阶段，选择合适的多维特征对点云数据进行提取，例如点数、距离、数据强度等特征，通过特征参数的确定、特征点的提取和特征值的计算，最终得到用于目标检测和识别的特征信息。这些特征信息对于提高车辆感知的准确性和可靠性具有重要意义[150]。三维点云数据具有不规则和高度稀疏的特点，直接在三维点云数据上移植二维检测方法比较困难。受图像中“像素”的影响，将点云数据转换为规则的网格形式，即“体素”[151–153]。早期，一些研究通过将体素编码为手工设计的特征，然后利用支持向量机（Support Vector Machines，SVM）或卷积神经

网络（Convolutional Neural Network，CNN）完成检测。然而这些手工设计的特征存在信息瓶颈，不足以应对多样的检测环境。随着人工智能技术的进步，机器学习特征替代手工设计特征，极大地推动了三维目标检测技术的发展。特别是引入三维稀疏卷积技术后[152]，基于体素的三维目标检测方法显示出更高的效率和更好的性能，实现了实时检测。尽管如此，体素化处理过程中不可避免地引入了量化误差，导致了一定的细粒度信息损失[154]。目前激光雷达点云目标检测方法主要有 LaserNet[155]、PIXOR[156]、PointPillars[157]、VoxelNet[151]、PV-RCNN[158]，可分为基于视图、体素和点云的方法，基于视图的方法首先将三维点云数据投影到二维平面上，如鸟瞰图（Bird Eye View，BEV），再使用二维卷积网络处理映射后的点云，大大降低计算的复杂度，如 PIXOR 利用高度和反射率将点云转换为 BEV 表示，是一种无需处理的、单级检测器，输出从像素级别的神经网络定向解码估计 3D 对象。基于体素的方法如 VoxelNet，是最早提出基于体素实现点云特征编码的模型之一。首先在三维空间执行体素划分，然后对每个体素内的点云随机采样，接着采用多层 PointNet[153] 结构作为体素特征编码器，实现体素特征编码。随后，通过三维卷积主干提取多尺度空间特征，并利用区域提议网络回归检测框的类别、位置和尺寸。此外如 PointPillars，通过学习组织在垂直列中的点云表示，包括三个阶段：将点云转换为稀疏伪图像的特征编码器网络；2D 卷积主干网络将伪图像处理为高级表示；检测并回归 3D 检测框的检测头。PointNet 是基于点云的方法，直接对原始点云处理，利用多层感知机（Multiplayer Perceptions，MLP）或图神经网络提取点的特征。PV-RCNN 结合了基于点和基于体素的特征编码方法优点，提出体素到关键点场景编码方案、栅格点的多尺度感兴趣域特征提取层。

多传感器融合感知通过有机地组合不同来源/传感器的信息片段，弥补了单一来源/传感器感知方法的缺陷，从而显著提升了感知性能。在多传感器融合中，主要有三种方法：数据层融合（前融合）、特征层融合（深度融合/中融合）和决策层融合（目标融合/后融合）。首先是数据层融合，它的优势在于数据信息丰富、完整度高、融合精度较高，但缺点是处理耗时长、运算量大、实时性差，可能存在噪声或缺失。其次是特征层融合，相对而言干扰数据较少、处理速度较快、精度较高，但需要单独设计特征处理及提取算法，且准确性受设计模型影响较大。最后是决策层融合，它所需通信量较小、容错能力强，但精度较低、难以考虑原始数据特征、误判概率较高。因此，在选择融合

方法时需要根据具体应用场景和要求选择和权衡，以达到最佳的感知性能。

协同感知技术是指将自动驾驶汽车与其他车辆或道路基础设施之间的数据融合，从而显著提升感知系统的效能和准确性。它可以看作是一种多传感器融合感知，只是融合的传感器数据是通过通信传输，来源于其他车辆或基础设施。根据数据共享和协作的不同阶段，协同感知技术可以分为前期协同、中期协同和后期协同。前期协同（也称为数据级协同或早期协同）涉及大量原始数据的传输和处理，需要开发高效的数据处理算法，以优化数据传输、减少延迟，并在保证精度的同时降低计算资源需求。中期协同（也称特征级协同）则涉及到特征提取过程中信息丢失和冗余问题，需要选择合适的特征选择算法来保证特征提取的准确性和有效性。后期协同（也称目标级协同）是在接收来自其他智能体的输出结果后进行空间变换和合并，这种合并策略虽然可能会导致数据的不完整或含噪声，但在资源利用和带宽占用上更为高效。协同感知技术有效地解决了单车智能感知中存在的遮挡、视角受限和远距离感知弱的问题。随着不同车辆之间传感器和感知模型的多样化，以及传感器模态和感知模型类别的增加，协同感知技术正朝着更加高效的异构模态和模型协作框架发展。

2.5 本章小结

本章主要介绍了车载边缘计算系统计算卸载所涉及的相关技术与原理。车路云一体化系统构成了宏观架构，其中车载边缘计算系统的计算卸载和协同感知是架构中的关键技术。同时，本章也详细阐述了强化学习领域的原理与代表性算法，这些算法用以解决计算卸载中的动态资源分配问题，并对协同感知技术的发展及其关键技术进行了详细介绍。总体而言，本章为后续论文研究工作梳理了方向，并奠定了理论基础。

第三章　面向程序关联性任务的车载边缘系统资源配置优化

3.1 引言

物联网（Internet of Things，IoT）和人工智能（Artificial Intelligence，AI）的快速发展导致了各种计算密集型和延迟敏感型车联网应用的出现，例如增强现实导航和自动驾驶。将所有计算任务卸载到远程云端会导致严重的回程负载和不可接受的延迟。通过车辆边缘计算系统，车辆用户可以通过 V2I 通信将任务卸载到边缘节点，以减少响应延迟，从而满足爆炸式增长的数据流量和车联网应用日益严格的要求。

任务计算需要用户的输入数据和相应的执行程序，大部分计算卸载的研究工作默认各个节点已经安装了相应的执行程序，而车辆之间的任务计算程序/模型可能不同，这就需要车辆缓存相应的任务计算程序，才具备执行任务计算的条件，称之为服务缓存，比如执行目标检测任务，输入数据为摄像头、雷达传感器采集的数据，执行该任务需要缓存目标检测服务程序，若下一时刻仍有相同任务需要执行计算则可以重复利用。相应的还有内容缓存，即内容分发网络（Content Delivery Network，CDN），与任务计算无关。然而，车辆、边缘服务器的缓存空间也有限，尤其是车辆任务类型需求增多时，同时不同边缘服务器的计算资源可能分布不均，加上程序关联性任务决定了缓存与卸载决策密切相关，如何在车载边缘系统进行服务缓存与计算卸载具有挑战性。

边缘计算系统中移动用户的计算卸载得到广泛的研究[159–162]，ZHANG L 等人通过联合缓存和计算卸载策略优化，利用新颖的双向计算任务模型最小化平均带宽消耗[159]。ELGENDY I A 等人提出了移动边缘计算的多用户多任务计算卸载和资源分配方案[162]。针对移动用户联合服务缓存、计算卸载和资源分配的研究取得了一些重要成果[163–167]。ZHANG G 等人研究了一般多用户多任务场景中的联合服务缓存、计算卸载和资源分配问题，旨在最小化所有用户计算成本和延迟成本的加权和[167]。车载边缘计算系统中的服务缓存与计算卸载只有少数工作[76–78, 168]，TANG C 等人将应用程序缓存应用于 VEC，以优化外包应用程序的响应时间，同时满足时隙跨度能耗约束。采用 Lyapunov 优化技术来求解约束问题。最后，将两种贪婪启发式算法合并到基于漂移加惩罚的算法中，以帮助找到近似最优解[76]。LAN D 等人提出了一种基于雾计算的三

层架构，其特征是具有服务缓存的计算卸载，并设计基于 DRL 算法的卸载策略，以减少任务计算延迟和能耗成本[77]。LI Z 等人提出基于 DRL 算法的协同任务卸载与服务缓存替换方案[78]。这些研究没有考虑车辆或边缘服务器存储和计算资源的限制。由于在车联网中车辆高速移动的特性，边缘节点服务器与不同网联车频繁交互，需要在更长时间维度上对缓存内容和卸载策略进行决策，因此，设计一种动态有效的服务缓存与计算卸载方法，既能保证车联网中车辆严格的多元化时延要求，又能满足各类计算设备的资源约束成为亟待解决的问题。

为应对上述挑战，本章研究了存储、计算资源有限的 VEC 系统服务缓存和计算卸载的联合优化，其本质是资源分配问题，考虑时变任务请求和动态网络拓扑，设计了一种边缘服务缓存和计算卸载框架，充分利用云端、边缘服务器和车辆之间的协作，平衡边缘服务器之间的计算负载。考虑动态多类型任务请求、卸载和服务缓存，以及每个时隙车辆与边缘服务器之间的动态信道条件，推导不同缓存和卸载耦合情形下的时延及能耗表达式，提出最小化长期累计平均任务处理时延优化问题，设计深度确定性策略梯度算法进行缓存和卸载决策，仿真结果表明与其他基线方案对比，取得更好的性能。

本章内容安排如下：3.1 节介绍了车联网资源分配与任务卸载研究现状、目前的不足，以及本章的主要贡献。3.2 节阐述了服务缓存与计算卸载模型并定义了优化问题。3.3 节设计基于深度强化学习的缓存与卸载策略。3.4 节搭建仿真实验环境并进行性能验证。3.5 节总结本章的研究内容。

3.2 车载边缘计算系统服务缓存与计算卸载模型

3.2.1 系统模型概述

本章提出车载边缘计算系统中的服务缓存与计算卸载场景，面向的即程序关联性任务卸载中的缓存与计算资源分配，如图3-1所示，车辆、路侧边缘计算设施配备不同的计算单元和缓存空间，处理移动车辆请求卸载的计算任务。计算任务及其类型由车辆随机产生，可以在车辆本地执行或通过 V2I 通信将任务上传到附近的边缘节点，边缘节点之间通过有线连接互相协作构成边缘池，即边缘节点可以将任务迁移到缓存有对应服务程序的其他边缘节点。最后，边缘节点为计算任务分配资源并定期更新缓存的服务程序。

考虑一个一般的边缘服务器协作组成的车载边缘计算系统网络，边缘节点集合表

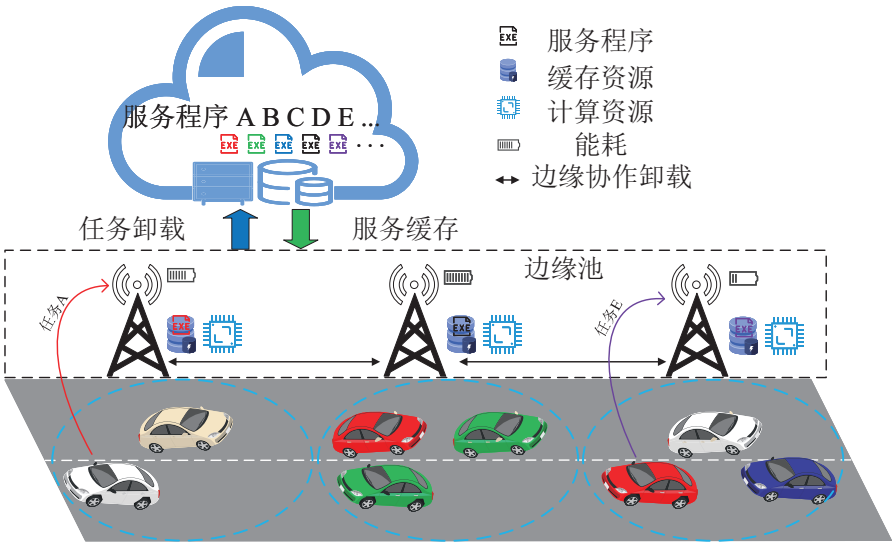


图 3-1 车载边缘计算系统服务缓存与计算卸载场景

Figure 3-1 Scenarios of service caching and computation offloading in vehicular edge computing systems

示为 $E=\{1,2,...e,...N_E\}$，移动车辆集合表示为 $V=\{1,2,...v,...\}$。云端提供所有的服务程序资源，假设一共有 N_K 个服务程序，在任意节点执行该任务必须缓存该任务对应的服务程序。如果服务程序缓存在边缘服务器池或云端，车辆通过 V2I 通信技术进行任务卸载，任务可任意进行分割，边缘池由一组互连的边缘服务器组成，互相协作用于平衡不同的计算和缓存资源。由于车辆的高速移动性，多个边缘服务器之间的协作卸载能够进一步提高缓存效率。云端的计算和存储资源丰富，边缘节点有限的缓存资源只能缓存一部分服务程序。时间划分为离散的时隙集合 $T=\{1,2,...t,...t_{end}\}$，每个时隙有相同的持续时间 Δt，车辆的程序关联性任务请求、边缘池的服务缓存和以及卸载策略在每个时隙进行更新。

3.2.2 通信模型

边缘节点能够感知到其覆盖范围区域的车辆，考虑多通道上行链路模型，每个边缘节点的总带宽为 B，假设不同边缘节点的频谱相同，并且与一个边缘节点相连接的车辆被分配一个正交信道。在此模型中，数据传输模式采用时分多址（Time Division Multiple Access，TDMA），其允许车辆在分配的时隙内传输数据，保障带宽合理分配并减少信道冲突。不同边缘节点的信道由于频谱重叠可能相互干扰，而同一边缘节点内的车辆通信不存在冲突。车辆和边缘节点之间的关联性通常发生在一个巨大的时间范围内（卸载事件的周期），该时间范围大于小规模衰落的时间范围。假设车辆在每个时隙 t 只能发

送一个任务请求，边缘节点 e 在时隙 t 内接收到的任务总和为 $N_e^{task}(t)$。在这种情况下，车辆 v 与边缘节点 e 之间的信号加干扰噪声比（Signal-to-Interference-plus-Noise-Ratio, SINR）为

$$\gamma_{v,e}(t)=\frac{g_{v,e}(t)p_{v,e}(t)}{\sum_{e=1}^{N_E}g_{v,e}(t)\sum_{v=1}^{N_e^{task}(t)}p_{v,e}(t)+\sigma^2} \tag{3.1}$$

其中 σ^2 是加性高斯白噪声的方差，$g_{v,e}(t)$ 表示边缘节点 e 与车辆 v 通信的平均信道增益，$p_{v,e}(t)$ 表示边缘节点 e 与车辆 v 通信的平均信道传输功率。

若车辆 v 在时隙 t 将任务输入数据卸载到最近的边缘节点 e，其无线传输速率根据香农公式表示为

$$R_{v,e}(t)=B_{v,e}(t)log_2(1+\gamma_{v,e}(t)) \tag{3.2}$$

其中 $B_{v,e}(t)$ 是边缘节点 e 分配给车辆 v 的带宽，$B_{v,e}(t)=B/N_e^{task}(t)$。

3.2.3 服务缓存与计算卸载

在每个时隙的开始阶段，车辆进入边缘节点的通信覆盖范围内，更新任务请求，完成任务卸载和计算，在一个时隙内完成该流程。在每个时隙的结束之前，边缘节点的服务缓存由部署在边缘节点的管理控制器做出更新决策，任务请求车辆的服务缓存由车辆自身根据兴趣进行决策。$K=\{1,2,...k,...N_k\}$ 表示服务程序集合，每个任务的相关属性可以表示为 $\{d_k,\theta_k\},k\in K$，d_k 表示任务输入数据大小，θ_k 代表任务 k 关联程序要求的存储空间大小，由于车辆和边缘节点存储空间有限，其约束如下

$$\sum_{k=1}^{N_K}c_{v,k}^V(t)\theta_k\le S_v^V,\forall v,t \tag{3.3}$$

$$\sum_{k=1}^{N_K}c_{e,k}^E(t)\theta_k\le S_e^E,\forall v,t \tag{3.4}$$

其中 $c_{v,k}^V(t)\in\{0,1\},c_{e,k}^E(t)\in\{0,1\}$ 是二进制决策变量，表示服务程序 k 是否缓存，$c_{v,k}^V(t)=1,c_{e,k}^E(t)=1)$ 表示车辆 v 和边缘节点 e 在时隙 t 内均缓存了服务程序。S_v^V 和 S_e^E 分别是每辆车和边缘节点的存储容量。

在系统模型中，车辆请求任务需要根据节点的缓存情形进而做出下一步的卸载策略，其任务卸载流程如图3-2所示，从任务卸载请求开始，判断任务关联程序是否缓存

在本地，如果是，则暂时进入等待状态，进一步对最近的边缘节点缓存程序进行判断，如果边缘节点没有缓存，这对应情形 1 的情况，任务在本地执行计算，无需进行任何卸载流程。若最近的边缘节点同样缓存了服务程序，由于任务是可以分割的，此时任务可以进行部分卸载，对应情形 2 的情况，此时任务进行分割同时在本地和边缘节点执行计算。若从任务卸载请求开始，车辆本地没有缓存相应的程序，则任务输入数据先完全卸载到边缘节点，若最近的边缘节点缓存有服务程序，则先进入等待，对边缘池其他节点缓存程序判断，若没有，则对应情形 3 的情况，此任务在最近的边缘节点执行该计算。若边缘池缓存有，则任务可以进行部分卸载，这对应情形 4 的情况，则此时任务在边缘池计算。若最近的边缘节点没有缓存，则此时对边缘池的其他节点缓存情形进行判断，若有，则对应情形 5，此时任务在边缘池的其他节点执行计算。若边缘池其他节点仍没有缓存对应的服务程序，则任务完全卸载到云端执行计算，这对应情形 6 的情况。边缘服务器的智能体负责服务缓存与卸载决策，即资源分配。这个过程中边缘池中边缘节点的资源存在不确定性，任务车辆不能及时准确获取边缘池的实时资源状态，因此，资源发现对计算卸载具有重要意义，利用 NFV 技术对分散的计算资源进行收集，形成虚拟计算资源池，利用 SDN 技术对计算资源有效配置来满足计算任务需求，智能体需要根据车辆动态请求和拓扑变化以及边缘节点资源不确定性快速做出反应，合理进行决策，提高车辆网中计算任务卸载的可靠性。

3.2.4 计算延迟和能耗分析

基于已有的任务卸载流程，对任务计算时延及能量消耗进行分析。任务 k 的计算时延和能量消耗表示如下[165]。

$$D_k = \frac{\omega_k}{f_k} \tag{3.5}$$

$$\varepsilon_k = \kappa f_k^{\alpha} D_k = \kappa \omega_k (f_k)^{\alpha-1} \tag{3.6}$$

其中 f_k 代表设备 CPU 的计算频率，受到最大频率的限制 f_{max} （$f_k \le f_{max}$）。κ ($\kappa > 0$) 表示计算能量效率参数，α (本文假设 $\alpha = 2$) 表示指数参数。ω_k 代表服务程序 k 需要的计算轮次，与计算任务输入数据大小相关，$\omega_k = \lambda d_k$，其中 λ ($\lambda > 0$) 由服务应用的计算复杂度决定[169]。f^V 和 f^E 分别表示车辆和边缘服务器节点的计算频率。

当车辆 v 进入到最近的边缘节点服务器 e 的通信范围, 并发起一个任务 k 的卸载请

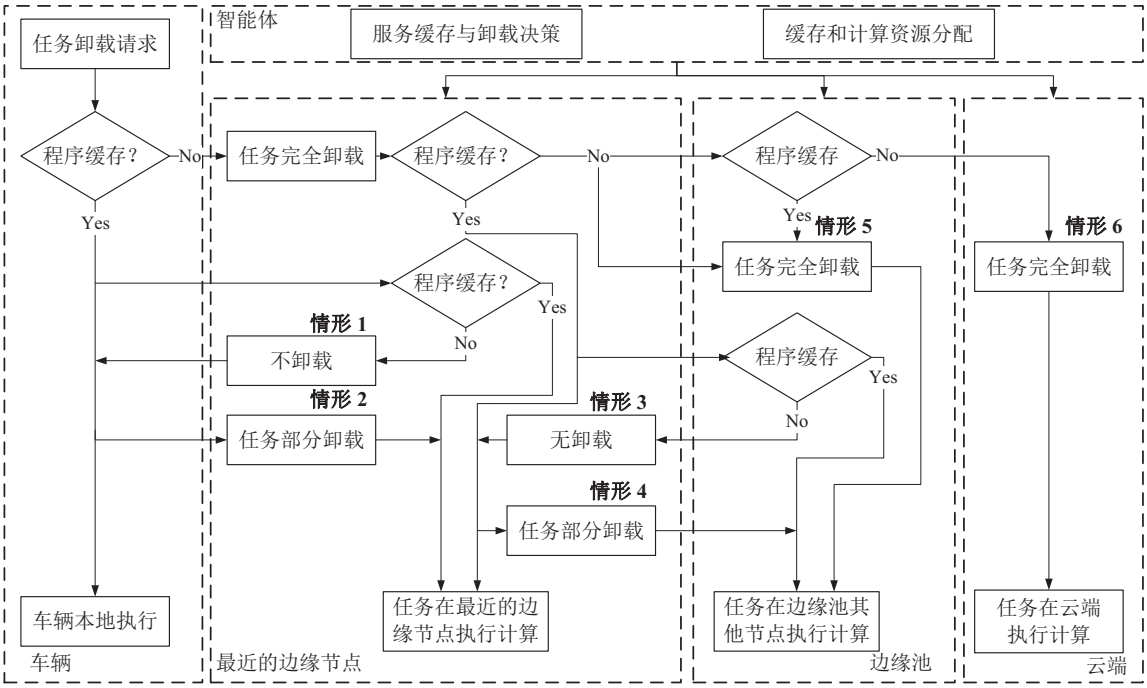


图 3-2 任务卸载流程

Figure 3-2 The process of task offloading

求，需要根据服务缓存的情形，对卸载的目标节点和卸载比例进行决策，在任务卸载流程中已经总结出了六种情形，需要分析出六种情形对应的服务缓存变量和卸载变量的表达式。$o^V_{v,k}(t) \in [0,1], o^E_{e,k}(t) \in [0,1]$ 均为连续变量，分别代表任务 k 的输入数据卸载到最近的边缘节点的比例和卸载到边缘池其他边缘节点的比例，$(1\text{-}o^V_{v,k}(t))$ 和 $(1\text{-}o^E_{e,k}(t))$ 分别代表任务 k 的输入数据在车辆 v 本地执行计算和在最近的边缘节点 e 执行计算的比例。表3-1展示了不同服务缓存情形对应的缓存决策变量表达式和相应的任务卸载表达式。接下来将基于表3-1推导相关时延和能耗的表达式。

时隙 t 内，车辆 v 请求任务 k 对应情形 1 的情况，其中 $\sum_{i=1,i\neq e}^{N_E} c^E_{i,k}(t) \geq 0$ 代表边缘池其他节点的缓存情形，但及时有也不再通过最近的边缘节点进行卸载，因此本地执行计算的时延如下

$$T^{local}_{v,e,k}(t) = c^V_{v,k}(t)(1 - o^V_{v,k}(t))\frac{\lambda d_k}{f^V} \tag{3.7}$$

若任务 k 卸载到最近的边缘节点 e，则上传时间为

$$T^{up}_{v,e,k}(t) = \begin{cases} \varphi(\sum_{e=1}^{N_E} c^E_{e,k}(t)) o^V_{v,k}(t)\frac{d_k}{R_{v,e}(t)}, & R_{v,e}(t) > 0 \\ 0, & R_{v,e}(t) = 0 \end{cases} \tag{3.8}$$

表 3-1 不同服务缓存情形任务卸载比例

Table 3-1 The task offloading ratio in various service caching scenarios

服务缓存情形		任务卸载比例			
情形	变量描述	任务在车辆本地执行	任务在最近的边缘节点执行计算	任务在边缘池其他节点执行	任务在云端执行
情形 1	$c_{v,k}^V(t)=1, c_{e,k}^E(t)=0, \sum_{i=1,i\neq e}^{N_E} c_{i,k}^E(t)\geq 0$	$1-o_{v,k}^V(t)=1$	$o_{v,k}^V(t)=0$	0	0
情形 2	$c_{v,k}^V(t)=1, c_{e,k}^E(t)=1, \sum_{i=1,i\neq e}^{N_E} c_{i,k}^E(t)\geq 0$	$1-o_{v,k}^V(t)$	$o_{v,k}^V(t)$	0	0
情形 3	$c_{v,k}^V(t)=0, c_{e,k}^E(t)=1, \sum_{i=1,i\neq e}^{N_E} c_{i,k}^E(t)=0$	$1-o_{v,k}^V(t)=0$	$o_{v,k}^V(t)(1-o_{e,k}^E(t))=1$	$o_{v,k}^V(t)o_{e,k}^E(t)=0$	0
情形 4	$c_{v,k}^V(t)=0, c_{e,k}^E(t)=1, \sum_{i=1,i\neq e}^{N_E} c_{i,k}^E(t)>0$	$1-o_{v,k}^V(t)=0$	$o_{v,k}^V(t)(1-o_{e,k}^E(t))$	$o_{v,k}^V(t)o_{e,k}^E(t)$	0
情形 5	$c_{v,k}^V(t)=0, c_{e,k}^E(t)=0, \sum_{i=1,i\neq e}^{N_E} c_{i,k}^E(t)>0$	$1-o_{v,k}^V(t)=0$	$o_{v,k}^V(t)(1-o_{e,k}^E(t))=0$	$o_{v,k}^V(t)o_{e,k}^E(t)=1$	0
情形 6	$c_{v,k}^V(t)=0, c_{e,k}^E(t)=0, \sum_{i=1,i\neq e}^{N_E} c_{i,k}^E(t)=0$	0	0	0	1

其中

$$\varphi(x)=\begin{cases}1, & \text{x>0}\\ 0, & \text{x=0}\end{cases} \tag{3.9}$$

式(3.8)中 $R_{v,e}(t)=0$ 表示缓存情况属于情形 1，任务不需要卸载到边缘节点。

任务在最近的边缘节点服务器执行计算的时延为

$$T_{v,e,k}^{edge}(t)=c_{e,k}^E(t)o_{v,k}^V(t)(1-o_{e,k}^E(t))\frac{\lambda d_k}{fE} \tag{3.10}$$

如果最近的边缘节点服务器 e 没有缓存，任务 k 的输入数据需要卸载到边缘池其他服务节点，传输时间为

$$T_{v,e,k}^{uppool}(t)=(1-c_{v,k}^V(t))\varphi(\sum_{i=1,i\neq e}^{N_E} c_{i,k}^E(t))o_{v,k}^V(t)o_{e,k}^E(t)\times\frac{d_k}{R_{edge}} \tag{3.11}$$

其中 R_{edge} 表示边缘服务器之间的传输速率，任务卸载到边缘池的计算时延为

$$T_{v,e,k}^{pool}(t) = (1 - c_{v,k}^{V}(t))\varphi(\sum_{i=1,i\neq e}^{N_E} c_{i,k}^{E}(t))o_{v,k}^{V}(t)o_{e,k}^{E}(t) \times \frac{\lambda d_k}{fE} \tag{3.12}$$

由于计算任务输出结果数据远小于任务输入数据，将结果返回过程的时延进行忽略[75]。云端的存储和计算资源丰富，忽略在云端的排队和计算延迟，考虑卸载到云端的传输时延。基于以上的分析，因此考虑所有情形下的缓存和卸载决策情形，任务 k 在时隙 t 内的任务处理时延可以表示为

$$\begin{aligned} T_{v,e,k}^{total}(t) = &\max\{T_{v,e,k}^{local}(t), T_{v,e,k}^{up}(t) + \max\{T_{v,e,k}^{edge}(t), T_{v,e,k}^{uppool}(t) + T_{v,e,k}^{pool}(t)\}\} \\ &+ (1 - c_{v,k}^{V}(t))(1 - \varphi(\sum_{e=1}^{N_E} c_{e,k}^{E}(t))) \times (\frac{d_k}{R_{v,e}(t)} + \frac{d_k}{R_{cloud}}) \end{aligned} \tag{3.13}$$

式(3.13)能够表示表3-1中的所有情形对应的任务卸载、传输以及计算的时延。R_{cloud} 表示边缘服务器到云端的传输时延。

此外，从边缘服务器运营商的角度考虑任务卸载、传输以及计算产生的能量消耗为

$$\begin{aligned} \varepsilon_{v,e,k}^{total}(t) = &\kappa(f^E)^2(T_{v,e,k}^{edge}(t) + T_{v,e,k}^{pool}(t)) + p_{edge}T_{v,e,k}^{uppool}(t) \\ &+ (1 - c_{v,k}^{V}(t))(1 - \varphi(\sum_{e=1}^{N_E} c_{e,k}^{E}(t)))\frac{p_{cloud}d_k}{R_{cloud}} \end{aligned} \tag{3.14}$$

其中 p_{edge} 表示边缘池中边缘服务器之间协作卸载的传输功率，p_{cloud} 表示边缘服务器与云端之间的传输功率。

定义时隙 t 内所有请求任务的平均处理时延为

$$T_d(t) = \frac{1}{N_E}\sum_{e=1}^{N_E}\frac{1}{N_e^{task}(t)}\sum_{v=1}^{N_e^{task}(t)} T_{v,e,k}^{total}(t) \tag{3.15}$$

时隙 t 内边缘服务器的平均能量消耗为

$$\varepsilon(t) = \frac{1}{N_E}\sum_{e=1}^{N_E}\frac{1}{N_e^{task}(t)}\sum_{v=1}^{N_e^{task}(t)} \varepsilon_{v,e,k}^{total}(t) \tag{3.16}$$

由于缓存变量和决策变量互相耦合，不便对平均处理时延 $T_{v,e,k}^{total}(t)$ 值的范围评估，因此对在不同服务缓存情形下的时延分析能够消去缓存变量，简化复杂的公式，接下来是不同情形下消去缓存变量的时延表达式。

1）情形 1

$$T_d^{Case1}(t)=\frac{\lambda d_k}{f^V} \tag{3.17}$$

2）情形 2

$$T_d^{Case2}(t)=\max\{(1-o_{v,k}^V(t))\frac{\lambda d_k}{f^V},o_{v,k}^V(t)(\frac{d_k}{R_{v,e}(t)}+\frac{\lambda d_k}{f^E})\} \tag{3.18}$$

3）情形 3

$$T_d^{Case3}(t)=\frac{d_k}{R_{v,e}(t)}+\frac{\lambda d_k}{f^E} \tag{3.19}$$

4）情形 4

$$T_d^{Case4}(t)=\frac{d_k}{R_{v,e}}+\max\{(1-o_{e,k}^E(t))\frac{\lambda d_k}{f^E},o_{e,k}^E(t)(\frac{d_k}{R_{edge}}+\frac{\lambda d_k}{f^E})\} \tag{3.20}$$

5）情形 5

$$T_d^{Case5}(t)=\frac{d_k}{R_{v,e}(t)}+\frac{d_k}{R_{edge}}+\frac{\lambda d_k}{f^E} \tag{3.21}$$

6）情形 6

$$T_d^{Case6}(t)=\frac{d_k}{R_{v,e}(t)}+\frac{d_k}{R_{cloud}} \tag{3.22}$$

从以上情形分析得到 $T_{v,e,k}^{total}(t)$ 的最大值为

$$\begin{aligned}\max\{T_{v,e,k}^{total}(t)\}&=\max_{\{i=1,\dots6\}}\{T_d^{Casei}(t)\}\\&=\max\{T_d^{Case1}(t),T_d^{Case3}(t),T_d^{Case4}(t),T_d^{Case5}(t),T_d^{Case6}(t)\}\\&=\max\{T_d^{Case1}(t),T_d^{Case5}(t),T_d^{Case6}(t)\}\\&=\max\{\frac{\lambda d_k}{f^V},\frac{d_k}{R_{v,e}(t)}+\frac{d_k}{R_{edge}}+\frac{\lambda d_k}{f^E},\frac{d_k}{R_{v,e}(t)}+\frac{d_k}{R_{cloud}}\}\end{aligned} \tag{3.23}$$

式(3.23)的分析过程为，先看情形 1、2、3，情形 1 和情形 3 已经确定，情形 2 由于两个函数均为关于卸载决策变量 $o_{v,k}^V(t)$ 的线性函数，故最大值在卸载变量为极端的 0 或 1 的情形，而极端的情形其取值即为情形 1 或情形 3 的值，随后看情形 3、4、6，此时情形 4 中两个函数也是均为关于卸载决策变量 $o_{v,k}^V(t)$ 的线性函数，故其极端的情形为情形 3 或情形 5 的值，比较情形 3 和 5 得出情形 5 的值较大，故最大任务处理时延发生的情况在情形 1、5、6 中，如果继续做进一步判断则与计算设备的计算频率、传输速

率有关。由于情形 1 和情形 3 是情形 2 的极端情况，所以最小值可能在情形 2，同时由于情形 3 和情形 5 是情形 4 的极端情况，所以最小值可能在情形 4 的情况，综合考虑，最小值发生情形 2、4、6 中，如下

$$\begin{aligned}\min\{T_{v,e,k}^{total}(t)\} &= \min_{\{i=1,\dots 6\}}\{T_d^{Casei}(t)\} \\ &= \min\{T_d^{Case2}(t), T_d^{Case4}(t), T_d^{Case6}(t)\}\end{aligned} \tag{3.24}$$

进一步分析情形 2 的最小值，令其相等:

$$(1-o_{v,k}^{V}(t))\frac{\lambda d_k}{f^V} = o_{v,k}^{V}(t)d_k(\frac{1}{R_{v,e}(t)}+\frac{\lambda}{f^E}) \tag{3.25}$$

得

$$o_{v,k}^{V}(t) = \frac{1}{1+\frac{f^V}{\lambda}(\frac{1}{R_{v,e}(t)}+\frac{\lambda}{f^E})} \tag{3.26}$$

所以当 $o_{v,k}^{V}(t) = \frac{1}{1+\frac{f^V}{\lambda}(\frac{1}{R_{v,e}(t)}+\frac{\lambda}{f^E})}$ 时，T_d^{Case2} 取得最小值:

$$T_{dmin}^{case2} = \frac{d_k(f^E+\lambda R_{v,e}(t))}{\frac{f^V f^E}{\lambda}+f^V R_{v,e}(t)+f^E R_{v,e}(t)} \tag{3.27}$$

分析情形 4 的最小值，令其相等:

$$(1-o_{e,k}^{E}(t))\frac{\lambda d_k}{f^E} = o_{e,k}^{E}(t)d_k(\frac{1}{R_{edge}+\frac{\lambda}{f^E}}) \tag{3.28}$$

得

$$o_{e,k}^{E}(t) = \frac{1}{2+\frac{f^E}{\lambda R_{edge}}} \tag{3.29}$$

所以当 $o_{e,k}^{E}(t) = \frac{1}{2+\frac{f^E}{\lambda R_{edge}}}$ 时，T_d^{Case4} 取得最小值

$$T_{dmin}^{case4} = \frac{\lambda d_k(\lambda R_{edge}+f^E)}{f^E(2\lambda R_{edge}+f^E)} \tag{3.30}$$

所以 $T_{v,e,k}^{total}(t)$ 最小值为

$$\min\{T_{v,e,k}^{total}(t)\} = \min\{\min\{T_d^{Case2}(t)\}, \min\{T_d^{Case4}(t)\}, T_d^{Case6}(t)\} \tag{3.31}$$

通过以上分析得到任务处理时延的最大值和最小值与每个节点设备的通信、计算、缓存资源相关，想最小化任务处理时延，则要尽可能避免做出产生时延较大的缓存情形

决策。

同理，对边缘服务器的能量消耗 $\varepsilon_{v,e,k}^{total}(t)$ 进行值的范围评估，在不同服务缓存情形下的能量消耗分析能够消去缓存变量，简化复杂的公式。接下来是不同情形下消去缓存变量的能耗表达式。

1）情形 1

$$\varepsilon^{Case1}(t) = 0 \tag{3.32}$$

2）情形 2

$$\begin{aligned} \varepsilon^{Case2}(t) &= \kappa(f^E)^2(o_{v,k}(t)\frac{\lambda d_k}{f^E}) \\ &= \kappa f^E o_{v,k}(t)\lambda d_k \end{aligned} \tag{3.33}$$

3）情形 3

$$\begin{aligned} \varepsilon^{Case3}(t) &= \kappa(f^E)^2(\frac{\lambda d_k}{f^E}) \\ &= \kappa f^E \lambda d_k \end{aligned} \tag{3.34}$$

4）情形 4

$$\begin{aligned} \varepsilon^{Case4}(t) &= \kappa(f^E)^2((1-o_{e,k}(t))\frac{\lambda d_k}{f^E} + o_{e,k}(t)\frac{\lambda d_k}{f^E}) + o_{e,k}(t)p_{edge}\frac{d_k}{R_{edge}} \\ &= \kappa f^E \lambda d_k + o_{e,k}(t)p_{edge}\frac{d_k}{R_{edge}} \end{aligned} \tag{3.35}$$

5）情形 5

$$\varepsilon^{Case5}(t) = \kappa f^E \lambda d_k + p_{edge}\frac{d_k}{R_{edge}} \tag{3.36}$$

6）情形 6

$$\varepsilon^{Case6}(t) = p_{cloud}\frac{d_k}{R_{cloud}} \tag{3.37}$$

通过以上情形得到 $\varepsilon_{v,e,k}^{total}(t)$ 的最小值为 0，因为是从边缘计算服务器运营商的角度考虑耗能。最大值如下

$$\begin{aligned} \max\{\varepsilon_{v,e,k}^{total}(t)\} &= \max_{\{i=1,\ldots 6\}}\{\varepsilon^{Casei}(t)\} \\ &= \max\{\varepsilon^{Case5}(t), \varepsilon^{Case6}(t)\} \\ &= \max\{\kappa f^E \lambda d_k + p_{edge}\frac{d_k}{R_{edge}}, p_{cloud}\frac{d_k}{R_{cloud}}\} \end{aligned} \tag{3.38}$$

综合以上推导和分析，为了减少任务处理延迟，需要尽可能做出情形 2、情形 4、情形 6 之类的服务缓存决策，任务尽可能卸载到边缘节点，利用边缘节点较大的计算存储资源减少任务计算时延。另一方面，能量消耗是边缘服务器运营商关注的关键指标，将任务卸载到边缘节点可以减少任务处理时延，但也会增加边缘服务器的能耗。

3.2.5 优化问题描述

任务处理时延是车联网相关业务的关键指标，本章针对一般性的应用场景，从边缘节点服务器运营商角度优化整体长期性的指标，旨在设计联合服务缓存和计算卸载方案最小化长期累计平均任务时延，其优化变量为每个时隙 t 内边缘服务器的缓存决策和车辆及边缘服务器的任务卸载决策，优化问题描述为

$$\min_{\{c_{e,k}^{E}(t),o_{v,k}^{V}(t),o_{e,k}^{E}(t)\}} \quad \sum_{t=1}^{t_{end}} \gamma^{t-1}\left(\frac{T_d(t)}{T_{dmax}}\right) \tag{3.39}$$

$$s.t.$$

$$c_{v,k}^{V}(t) \in \{0,1\}, \forall v \in V, \forall k \in K, \forall t \in \{1,2,...,t_{end}\} \tag{3.40}$$

$$c_{e,k}^{E}(t) \in \{0,1\}, \forall e \in E, k, t \tag{3.41}$$

$$0 \le o_{v,k}^{V}(t) \le 1, \forall v, k, t \tag{3.42}$$

$$0 \le o_{e,k}^{E}(t) \le 1, \forall e, k, t \tag{3.43}$$

$$0 \le \gamma \le 1 \tag{3.44}$$

$$\sum_{k=1}^{N_K} c_{v,k}^{V}(t)\theta_k \le S_v^{V}, \forall v, t \tag{3.45}$$

$$\sum_{k=1}^{N_K} c_{e,k}^{E}(t)\theta_k \le S_e^{E}, \forall e, t \tag{3.46}$$

其中 γ 是折扣因子，T_{dmax} 是最大容忍延时，$T_{dmax} = \max\{\max\{T_d(t)\}\} = \max\{\max\{T_{v,e,k}^{total}(t)\}\}$，约束(3.40)和(3.41)是缓存变量取值范围，约束(3.42)和(3.43)是任务卸载变量的取值范围，约束(3.45)和(3.46)是每个车辆和边缘服务器的缓存容量限制。此问题是一个长期混合整数非线性规划（Mixed Integer Nonlinear Programming，MINLP）问题。每个时隙需要对服务缓存和任务卸载做出适当的决策，每个时隙参与的车辆状态变化以及短暂的交互增加了边缘管理控制器的操作复杂性，系统状态空间随车辆和边缘节点服务器数量增加而增加，需要找到一种有效方法解决此类问题。传统优化算法无法满足复杂、动态车联网环境中计算任务卸载问题的求解，深度强化学习算法能够充分利用在环境感知、交互和智能决策等方面的性能优势，通过不断的探索和利用，

可以适应动态变化的环境，并优化长期的资源配置策略，满足该场景中服务缓存与计算任务卸载的需求。

3.3 基于深度强化学习的服务缓存与计算卸载策略

3.3.1 马尔可夫决策过程

由于不同用户需求、有限的计算和缓存资源以及动态变化的任务请求，最小化累计系统平均延时是复杂的，凸优化[170]和博弈论[171]等广泛应用于车联网系统解决缓存和计算卸载问题，通常都假设已知某些关键因素为前提，例如无线信道状态等，这些因素通常是时变的且难以获得的，且通常实现某一个时间点的最优或接近最优结果，忽略了当前的决策对后续长期时间维度上的影响[172]。深度强化学习被视为通过优化长期累计奖励来解决动态环境中复杂问题的有效方法，智能体收集请求用户不同需求和车联网中可用资源的所需信息，智能体执行动作管理缓存和计算资源，并做出卸载决策，能够通过不断反馈学习调整策略实现长期奖励最大化。本章优化问题既有离散又有连续变量，采用基于深度确定性策略梯度算法能够有效处理具备连续行为空间的问题。

布置在边缘缓存服务器的车联网中心控制器接收所有边缘节点收集到的车辆任务相关信息和服务器的状态相关信息作为环境输入，然后建立三要素：状态、动作和奖励，通常描述为马尔可夫决策过程，下面将依次描述服务缓存与计算卸载模型抽离出的状态空间、动作空间和奖励。

1）状态空间：在每个时隙 t 的初始阶段，每个边缘服务器收集所有的环境参数，包括以下部分：

- $I_{v,k}(t)$：车辆 v 在时隙 t 内处于边缘节点 e 的通信覆盖范围内对任务 k 的请求指示因子，
- $c^{V}_{v,k}(t)$：车辆 v 在时隙 t 内处于边缘节点 e 的通信覆盖范围内对任务关联性程序 k 的服务缓存指示因子，
- $B_{v,e}(t), \gamma_{v,e}(t)$：时隙 t 内，边缘节点 e 分配给车辆 v 的的带宽以及边缘节点的接收 SINR，
- $c^{E}_{e,k}(t)$：边缘节点 e 在时隙 t 内的对任务关联性程序 k 的服务缓存指示因子。

边缘节点 e 在时隙 t 的状态表示为

$$\mathbf{s}_e(t) = \{[\mathbf{I}(t)]^{\rho_{max}\times N_K}, [\mathbf{c}^V(t)]^{\rho_{max}\times N_K}, [\mathbf{B}(t)]^{\rho_{max}\times N_K}, [\boldsymbol{\gamma}(t)]^{\rho_{max}\times N_K}, [\mathbf{c}^E(t)]^{\rho_{max}\times N_K}\} \quad (3.47)$$

其中 ρ_{max} 表示每个边缘节点 e 的最大连接车辆密度，加粗的字母代表矩阵。整个系统的环境状态由所有边缘节点的状态组成，表示如下

$$s_t = \{s_1(t), s_2(t), ..., s_{N_E}(t)\} \quad (3.48)$$

其中边缘节点 e 的状态经过降维和归一化后表示为 $s_e(t)$。

2）动作空间：智能体获取边缘节点服务缓存的状态信息、车辆与边缘节点通信状态信息，随后对所有请求任务的卸载比例进行决策，并更新所有边缘节点的服务缓存信息，包含如下部分：

- $c^E_{e,k}(t)$：时隙 t 内边缘节点 e 完成所有任务 k 的卸载及计算后的服务缓存指示因子，
- $o^V_{v,k}(t)$：时隙 t 内车辆 v 卸载任务 k 的输入数据到边缘节点 e 的比例。
- $o^E_{e,k}(t)$：时隙 t 内边缘节点 e 卸载任务 k 的输入数据到边缘池其他节点的比例。

边缘节点 e 在时隙 t 内执行的动作表示为

$$\mathbf{a}_e(t) = \{[\mathbf{c}^E(t)]^{\rho_{max}\times N_K}, [\mathbf{o}^V(t)]^{\rho_{max}\times N_K}, [\mathbf{o}^E(t)]^{\rho_{max}\times N_K}\} \quad (3.49)$$

所有边缘节点执行的动作经过降维和归一化后表示为 $a_e(t)$，整个系统在时隙 t 执行的动作由所有边缘节点的动作组成，表示如下

$$a_t = \{a_1(t), a_2(t), ..., a_{N_E}(t)\} \quad (3.50)$$

3）奖励：智能体的行为基于奖励，奖励与优化问题目标函数相关，时隙 t 的奖励定义为

$$r_t = r(s_t, a_t) = -(\frac{T_d(t)}{T_{dmax}}) \quad (3.51)$$

其奖励累计值即为优化问题的目标函数的负值，等价于最大化累计奖励值，这正好是强化学习的优化目标。

3.3.2 基于深度确定性策略算法的缓存与卸载方案

由于状态空间由大量动态环境信息组成，而动作空间包含连续值，因此利用深度确定性策略梯度（DDPG）算法（一种无模型和演员评论家算法）来解决联合服务缓存和计算卸载问题。基于 DDPG 的方法的框架如图3-3所示，由主网络、目标网络和经验回放区组成，主网络和目标网络分别基于 Actor 模型和 Critic 模型的深度神经网络构成，Actor 网络和 Critic 网络分别去近似确定性策略函数 $\mu(s_t)$ 和动作价值函数 $Q(s_t, a_t)$，Actor 网络负责做决策输出动作，Critic 网络负责对动作进行打分，环境给定的奖励是一个真实值，而 Actor 网络和 Critic 网络都是需要训练的，Actor 网络根据 Critic 网络的打分不断改进动作，而 Critic 网络依据环境给定的奖励不断提高打分标准，这样 Actor 网络执行动作越来越规范，Critic 网络打分越来越专业标准。

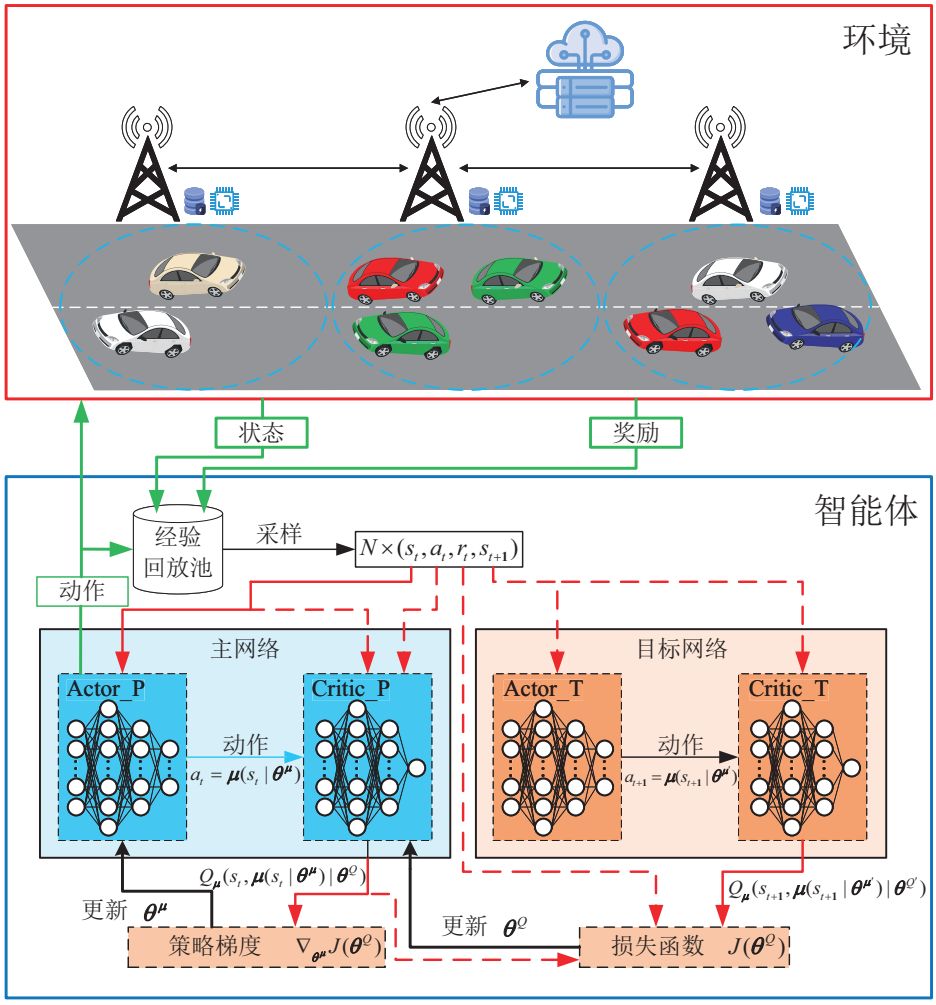


图 3-3 DDPG 算法框架图

Figure 3-3 Framework diagram of DDPG algorithm

确定性策略函数和动作价值函数之间满足贝尔曼方程：

$$Q_{\mu}(s_t, a_t) = \mathbb{E}_{\mu}\left[R_t | s_t, a_t\right] + \gamma \mathbb{E}_{\mu}\left[Q_{\mu}(S_{t+1}, A_{t+1}) | s_t, a_t\right] \tag{3.52}$$

对其进行蒙特卡洛近似：

$$Q_{\mu}(s_t, a_t) \approx r_t + \gamma Q_{\mu}(s_{t+1}, a_{t+1}) \tag{3.53}$$

主网络中使用两个不同的深度神经网络近似策略函数 $\mu(s_t|\theta^{\mu})$ 和动作价值函数 $Q(s_t, a_t|\theta^{Q})$，目标网络中同样使用两个不同的神经网络近似策略函数 $\mu(s_{t+1}|\theta^{\mu'})$ 和动作价值函数 $Q(s_{t+1}, a_{t+1}|\theta^{Q'})$，目标网络和主网络结构完全相同，仅仅参数不同，能够改善高估问题，经过深度神经网络近似后如下

$$Q(s_t, \mu(s_t|\theta^{\mu})|\theta^{Q}) \approx r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1}|\theta^{\mu'})|\theta^{Q'}) \tag{3.54}$$

目标动作价值表示为 y_t

$$y_t = r(s_t, a_t) + \gamma Q_{\mu}(s_{t+1}, \mu(s_{t+1}|\theta^{\mu'})|\theta^{Q'}) \tag{3.55}$$

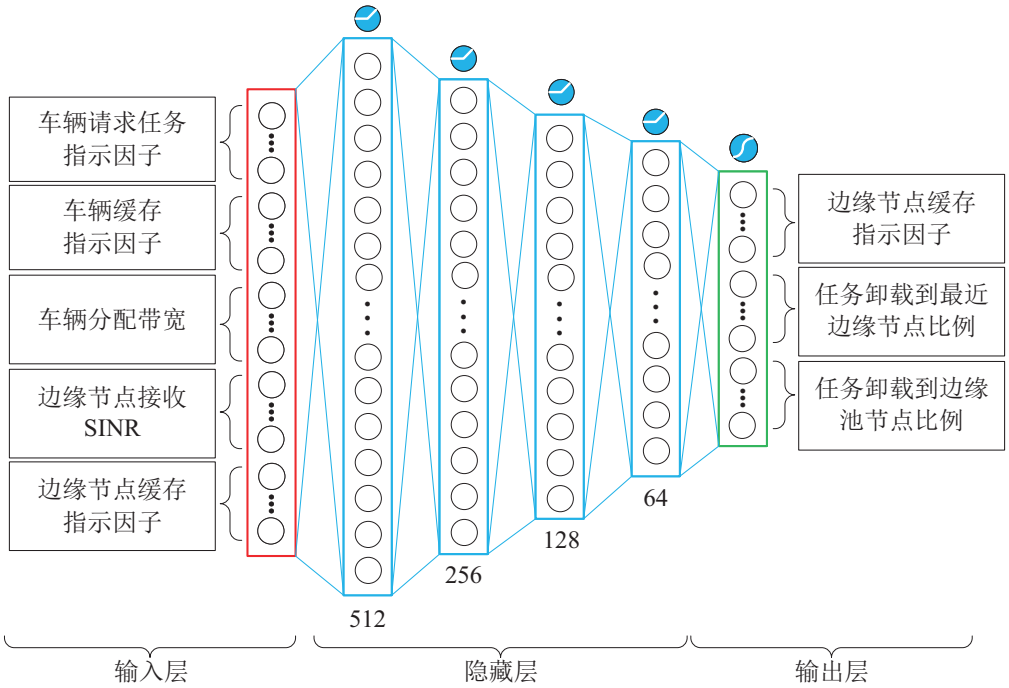


图 3-4 Actor 网络架构

Figure 3-4 Architecture of the Actor network

Actor 网络的结构如图3-4所示，隐藏层有四层网络，其输入层包含：车辆请求任务

指示因子、车辆缓存指示因子、车辆分配带宽、边缘节点接收 SINR 和边缘节点缓存指示因子。输出层为边缘节点缓存指示因子、任务卸载到最近的边缘节点比例和卸载到边缘池其他节点比例。输入层即当前状态 s_t，输出为动作 a_t，其中动作变量 $c_{e,k}^{E}(t)$ 需要进行离散化（$\lceil c_{e,k}^{E}(t) \rceil$，$\lceil x \rceil$ 是向上取整函数 $\lceil x \rceil = \min\{n \in \mathbb{Z} | n \geq x\}$）。

算法 3-1: 基于深度强化学习的服务缓存与计算卸载策略

输入: 折扣因子 γ，主网络 Actor 网络参数 θ^{μ}，目标网络 Actor 网络参数 $\theta^{\mu'}$，主网络 Critic 网络参数 θ^{Q}，目标网络 Critic 网络参数 $\theta^{Q'}$，软更新因子 τ，总训练次数 M，最大时隙 t_{end}，经验池大小 D，小批量样本数量 N

输出: 主网络中 Actor 网络最优参数 θ^{μ}

```
初始化主网络和目标网络权重
初始化经验池
for episode = 1 to M do
    初始化观测状态 s_0
    给 Actor 主网络输出添加高斯噪声 n_t
    for t = 1 to t_end do
        智能体接收归一化的观测状态 s_t
        选择动作 a_t = μ(s_t|θ^μ) + n_t
        执行动作 a_t，计算即时奖励 r_t，并获取下一个归一化状态 s_{t+1}
        if 回放经验池空间剩余 then
            存储元组（s_t, a_t, r_t, s_{t+1}）到经验池 D 中
        else
            随机在经验池选择一个元组，用元组（s_t, a_t, r_t, s_{t+1}）进行替换
            随机冲经验池中抽取 N 个小批量元组样本（s_j, a_j, r_j, s_{j+1}），∀j = 1, 2, ..., N
              from D
            计算目标动作价值 y_j = r(s_j, a_j) + γQ_μ(s_{j+1}, μ(s_{j+1}|θ^{μ'})|θ^{Q'})
            通过式(3.56)最小化损失函数来更新 Critic 主网络参数 θ^Q
            根据式(3.57)策略梯度更新 Actor 主网络参 θ^μ
            根据式(3.58)软更新目标网络参数 θ^{Q'} 和 θ^{μ'}
        end
    end
end
```

在每个时隙 t 的开始，智能体收集环境信息得到当前系统状态 s_t，为了解决确定性

策略的探索问题，在动作空间添加行为噪声 n_t 获得动作 $a_t = \mu(s_t|\theta^\mu) + n_t$，当车辆和边缘节点服务器基于智能体的动作 a_t 做出服务缓存和计算卸载决策，环境给予反馈，观测到下一个状态 s_{t+1} 并得到即时奖励 r_t，此时能够得到元组 (s_t, a_t, r_t, s_{t+1}) 并将其存储在经验回放池中，经验回放的好处在于打破序列的相关性，能够重复利用收集到的经验。算法在回放经验池中随机选择 N 个元组组成小批量训练数据，将其输入到主网络和目标网络中，通过小批量数据，Actor 网络输出动作作为 Critic 网络的输入。主网络的 Critic 网络更新 θ^Q 通过损失函数进行更新：

$$J(\theta^Q) = \frac{1}{N}\sum_{j=1}^{N}(y_j - Q_\mu(s_j, \mu(s_j|\theta^\mu)|\theta^Q))^2 \tag{3.56}$$

主网络中的 Actor 网络参数 θ^μ 通过策略梯度公式进行更新：

$$\bigtriangledown_{\theta^\mu} J(\theta^Q) = \frac{1}{N}\sum_{j=1}^{N}[\bigtriangledown_a Q(s, a|\theta^Q)|_{s=s_j, a=\mu(s_j|\theta^\mu)} \bigtriangledown_{\theta^\mu} \mu(s|\theta^\mu)|_{s=s_j}] \tag{3.57}$$

最后，使用软更新的方式代替复制主网络参数来更新目标网络中的参数：

$$\begin{aligned} \theta^{Q'} &\leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'} \\ \theta^{\mu'} &\leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'} \end{aligned} \tag{3.58}$$

DDPG 算法的主要步骤总结见算法 3-1。

3.4 实验仿真及结果分析

本节对基于深度强化学习的服务缓存与计算卸载策略进行了仿真和分析。首先对仿真场景及参数和仿真方案进行设计，然后从收敛性、不同场景参数影响对所提策略性能进行了验证。

3.4.1 实验参数设置

实验使用 Python3.6 构建提出的服务缓存与计算卸载仿真环境，利用 TensorFlow1.14.0 深度学习框架仿真 DDPG 算法。模拟中每辆车在每个时隙的开始随机请求感兴趣任务，时隙的持续时间保证智能体有足够的时间做出缓存和卸载决策，卸载到其他节点完成任务计算能够在时隙结束前完成，在此前提下，尽可能最小化长期系统平均任务完成时延。实验仿真参数设置见表3-2，该参数参考了已发表的工作 [75，76，165] 。在实验中，从相同参数设置的基于 DDPG 的服务缓存与计算卸载算法多次运行中获得平均累计奖励用于性能比较，与以下基准算法进行对比：

表 3-2 仿真参数设置

Table 3-2 Simulation Parameters

参数	值	参数	值
时隙总数 (t_{end})	40	车辆存储空间（S_v^V）	50 Mb
每个时隙持续时间 Δt	30 s	边缘服务器存储空间（S_e^E）	100 Mb
边缘节点车辆密度范围（ρ）	[2,5]	计算能量效率参数（κ）	1×10^{-26}
边缘节点通信范围	500 m	边缘服务器之间传输速率（R_{edge}）	15 Mbps
边缘节点服务器通信总带宽（B）	20 MHz	边缘服务器到云端传输速率（R_{cloud}）	10 Mbps
SINR	4~5 dB	边缘服务器之间传输功率（p_{edge}）	1 W
边缘节点数量（N_E）	3	边缘服务器到云端传输功率（p_{cloud}）	2 W
服务类型数量（N_K）	5	折扣因子（γ）	0.99
任务输入数据大小（d_k）	20 Mb	Actor 网络学习率（lr_a）	0.001
服务程序存储空间（θ_k）	50 Mb	Critic 网络学习率（lr_c）	0.002
车辆 CPU 计算频率（f^V）	5×10^8 cycles/s	软更新系数（τ）	0.01
边缘服务器 CPU 计算频率（f^E）	1×10^9 cycles/s	小批量样本大小（N）	128

1）无边缘缓存卸载方案（Offloading without edge caching）：车辆任务请求只能在本地或卸载到云端服务器执行计算[172]。

2）基于时延最小化的卸载方案（Offloading based on latency minimization）：车辆任务基于缓存决策情形下的时延最小对应的最优卸载比例进行卸载[76]。

3）基于能量最小化的卸载方案（Offloading based on energy minimization）：车辆任务基于缓存决策情形下的能量消耗最小对应的最优卸载比例进行卸载[168]。

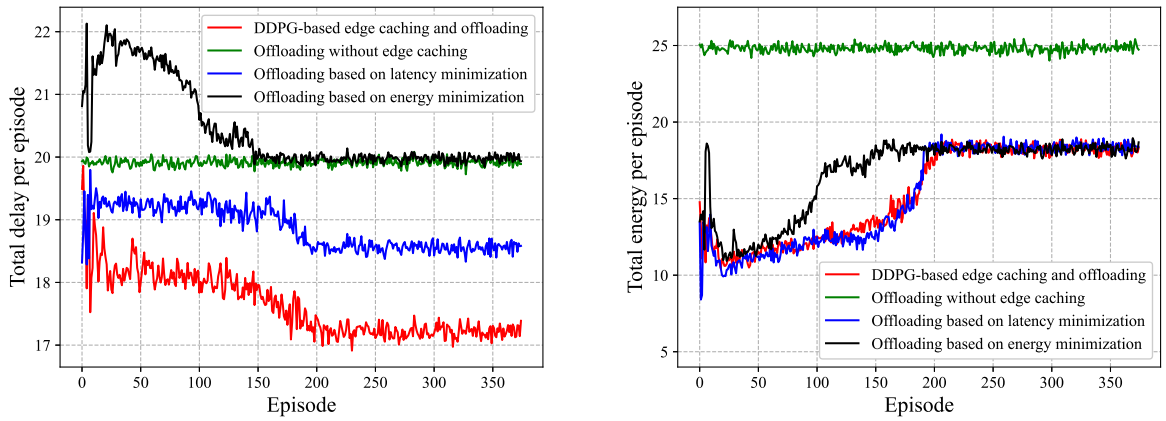
4）随机服务缓存与计算卸载方案:（Random edge caching and offloading）每个时隙内服务缓存决策和任务卸载比例随机确定。

5）最近最少使用缓存与卸载方案（Least recently used (LRU) edge caching and offloading）：缓存新程序时，上一时隙请求过的继续缓存，没有请求的随机进行替换[173]，任务卸载比例根据每种缓存情形下时延最小计算卸载比例。

6）所有任务在云端服务器执行方案（Executing all tasks in the cloud）：所有任务卸载到云端服务器执行计算[167]。

3.4.2 结果分析

本节提出的服务缓存与计算卸载方案的收敛性表现如图3-5所示，图3-5(a)和图3-5(b)分别表示系统总时延和总能耗的收敛情况，需要说明的是优化目标是时延，由于能量消耗的计算与时延相关，所以能耗也呈现出收敛性。随着回合数增加，所有方案的系统总时延持续下降并达到稳定，除了无边缘缓存方案外，基于能量最小化的卸载方案最终收敛到和无边缘缓存方案持平，但无边缘缓存方案的能耗消耗最高，其主要是卸载到云端计算进而导致较高时延和能耗，边缘服务器无法提供计算和缓存资源，同时智能体也无法参与决策，而基于能耗最小化的卸载方案能够通过决策减少能量消耗，且其能耗收敛的速度更快。基于时延最小化的卸载方案并没有针对服务缓存进行优化，只是在每种服务缓存情形下做出最优的卸载决策，而基于 DDPG 的联合服务缓存与计算卸载方案综合考虑其耦合影响，在相同能耗收敛下能够实现最低的系统任务完成时延。



(a) 系统总时延收敛性　　(b) 系统总能耗收敛性

图 3-5 不同策略收敛性分析

Figure 3-5 Convergence analysis of different policies

图3-6表示不同方案边缘节点范围内车辆密度对系统总时延的影响，随着车辆密度 ρ 的增加，任务请求数量增加，分配给每辆车的带宽和上传到边缘节点的传输速率减少，进而导致任务处理总时延增加。当 $\rho = 2,3$ 时，所有任务在云端服务器执行计算的时延低于无边缘缓存的卸载方案，此时云端丰富的计算资源更占优势，随着车辆密度继续增加，任务在云端服务器执行计算的时延高于无边缘缓存的卸载方案，此时传输速率减小，此时传输时延占主导因素。联合服务缓存与计算卸载策略在周期总延时方

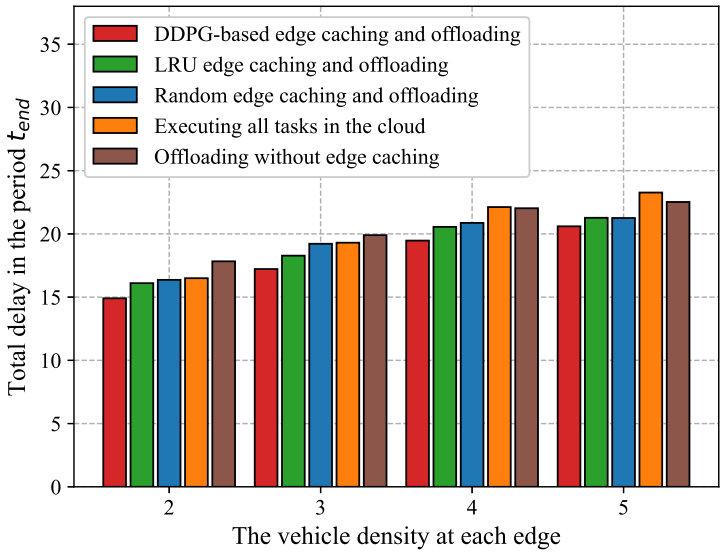


图 3-6 车辆密度 ρ 对总任务处理时延影响

Figure 3-6 The effect of the vehicle density ρ on the total task processing delay

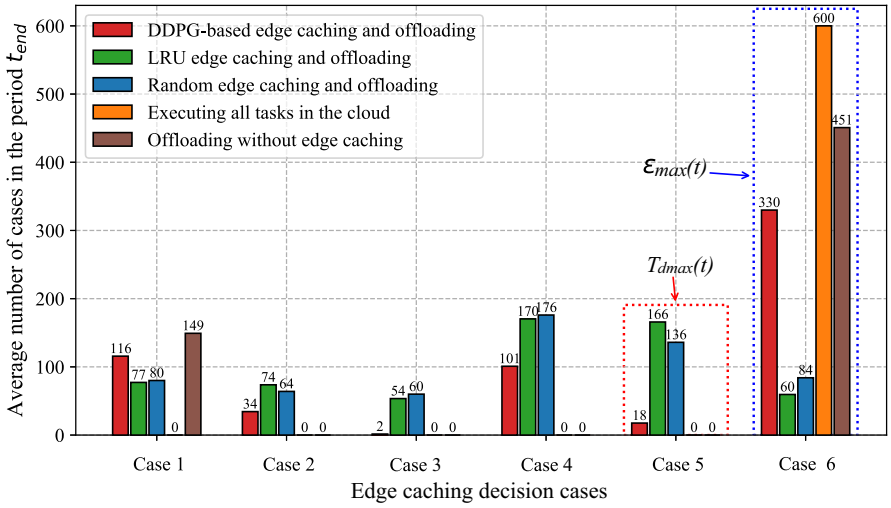


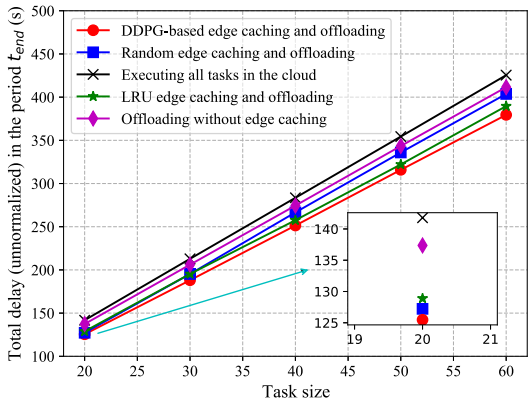
图 3-7 $\rho = 5$ 时，周期 t_{end} 内不同缓存决策情形平均次数

Figure 3-7 The average number of edge caching decision cases in the period t_{end} with $\rho = 5$

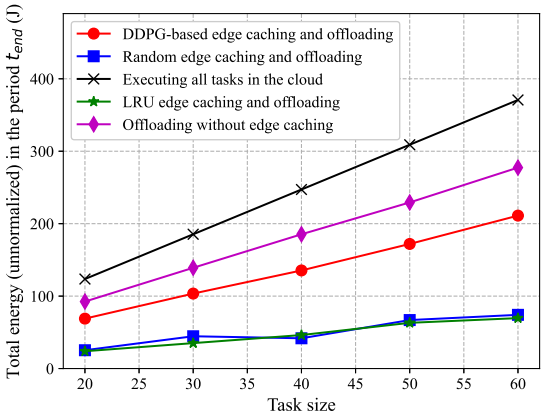
面优于其他方案。

图3-7统计了 $\rho = 5$ 时，不同方案中所有服务缓存决策情形的平均次数，根据仿真参数设置，可以推出 $\max\{T_{v,e,k}^{total}(t)\} = T_d^{Case5}(t)$，$\max\{\varepsilon_{v,e,k}^{total}(t)\} = \varepsilon^{Case6}(t)$。单次时隙内做出缓存情形 5 的决策会产生最大的任务处理时延，做出缓存情形 6 的决策会产生最大的能量消耗。从图中可以看出，基于 DDPG 的缓存与决策方案尽可能避免了时延最大的情形下的缓存决策，这会大大减少任务处理时延。无边缘缓存方案仅做出情形 1 和情形 6 的决策，无边缘节点参与和决策。在云端服务器执行所有任务计算的方案全

部做出情形 6 的决策。图3-8表示任务输入数据大小对系统总时延和总能耗的影响，总



(a) $\rho = 5$ 时，任务大小对系统总时延的影响



(b) $\rho = 5$ 时，任务大小对系统总能耗的影响

图 3-8 $\rho = 5$ 时，任务大小对系统总时延和能耗的影响

Figure 3-8 The total delay and energy (unnormalized) versus task size with $\rho = 5$

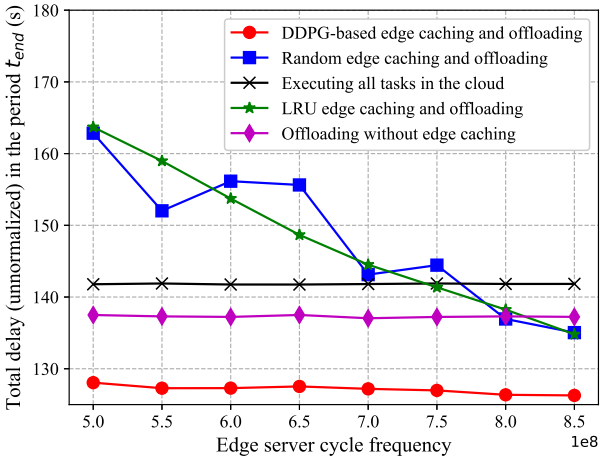


图 3-9 $\rho = 5$ 时，不同边缘服务器计算频率对总时延的影响

Figure 3-9 The total delay (unnormalized) versus edge sever cycle frequency with $\rho = 5$

时延和总能耗随任务输入数据大小呈线性增加，因为时延和能耗的函数与任务大小 d_k 呈正比，任务输入数据大小增加会增加所有方案的传输延迟、计算延迟和能耗，所有任务在云端服务器执行计算的方案产生最高的时延，传输至云端也产生了最高的能耗。从图3-7中也可以得出，由于全部卸载到云端和无边缘缓存方案做出了大量情形 6 的决策，其属于能耗最高的决策，因此导致总能耗的增加，联合服务缓存与计算卸载策略在系统总延时方面优于其他方案，其代价是卸载消耗了较大能量。

图3-9表示边缘服务器计算频率对总时延的影响，由于无边缘缓存方案和全部在云

端执行计算的两个方案不涉及到边缘服务器的计算，与其计算频率无关，所以对总时延无影响。其他方案由于边缘服务器计算能力的增加，时延总体趋势减小。LRU 方案和随机缓存及卸载方案受其影响较大，增加到一定程度，其总时延低于无边缘缓存方案。随机边缘缓存和卸载方案性能不太稳定。联合服务缓存与计算卸载策略保持最低的任务完成时延。

3.5 本章小结

本章提出了车载边缘计算系统中服务缓存与计算卸载场景，具体构建了场景中的通信、缓存、计算卸载模型，详细分析了任务卸载流程中不同缓存情形对应的卸载情况，推导计算、传输延迟以及能量消耗。在此基础上形式化定义了最小化长期累计平均任务时延，进一步将问题描述为马尔可夫决策过程，提出基于深度确定性策略梯度的强化学习算法，设计主网络和目标网络结构，根据动作价值评估智能体动作。最后，进行全面的性能评估，证明所提联合服务缓存与计算卸载策略的优越性。

第四章　面向网联自动驾驶的协同感知任务资源配置优化

4.1 引言

自动驾驶从单车智能技术向车路协同技术演进，为智能交通产业带来机遇，车路协同能够进一步缓解交通拥堵，提升交通效率和安全。车路协同应用第一阶段是以 LTE-V2X 为核心的直连无线通信支持车车、车路直接通信，实现较为基础的消息提醒和安全预警类应用；第二阶段在 C-V2X 车路云高效通信的基础上，以人工智能（Artificial Intelligence，AI）和边缘计算为核心的路端感知技术，发挥路端视角好、观测时间长、容易部署等优势，解决自动驾驶汽车遮挡、盲区、不利照明、极端天气相关感知长尾问题，扩展自动驾驶汽车的感知范围，提升感知能力，进而保障车辆安全高效行驶；第三阶段，在车路协同感知基础上，进一步发挥路端和云端的全局性优势，支持实现协同决策规划和协同控制等应用，全方位保障自动驾驶安全连续运行[174]。目前正处于第二发展阶段，自动驾驶协同感知已发布相关仿真框架和数据集[103, 105, 108]，为后续研究提供数据基础。

单车智能依靠增加传感器提升感知能力、增加运算单元提升车端算力，这会导致自动驾驶汽车成本无限增加，通过车端、路端感知能力和算力的有效分担，突破单车算力极限，实现算力的合理分配和优化均衡。从协同感知的角度看，协同感知任务是多车感知任务的结果融合，感知任务计算可以简化为两个阶段，第一阶段，传感器采集原始数据经过特征提取到特征数据，第二阶段，特征数据经过检测模型识别输出目标数据。自动驾驶的协同感知分为原始数据层级、特征数据层级和目标数据层级，不同层级数据进行融合的隐含假设是两车算力和感知配置相当，然而现实情况下，传感器模态和感知检测模型是异构的，车辆算力配置是不均衡的，L2 级别自动驾驶需要算力 10 TOPS，TOPS 代表处理器每秒进行一万亿次操作。L3 级别需要 30-60 TOPS，L4 级别至少大于 100 TOPS，L5 级别算力至少为 500-1000 TOPS[175]。协同感知任务的数据卸载应该根据车辆算力资源配置选择阶段进行卸载，考虑车辆移动性，任务组合性，延迟要求等选择合适的策略完成协同感知任务的计算卸载。自动驾驶任务卸载的研究受到广泛关注，LIU L 等人[176] 基于车辆移动性分析，设计高效 V2V 多跳卸载方案，建

模为最小化任务执行计算时延和计算成本的加权和，提出自适应调整的半定松弛方法求解该问题，显著改善响应延迟。KU Y 等人考虑频繁上行链路条件、顺序和并行任务的依赖性以及动态驾驶环境未知目标检测输出引起的不确定任务组合，提出一种实时机制，基于动态规划的启发式算法、两阶段任务卸载和调度自适应算法确定多车感知融合应用涉及的任务带宽分配、任务划分、卸载和执行调度，显著减少端到端融合应用延迟[177]。LV P 等人提出自动驾驶车辆环境感知的任务卸载模型，为车辆产生的感知任务分配动态优先级，优化目标为最大化服务器接收任务优先级之和，提高车辆环境感知质量，采用深度强化学习算法进行卸载决策，执行任务队列调度算法确定执行顺序[178]。YANG J 等人提出基于随机几何的交通模型模拟自动驾驶真实交通环境，设计分布式计算卸载框架，旨在最小化处理任务的平均成本，将该问题分解为子问题并利用拉格朗日乘子法优化任务分割比例进行求解[179]。XIAO Z 等人提出协作计算的多层感知任务卸载框架，自动驾驶汽车的移动建模为三阶段过程，根据每个阶段的位置特征识别协同计算的候选共同感知者，在任务依赖性约束下最小化感知延迟，提出两层二进制智能萤火虫算法求解感知任务分配、卸载和资源分配的联合问题[175]。然而上述研究没有考虑到协同感知任务中的传感器模态、检测模型的异构性，考虑车辆移动性、协同感知任务组合性、任务多阶段性的任务卸载策略还未深入研究。

研究网联自动驾驶协同感知任务卸载与资源分配可以更有效地利用网络中各个节点（车辆或服务器）的计算、存储和通信资源。通过协同合作，可以避免资源浪费和重复计算，提高整个系统的资源利用率。通过协同感知任务卸载，可以实现感知数据的共享和集成处理，从而提高感知性能和数据质量。此外，协同感知还可以扩大感知覆盖范围，实现对更广泛区域的感知和监测。车辆和设备通常具有有限的计算能力和能源供应。通过将感知任务卸载到其他节点上进行处理，可以降低本地计算成本和能源消耗，延长车辆的续航里程。协同感知任务卸载和资源分配可以实现分布式并行处理，从而加快感知数据的处理速度和系统的响应速度。这对于自动驾驶系统来说尤为重要，能够更及时地做出决策和应对变化的交通环境。从交通层面看，研究协同感知任务卸载与资源分配不仅有助于优化网联自动驾驶系统的性能，还可以为智能交通系统的发展提供重要的技术支持。

本章研究了基于 VEC 系统的网联自动驾驶协同感知任务的卸载及计算资源分配

问题，考虑协同感知任务的阶段性、协作任务组合性、车辆高移动性和算力资源不均衡性，提出基于协同感知的任务需求和服务模型，推导任务处理时延的表达式，提出最小化任务处理时延的优化问题，设计两层最优任务卸载与资源分配方案，仿真结果证明了该方案的优越性。

本章具体内容安排如下：4.1 节是本章的引言，介绍了车载边缘计算系统中协同感知任务卸载的研究现状及存在不足，同时阐述本章的主要贡献。4.2 节阐述了协同感知任务卸载模型并定义了优化问题。4.3 节设计了双层任务卸载与资源分配算法。4.4 节搭建实验仿真模型并进行了性能验证。4.5 节总结了本章的研究工作。

4.2 协同感知任务卸载模型

4.2.1 系统模型

本章提出针对网联自动驾驶车辆协同感知任务的卸载与资源分配场景，如图4-1所示，车辆、边缘计算服务器具有不同的计算资源，由于涉及 V2V 卸载，将车辆划分为任务车辆和服务车辆，对于自主（Ego）车辆，其感知范围的车辆的感知信息能够进一步帮助自主车辆扩大其感知范围，并消除遮挡或填补盲区等，所以将自主车辆及其感知范围内的车辆定义为感兴趣域（Regions of Interest，RoI）任务车辆，见图中白色车辆。自主车辆表示为 v_{ego}，其传感器感知范围内的车辆集合表示为 C_{ego}^{sensor}。服务车辆即有空闲计算资源为任务需求车辆提供计算服务的车辆，仅考虑提供单跳的服务车辆，将位置处于自车车辆感知范围外和最大 V2V 通信范围内的车辆定义为 RoI 服务车辆，见图中黑色车辆。路侧单元与车载边缘服务器连接，路侧单元具备感知和通信能力，车载边缘服务器提供计算能力，路侧单元用符号 r 表示。RoI 任务车辆集 Ω_v^{RoI} 定义为

$$\Omega_v^{RoI} = \{v_i; \forall v_i \in C_{ego}^{sensor} \cap C_{ego}^{V2V} \cup v_{ego}\} \tag{4.1}$$

RoI 服务车辆集 Ω_{server}^{RoI} 定义为

$$\Omega_{server}^{RoI} = \{v_j; \forall v_j \notin \Omega_v^{RoI} \cap C_{v_i}^{V2V} \cap C_{ego}^{V2V}, v_i \in \Omega_v^{RoI}\} \cup \{r\} \tag{4.2}$$

RoI 服务车辆是基于 RoI 任务车辆进行定义，其中 $C_{v_i}^{V2V} \cap C_{ego}^{V2V}$ 表示服务车辆既应该处于任务车辆范围内提供卸载模式又应该处于自主车辆通信范围内返回计算结果。

对于 RoI 任务车辆集合，从传感器采集原始数据 I_{raw}，经过第一阶段特征提取得

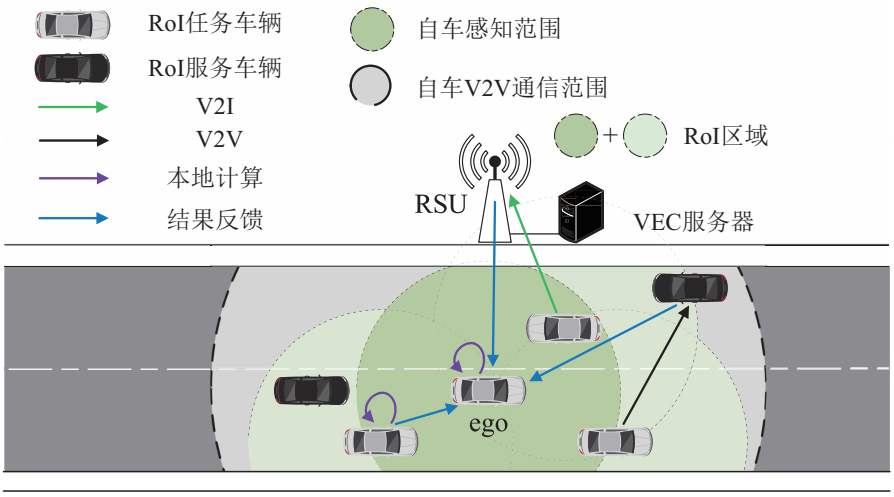


图 4-1 车载边缘计算系统协同感知任务卸载场景

Figure 4-1 Scenario for cooperative perception task offloading in vehicular edge computing systems

到特征数据 I_{fea}，经过第二阶段检测模型输出目标数据 I_{obj}，由于车辆算力资源不均衡，车辆可以在任意阶段进行卸载，车辆 v_i 采集原始数据执行第一阶段要求的计算密度表示为 $\lambda_{v_i}^{raw}$，随后得到特征数据执行第二阶段要求的计算密度为 $\lambda_{v_i}^{fea}$。任务车辆的计算资源表示为 $f_{v_i}, v_i \in \Omega_v^{RoI}$，服务节点（RoI 服务车辆和 RSU）的计算资源表示为 $f_{s_j}, s_j \in \Omega_{server}^{RoI}, s_0 = r$，$s_0$ 表示为 RSU。

4.2.2 计算卸载模型

由于任务车辆集合的算力资源不均衡或动态变化，任务车辆之间互相协作完成协同感知任务，进而扩大自主车辆的感知范围，消除遮挡或填补盲区，其对任务完成计算的实时性要求很高，车辆可以进行卸载借助服务车辆或 RSU 空间丰富的算力资源完成计算，RoI 任务车辆需要决策任务数据在哪个阶段进行卸载，卸载到服务节点中的哪个车辆或 RSU，服务节点的算力资源如何分配是本章需要解决的主要挑战。

为了描述车辆 v_i 的任务数据卸载处于哪个阶段，引入变量 $x_i \in \{0,1,2\}$ 表示，$x_i = 0$ 表示车辆任务的第一第二阶段均在本地执行，输出感知最终结果。$x_i = 1$ 表示任务的第一阶段在本地执行，即将特征数据进行卸载。$x_i = 2$ 表示不执行任何阶段，直接卸载传感器采集的原始数据。使用二进制变量 $o_{ij} \in \{0,1\}$ 作为卸载指示因子，$o_{ij} = 1$ 表示车辆 v_i 卸载任务数据到服务节点 s_j，$o_{ij} = 0$ 表示未进行任务卸载。任务卸载到服务节点后，变量 $\mu_{ij} \in [0,1]$ 表示服务节点 s_j 对车辆 v_i 卸载的任务分配的计算资源比例。接下来对自主车辆 RoI 任务完成时延进行推导。

任务车辆 v_i 选择 $x_i=0$，即任务所有阶段均在本地执行计算，随后将计算结果返回自主车辆。任务完成时延表示为 $D_{v_i}(x_i=0)$，如下式

$$D_{v_i}(x_i=0)=\begin{cases}\frac{I_{raw}\lambda_{v_i}^{raw}+I_{fea}\lambda_{v_i}^{fea}}{f_{v_i}}, & v_i=v_{ego}\\ \frac{I_{raw}\lambda_{v_i}^{raw}+I_{fea}\lambda_{v_i}^{fea}}{f_{v_i}}+\frac{I_{obj}}{R_{v_i\to ego}}, & v_i\in\Omega_v^{RoI},v_i\neq v_{ego}\end{cases} \tag{4.3}$$

其中 v_i 分为自主车辆情况和非自主车辆情况，自主车辆情况无需进行计算结果回传，$R_{v_i\to ego}$ 表示车辆 v_i 与自主车辆之间的传输速率。

任务车辆 v_i 选择 $x_i=1$，即任务的第一阶段在本地执行，输出特征数据，第二阶段任务执行则卸载到服务节点 s_j，任务完成计算的总时延表示为

$$D_{v_i}(x_i=1,o_{ij},\mu_{ij})=\frac{I_{raw}\lambda_{v_i}^{raw}}{f_{v_i}}+\sum_{j=1}^{|\Omega_{server}^{RoI}|}o_{i,j}\left(\frac{I_{fea}}{R_{v_i\to s_j}}+\frac{I_{fea}\lambda_{v_i}^{fea}}{\mu_{ij}f_{s_j}}+\frac{I_{obj}}{R_{s_j\to ego}}\right), \tag{4.4}$$
$$v_i\in\Omega_v^{RoI},s_j\in\Omega_{server}^{RoI}$$

考虑任务的不可分割性，任务只能选择一个服务节点进行卸载，满足以下约束

$$\sum_{j=1}^{|\Omega_{server}^{RoI}|}o_{i,j}=1,\forall v_i\in\Omega_v^{RoI},\forall s_j\in\Omega_{server}^{RoI} \tag{4.5}$$

同时，服务节点计算资源分配满足以下约束

$$\sum_{i=1}^{|\Omega_v^{RoI}|}\mu_{ij}\le 1,v_i\in\Omega_v^{RoI},\forall s_j\in\Omega_{server}^{RoI} \tag{4.6}$$

任务车辆 v_i 选择 $x_i=2$，即任务原始数据卸载到服务节点 s_j，在服务节点完成任务的第一阶段和第二阶段计算，随后返回计算结果，任务完成计算的总时延表示为

$$D_{v_i}(x_i=2,o_{ij},\mu_{ij})=\frac{I_{raw}}{R_{v_i\to s_j}}+\sum_{j=1}^{|\Omega_{server}^{RoI}|}o_{i,j}\left(\frac{I_{raw}\lambda_{v_i}^{raw}+I_{fea}\lambda_{v_i}^{fea}}{\mu_{ij}f_{s_j}}+\frac{I_{obj}}{R_{s_j\to ego}}\right), \tag{4.7}$$
$$v_i\in\Omega_v^{RoI},s_j\in\Omega_{server}^{RoI}$$

任务车辆 v_i 与服务节点 s_j 之间的传输速率根据香农公式表示如下

$$R_{v_i\to s_j}=Blog_2\left(1+\frac{p|h_{ij}|^2\left(\sqrt{d_{ij}^2+(H_j-l)^2}\right)^{-\alpha}}{\sigma_j^2+\sum_{v_k\in C_{s_j}^{V2I/V2V}\setminus v_i}ph_{kj}}\right) \tag{4.8}$$

其中 $v_k\in C_{s_j}^{V2I/V2V}$ 表示与服务节点 s_j 通信的车辆集合，B 代表分配的总带宽，参数

p 和 σ_j^2 分别表示车辆的传输功率和噪声功率。h_{ij} 代表从车辆 v_i 到服务节点 s_j 的信道衰落系数，α 是路径损失指数，d_{ij} 是当前车辆 v_i 和服务节点 s_j 之间的距离，H_j 表示服务节点 s_j 的天线高度，l 表示车辆的天线高度。

任务车辆将数据卸载后，服务节点的计算及结果返回时间不得超过 V2V 或 V2I 的通信持续时间，为了简化问题，将车辆 v_i 的位置定义为 p_i，仅考虑 x 轴向的坐标值，其速度为 ω_i，从左到右行驶则为正直，相反方向行驶则为负值。RSU 的 V2I 通信覆盖半径为 R_{V2I}，RSU 到道路中心的垂直距离为 e，则任务车辆 v_i 驶出 RSU 的覆盖范围的剩余距离定义为 L_i，如下

$$L_i = \sqrt{R_{V2I}^2 - e^2} - \frac{w_i}{|w_i|} p_i \tag{4.9}$$

假设车辆之间维持通信的最大距离固定为 R_{V2V}，则车辆 v_i 与服务节点之间的通信持续时间定义为 τ_{ij}^{hold}，如下

$$\tau_{ij}^{hold} = \begin{cases} \frac{L_i}{|w_i|}, & j = 0 \\ \left| \frac{R_{V2V} - d_{ij}}{w_j - w_i} \right|, & \frac{p_i - p_j}{w_i - w_j} > 0, j \neq 0 \\ \left| \frac{2R_{V2V} - d_{ij}}{w_j - w_i} \right|, & \frac{p_i - p_j}{w_i - w_j} < 0, j \neq 0 \end{cases} \tag{4.10}$$

其中 $j = 0$ 表示处于 RSU 的覆盖范围内，下面两种情形分别表示车辆同向行驶和反向行驶的情况。任务车辆 v_i 的最大任务容忍时延为 τ_i^{tole}，因此，其任务计算完成时延满足约束

$$D_{v_i}(x_i, o_{ij}, \mu_{ij}) \leq min\{\tau_i^{tole}, \tau_{j,ego}^{hold}\} \tag{4.11}$$

自主车辆的 RoI 任务完成时延计算如下

$$D_{ego}^{RoI} = max\{D_{v_i}\}, v_i \in \Omega_v^{RoI}, s_j \in \Omega_{server}^{RoI} \tag{4.12}$$

4.2.3 优化问题描述

协同感知任务卸载涉及任务阶段选择、服务节点选择以及服务节点资源分配，最终目标是最小化协作计算任务的完成时延，优化问题表述如下。

$$\underset{\{x_i, o_{ij}, \mu_{ij}\}}{\text{minimize}} \; D_{ego}^{RoI} \tag{4.13}$$

$$\text{subject to } D_{v_i}(x_i, o_{ij}, \mu_{ij}) \leq min\{\tau_i^{tole}, \tau_{j,ego}^{hold}\} \tag{4.14}$$

$$\sum_{j=1}^{|\Omega_{server}^{RoI}|} o_{i,j} = 1, v_i \in \Omega_v^{RoI}, s_j \in \Omega_{server}^{RoI} \tag{4.15}$$

$$\sum_{i=1}^{|\Omega_v^{RoI}|} \mu_{ij} \leq 1, v_i \in \Omega_v^{RoI}, s_j \in \Omega_{server}^{RoI} \tag{4.16}$$

$$x_i \in \{0, 1, 2\} \tag{4.17}$$

$$o_{ij} \in \{0, 1\} \tag{4.18}$$

$$\mu_{ij} \in [0, 1] \tag{4.19}$$

约束(4.14)表示任务时延约束，约束(4.15)表示允许卸载到一个服务节点，约束(4.16)表示服务节点的计算资源分配比例和不超过 1。由于变量 x_i 和 o_{ij} 为整数变量，μ_{ij} 为连续变量，且目标函数为 $max\{D_{v_i}\}$ 是非线性函数，故该优化问题是混合整数非线性规划问题。

4.3 双层最优任务卸载与资源分配算法设计

优化问题中 x_i 和 o_{ij} 是离散变量，μ_{ij} 为连续变量，变量之间有较强的关联性，通常为 NP 难问题，常用分支定界法、外界近似法、启发式算法等求解。本节先对离散变量进行遍历，固定离散变量后，该问题变为非线性规划函数，降低了其非凸性，因此可以采用基于梯度的优化算法进行求解。当离散变量确定时，约束(4.14)和(4.16)能够对变量 μ_{ij} 的范围进行一定的约束。在求解之前，任务阶段的所有决策变量 x_i 用矩阵表示 $\mathbf{X}$，其维度为 $|\Omega_v^{RoI}|\times 1$。所有卸载决策变量 o_{ij} 用矩阵 $\mathbf{O}$ 表示，其维度为 $|\Omega_v^{RoI}|\times|\Omega_{server}^{RoI}|$，即每辆任务车可以向任意服务节点进行卸载，但由于约束(4.15)的存在，只允许卸载到其中一个节点。所有资源分配变量 μ_{ij} 用矩阵 $\mathbf{U}$ 表示，其维度为 $|\Omega_v^{RoI}|\times|\Omega_{server}^{RoI}|$，与任务卸载矩阵 $\mathbf{O}$ 维度相同，且有对应的关系，即卸载到相应的服务节点才会进行资源分配，由于对应关系的存在，当离散变量固定后，其会对连续变量矩阵 $\mathbf{O}$ 有相应的约束关系，会影响到连续变量的可行域，因此在固定离散变量后，根据约束(4.14)和(4.16)进行可行域检查是十分必要的。

最优任务卸载与资源分配算法的流程见算法 4-1，首先遍历所有离散变量的组合形式，其搜索复杂度为 $M = (1 + 2\left|\Omega_{server}^{RoI}\right|)^{|\Omega_v^{RoI}|}$。确定整数变量后并不一定满足对应关系，例如目标级数据卸载则无需进行卸载，而原始级和特征级数据数据卸载则确定了卸载变量的数值，但不确定卸载的目标。对不同形式的任务阶段和卸载组合根据约束(4.14)和(4.16)进行可行域检查，主要目的是找到一个可行的初值 $\mathbf{O}$，

算法 4-1: 最优任务卸载与资源分配算法

输入: 任务车辆集 Ω_v^{RoI}，服务车辆集 Ω_{server}^{RoI}，车辆计算频率 f_{v_i}，服务节点计算频率 f_{r_j}，V2V 传输速率 R_{V2V}，V2I 传输速率 R_{V2I}，任务容忍时延 τ_i^{tole}

输出: 任务卸载决策 $\mathbf{X}^*, \mathbf{O}^*$ 和资源分配决策 $\mathbf{U}^*$

/* **遍历或启发式算法** */

遍历离散变量所有组合形式 $\mathbf{X}, \mathbf{O}$，总遍历次数 $M = (1+2\left|\Omega_{server}^{RoI}\right|)^{\left|\Omega_v^{RoI}\right|}$

for $m = 1$ **to** M **do**

  /* **可行域检查** */

  for $i = 1$ **to** $\left|\Omega_v^{RoI}\right|$ **do**

    for $j = 1$ **to** $\left|\Omega_{server}^{RoI}\right|$ **do**

      if $x_i = 0, D_{v_i}(0) \le \min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\}$ **then**

        继续

      else if $x_i = 1, o_{ij} = 1, \frac{I_{raw}\lambda_{v_i}^{raw}}{f_{v_i}} + \frac{I_{fea}}{R_{v_i \to s_j}} + \frac{I_{obj}}{R_{s_j \to ego}} < \min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\}$ **then**

        $\mu_{ij} = \frac{I_{fea}\lambda_{v_i}^{fea}}{f_{s_j}} / (\min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\} - \frac{I_{raw}\lambda_{v_i}^{raw}}{f_{v_i}} - \frac{I_{fea}}{R_{v_i \to s_j}} - \frac{I_{obj}}{R_{s_j \to ego}})$

      else if $x_i = 2, o_{ij} = 1, \frac{I_{raw}}{R_{v_i \to s_j}} + \frac{I_{obj}}{R_{s_j \to ego}} < \min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\}$ **then**

        $\mu_{ij} = \frac{I_{raw}\lambda_{v_i}^{raw} + I_{fea}\lambda_{v_i}^{fea}}{f_{s_j}\left(\min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\} - \frac{I_{raw}}{R_{v_i \to s_j}} - \frac{I_{obj}}{R_{s_j \to ego}}\right)}$

      else 跳转到算法第三行;

    end

  end

  for $j = 1$ **to** $\left|\Omega_{server}^{RoI}\right|$ **do**

    if $\sum_{i=1}^{\left|\Omega_v^{RoI}\right|} \mu_{ij} \le 1$ **then**

      继续

    else

      跳转到算法第三行

    end

  end

  /***NLP 求解器** */

  基于 $\mathbf{X}_m, \mathbf{O}_m$ 生成满足可行域的初始值 $\mathbf{U}_0$

  定义不等式约束

  通过 NLP 优化器求解最优 $\mathbf{U}_m$

end

获得最优卸载决策 $\mathbf{X}^*, \mathbf{O}^*$ 和资源分配决策 $\mathbf{U}^*$

当 $x_i=0, D_{v_i}(0)\le \min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\}$ 时，满足约束则继续。若 $x_i=1, o_{ij}=1$, 此时应该满足

$$\begin{aligned} D_{v_i}(x_i=1,o_{ij}=1,\mu_{ij}) &= \frac{I_{raw}\lambda_{v_i}^{raw}}{f_{v_i}} + \sum_{j=1}^{|\Omega_{server}^{RoI}|} o_{i,j}\left(\frac{I_{fea}}{R_{v_i\to s_j}} + \frac{I_{fea}\lambda_{v_i}^{fea}}{\mu_{ij}f_{s_j}} + \frac{I_{obj}}{R_{s_j\to ego}}\right) \\ &= \frac{I_{raw}\lambda_{v_i}^{raw}}{f_{v_i}} + \frac{I_{fea}}{R_{v_i\to s_j}} + \frac{I_{fea}\lambda_{v_i}^{fea}}{\mu_{ij}f_{s_j}} + \frac{I_{obj}}{R_{s_j\to ego}} \\ &\le \min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\} \end{aligned} \tag{4.20}$$

这里带入了约束(4.15)，进而消掉了求和符号，得到 μ_{ij} 的上限值如下

$$\mu_{ij} \le \frac{I_{fea}\lambda_{v_i}^{fea}}{f_{s_j}} / (\min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\} - \frac{I_{raw}\lambda_{v_i}^{raw}}{f_{v_i}} - \frac{I_{fea}}{R_{v_i\to s_j}} - \frac{I_{obj}}{R_{s_j\to ego}}) \tag{4.21}$$

若 $x_i=2, o_{ij}=1, \frac{I_{raw}}{R_{v_i\to s_j}} + \frac{I_{obj}}{R_{s_j\to ego}} \le \min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\}$ 时，此时应满足

$$\begin{aligned} D_{v_i}(x_i=2,o_{ij},\mu_{ij}) &= \frac{I_{raw}}{R_{v_i\to s_j}} + \sum_{j=1}^{|\Omega_{server}^{RoI}|} o_{i,j}\left(\frac{I_{raw}\lambda_{v_i}^{raw} + I_{fea}\lambda_{v_i}^{fea}}{\mu_{ij}f_{s_j}} + \frac{I_{obj}}{R_{s_j\to ego}}\right) \\ &= \frac{I_{raw}}{R_{v_i\to s_j}} + \frac{I_{raw}\lambda_{v_i}^{raw} + I_{fea}\lambda_{v_i}^{fea}}{\mu_{ij}f_{s_j}} + \frac{I_{obj}}{R_{s_j\to ego}} \\ &\le \min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\} \end{aligned} \tag{4.22}$$

同理，带入约束(4.15)，进而消掉了求和符号，得到 μ_{ij} 的上限值如下

$$\mu_{ij} \le \frac{I_{raw}\lambda_{v_i}^{raw} + I_{fea}\lambda_{v_i}^{fea}}{f_{s_j}\left(\min\{\tau_0^{tole}, \tau_{j,ego}^{hold}\} - \frac{I_{raw}}{R_{v_i\to s_j}} - \frac{I_{obj}}{R_{s_j\to ego}}\right)} \tag{4.23}$$

经过约束(4.14)和(4.15)确定资源分配变量的 μ_{ij} 的上限值，因此还需要判断是否满足约束(4.16)，在算法第 17 行进行判断，若不满足，则说明此时的离散变量方案无满足可行域的连续变量值。若所有离散变量均满足，则进入到对连续变量的求解。当确定离散变量后，优化问题转换为

$$\min_{\{\mu_{ij}\}} max\{D_{v_i}(\mu_{ij})\} \tag{4.24}$$

$$s.t.\ D_{v_i}(\mu_{ij}) \le min\{\tau_i^{tole}, \tau_{j,ego}^{hold}\}, \tag{4.25}$$

$$\sum_{j=1}^{|\Omega_{server}^{RoI}|} o_{i,j} = 1, v_i \in \Omega_v^{RoI}, s_j \in \Omega_{server}^{RoI} \tag{4.26}$$

$$\sum_{i=1}^{|\Omega_v^{RoI}|} \mu_{ij} \leq 1, v_i \in \Omega_v^{RoI}, s_j \in \Omega_{server}^{RoI} \tag{4.27}$$

$$\mu_{ij} \in [0,1] \tag{4.28}$$

将式(4.21)和式(4.23)带入可进一步转换为

$$\min_{\{\mu_{ij}\}} max\{D_{v_i}(\mu_{ij})\} \tag{4.29}$$

$$s.t.\ (4.21), x_i = 1, o_{ij} = 1, v_i \in \Omega_v^{RoI}, s_j \in \Omega_{server}^{RoI} \tag{4.30}$$

$$(4.23), x_i = 2, o_{ij} = 1, v_i \in \Omega_v^{RoI}, s_j \in \Omega_{server}^{RoI} \tag{4.31}$$

$$\sum_{i=1}^{|\Omega_v^{RoI}|} \mu_{ij} \leq 1, v_i \in \Omega_v^{RoI}, s_j \in \Omega_{server}^{RoI} \tag{4.32}$$

$$\mu_{ij} \in [0,1] \tag{4.33}$$

此时问题简化为只有不等式约束的非线性优化问题, 不等式约束的个数为 $K = \left|\Omega_v^{RoI}\right| + \left|\Omega_{server}^{RoI}\right|$，为了简化，不等式约束用 g_k 表示，则优化问题表示为

$$\min_{\{\mu_{ij}\}} f(\mu_{ij}) = max\{D_{v_i}(\mu_{ij})\} \tag{4.34}$$

$$s.t.\ g_k \leq 0, k = 1, ...K. \tag{4.35}$$

其拉格朗日函数为

$$\mathcal{L}(\mu_{ij}, q) = f(\mu_{ij}) + \sum_{k=1}^{K} q_k g_k \tag{4.36}$$

若存在唯一 $\mathbf{q}^*$ 且满足

$$\nabla_{\mu_{ij}} \mathcal{L}(\mu_{ij}^*, q^*) = 0, \tag{4.37}$$

$$q_k^* \geq 0, k = 1, ...K \tag{4.38}$$

$$q_k^* g_k(\mu_{ij}^*) = 0, k = 1, ...K \tag{4.39}$$

$$g_k(\mu_{ij}^*) \leq 0, k = 1, ...K \tag{4.40}$$

$$\mathbf{y}^t \nabla_{\mu_{ij}\mu_{ij}} \mathcal{L}(\mu_{ij}^*)\mathbf{y}, \forall \mathbf{y} \text{ orthogonal to } \nabla_{\mu_{ij}} g(\mu_{ij}^*) \tag{4.41}$$

其为优化问题的 KKT 条件，满足该条件即找到最优资源配置 μ_{ij}^*，通过非线性优化求解器（SciPy，fmincon）进行求解。在可行域检查部分的复杂度为 $O(M\left|\Omega_v^{RoI}\right|\left|\Omega_{server}^{RoI}\right|)$, 非线性求解器的算法复杂度为 $O(K)$, 算法总体复杂度为 $O(M\left|\Omega_v^{RoI}\right|\left|\Omega_{server}^{RoI}\right|K)$。协同感知任务主要解决遮挡和远距离感知需求，自动驾驶车辆协同往往只需要关键性的遮挡信息，协同的数量越多，会造成大量的信息冗余，与感兴趣域内的少数关键车辆协

作，其复杂度不会很高，通过遍历寻找最优整数变量的计算复杂度处于可接受范围之内。

4.4 实验仿真及结果分析

本节对多车协同感知任务卸载策略进行仿真和分析。首先对仿真场景及参数和仿真方案进行设计，然后从不同场景参数影响对所提任务卸载与资源分配策略性能进行了验证。

4.4.1 实验参数设置

实验使用 Python3.6 构建仿真场景，考虑一个双向四车道道路，车道宽度为 3.75 米，RSU 到道路中心线的距离为 100 米，每个车辆配备不同的传感器，比如摄像头和激光雷达等。原始数据、特征数据和目标数据的平均大小设定为 300 Mbit，150 Mbit 和 1 Mbit[108]。每个任务的容忍时延 τ 设定为 0.5 s，其他参数设置见表4-1，表中参数参考了已发表工作 [175，176，178] 。在实验中，自主车辆的 RoI 任务完成时延经过多次运行取平均值用于性能比较，本文提出的最优任务卸载与资源分配算法（Optimal Task Offloading and Resource Allocation，OTORA）与以下基准算法进行对比：

1）目标级数据卸载方案（Object Data Offloading （ODO）scheme）：所有任务车辆卸载目标层级数据，即所有车辆的原始数据均在本地执行计算[86]。

2）特征级数据卸载方案（Feature Data Offloading（FDO）scheme）：所有任务车辆卸载特征级数据，即任务的第一阶段在车辆本地执行计算，第二阶段的卸载决策和资源分配与 OTORA 方案相同。

3）原始级数据卸载方案（Raw Data Offloading （RDO）scheme）：所有任务车辆卸载原始级别数据，即任务车辆不在本地执行任何计算，其卸载决策和资源分配方案与 OTORA 方案相同[175]。

4）贪婪任务卸载方案（Greedy Task Offloading (GTO) scheme）：任务数据优先卸载到算力资源较大的服务节点进行计算，资源分配方案与 OTORA 方案相同[180]。

5)OTORA 资源均等分配方案(OTORA Equal Computing Resource Allocation (ECRA) scheme)：任务车辆进行任务卸载后，其计算资源均等分配，卸载策略与 OTORA 方案相同[181]。

表 4-1 仿真参数设置

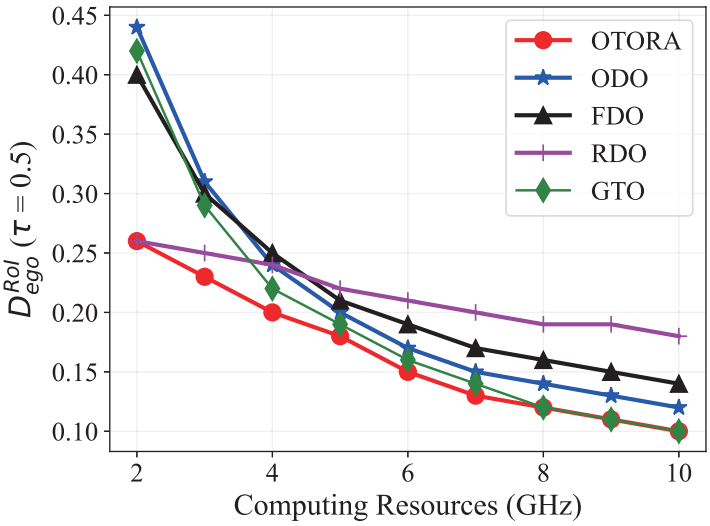
Table 4-1 Simulation Parameters

参数	值	参数	值	参数	值
R_{V2V}	70 m	λ_{raw}	[1800, 2000] cycles/bit	B	20 MHz
R_{V2I}	200 m	λ_{fea}	[1500, 2000] cycles/bit	σ_j	5×10^{-10}
p	0.2 W	f_{v_j}	[2, 8] GHz	α	4
H_{rsu}	20 m	f_{rsu}	40 GHz	ω_i	[50, 60] km/h

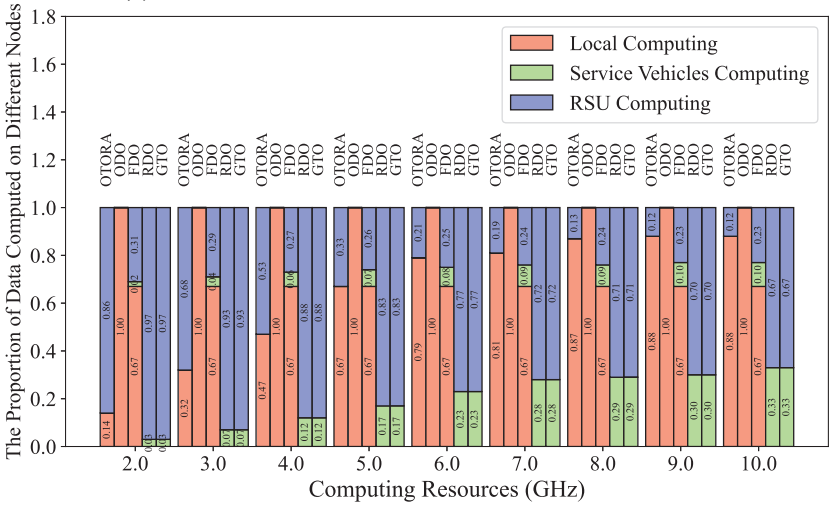
4.4.2 结果分析

图4-2(a)表示随车辆计算资源变化，对不同任务卸载与资源分配方案下 RoI 任务处理时延的影响，参数设置为 $\left|\Omega_v^{RoI}\right|=4, \left|\Omega_{server}^{RoI}\right|=5, \tau=0.5$ s。随着车载计算资源的增加，所有任务卸载与资源分配方案的任务处理时延均减少，因为任务车辆和服务车辆的算力均增加，进而引起整体的计算时延降低。当车辆算力资源较小时（小于 4 GHz），OTORA 方案和 RDO 方案的完成任务时延较低，从图4-2(b)表示的任务数据在不同节点计算的比例可以看出，此时的原因在于，OTORA 和 RDO 方案的数据比例大部分在算力资源丰富的 RSU 处完成计算，而依靠本地计算的 ODO 和 FDO 方案此时完成任务时延较高。随着任务车辆和服务车辆的计算资源增加，OTORA 方案始终保持最优的任务完成时延，其计算数据分配在本地和 RSU，并且随着算力的增加，分配给车辆的占比增加。ODO 方案一直保持在本地计算，车辆算力增加后能保持较低的完成时延。对于 RDO 方案，由于自车的算力资源无法充分利用，随着算力资源增加，其卸载到服务车辆的数据比例逐渐增加，但仍旧保持较高时延。GTO 方案倾向选择较大算力资源的服务节点，未考虑与数据类型选择的联合优化，其性能优于 ODO 和 FDO 方案，与 OTORA 方案还有差距，原因在于其本地计算的数据占比高于 OTORA 方案。

图4-3(a)表示任务车辆数量增加，对不同任务卸载与资源分配方案下 RoI 任务处理时延的影响，参数设置为 $\left|\Omega_{server}^{RoI}\right|=5, \tau=0.5$ s。随着任务车辆数量增加，对算力的需求增加，而总的算力资源有限，每个任务分摊的平均算力资源减少，故 RoI 任务处理时延增加。图4-3(b)表示随任务车辆增加，任务数据在不同节点的计算比例，OTORA 方案随着车辆数增加，RSU 计算比例占比逐渐减少，车辆本地计算的占比增加，RSU 能够分配给每个任务的资源减少，而任务车辆和服务车辆的算力资源较小，导致 OTORA



(a) 车辆计算资源增长对 RoI 任务处理时延的影响



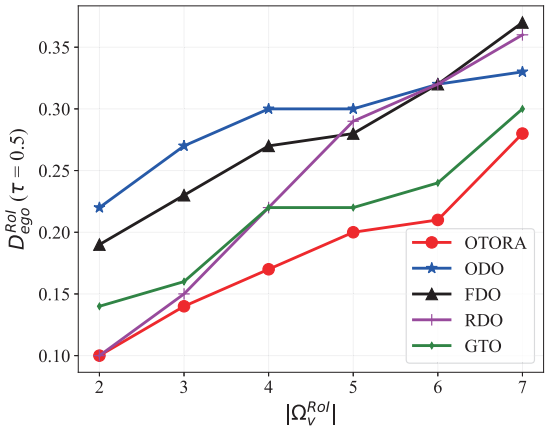
(b) 车辆计算资源增长对不同节点计算数据比例影响

图 4-2 车辆计算资源增长对任务处理时延和数据计算比例的影响

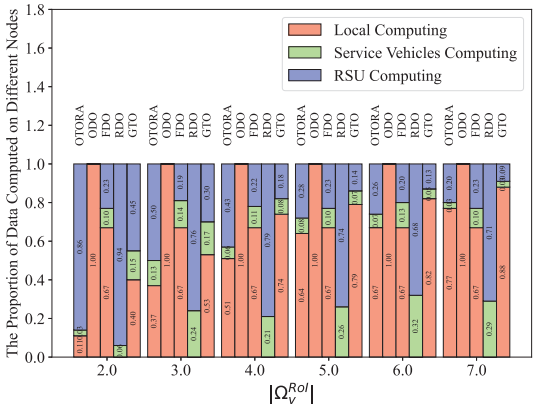
Figure 4-2 Effect of vehicle computing resource increase on (a) RoI task processing delay and (b) data computation proportions among various nodes ($\left|\Omega_v^{RoI}\right| = 4, \left|\Omega_{server}^{RoI}\right| = 5$)

方案任务处理时延增加。ODO 方案无法利用 RSU 和服务车辆的算力资源，因此保持较高的任务完成时延。RDO 方案随着任务车辆数增加，自身的算力没有充分利用，卸载到服务车辆进行计算增加了传输时间。

图4-4(a)表示 RSU 的计算资源增加，对不同任务卸载与资源分配方案下 RoI 任务处理时延的影响，参数设置为 $\left|\Omega_v^{RoI}\right| = 4, \left|\Omega_{server}^{RoI}\right| = 1, \tau = 0.5$ s。随着 RSU 的计算资源增加，由于车辆的计算资源远小于 RSU 的计算资源，所有任务卸载与资源分配方案尽可能利用 RSU 的计算资源，分配到每个任务的算力资源增加，RoI 任务完成时延逐



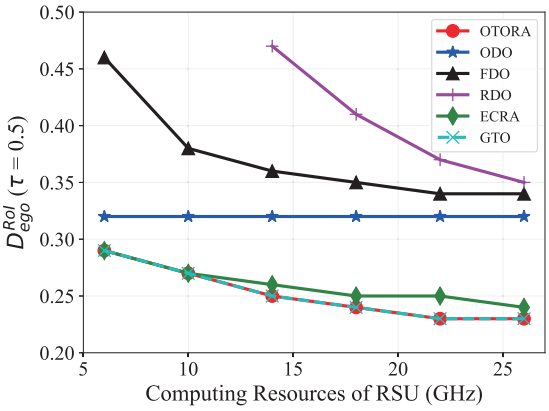
(a) 任务车辆数量增长对 RoI 任务处理时延的影响



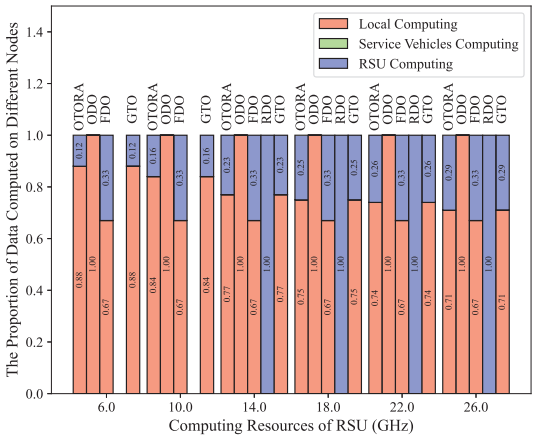
(b) 任务车辆数量增长对不同节点计算数据比例影响

图 4-3 任务车辆数量增长对任务处理时延和数据计算比例的影响

Figure 4-3 Effect of increasing number of task vehicles on (a) RoI task processing delay and (b) data computation proportions among various nodes ($\left|\Omega_{server}^{RoI}\right| = 5$)



(a) RSU 计算资源增长对 RoI 任务处理时延的影响

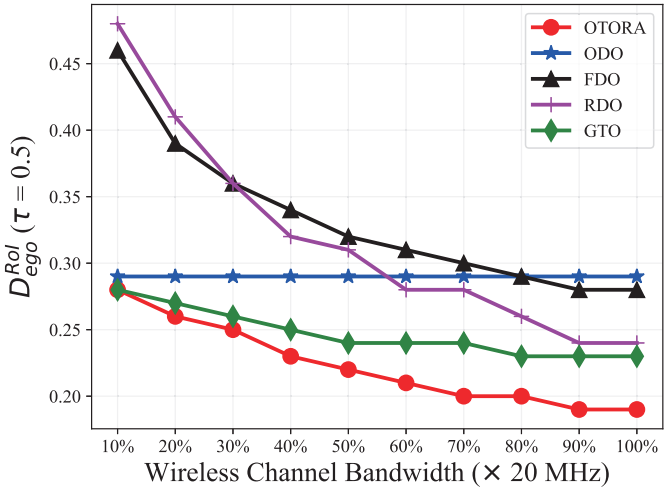


(b) RSU 计算资源增长对不同节点计算数据比例影响

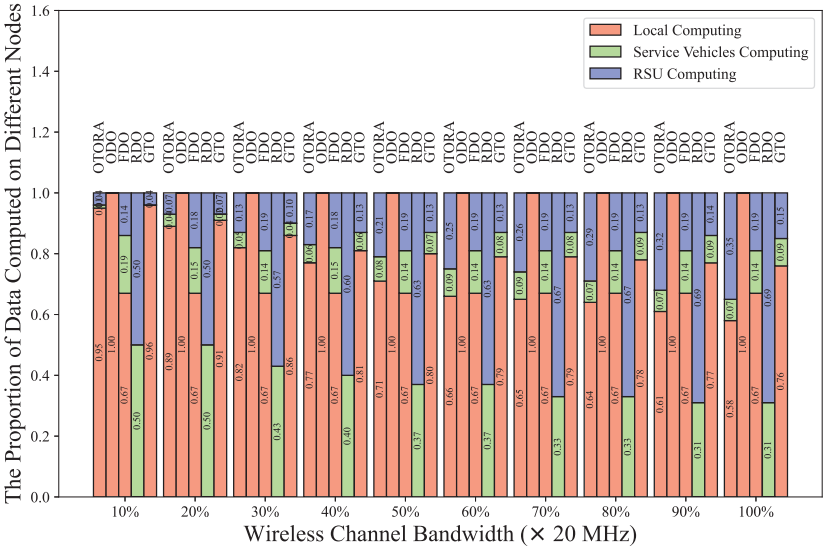
图 4-4 RSU 计算资源增长对任务处理时延和数据计算比例的影响

Figure 4-4 Effect of RSU computing resource increase on (a) RoI task processing delay and (b) data computation proportions among various nodes ($\left|\Omega_v^{RoI}\right| = 4, \left|\Omega_{server}^{RoI}\right| = 1, \tau = 0.5$ s)

渐降低，除了 ODO 方案，因为 ODO 方案完全在本地执行计算，不进行任务卸载，故任务完成时延与 RSU 的计算资源无关。图4-4(b)表示随着 RSU 的计算资源增加，任务数据在不同节点的计算比例，FDO 方案的本地计算和 RSU 计算的数据比例不变，但 RSU 的计算资源增加，所以任务完成时延减少。RDO 方案在 RSU 计算资源为 6 GHz 和 10 GHz 时，由于不满足任务的时延要求，无可行解，当 RSU 计算资源增加到 14 GHz 后，



(a) 无线信道带宽增长对 RoI 任务处理时延的影响



(b) 无线信道带宽增长对不同节点计算数据比例影响

图 4-5 RSU 计算资源增长对任务处理时延和数据计算比例的影响

Figure 4-5 Effect of wireless channel bandwidth increase on data computation proportions among various nodes ($\left|\Omega_v^{RoI}\right| = 4, \left|\Omega_{server}^{RoI}\right| = 5, \tau = 0.5$ s)

RSU 的算力资源能够满足其容忍时延要求。OTORA 方案的任务完成时延低于资源平均分配 ECRA 方案，其资源分配方式更加合理科学。GTO 方案的性能与 OTORA 方案一致，原因在于此时数据占比分配完全一致，资源分配方案相同。而 ECRA 方案由于是平均分配资源的，此时性能弱于 OTORA 方案和 GTO 方案。

图4-5(a)表示无线信道带宽对 RoI 任务处理时延的影响，参数设置为 $\left|\Omega_v^{RoI}\right| = 4, \left|\Omega_{server}^{RoI}\right| = 5, \tau = 0.5$ s。随着无线信道带宽的增加，任务车辆卸载到服务节点的传输速率增加，任务传输时延减少，总任务完成时延减少，除 ODO 方案外，因为其任务数

据全部在本地计算，无需进行卸载。RDO 和 FDO 方案均需要卸载到选择的服务节点，随着无线信道带宽增加，其卸载带来的传输时延减少收益大于 ODO 方案。图4-5(b)表示随着无线信道带宽增加，任务数据在不同节点的计算比例，ODO 方案数据全程在本地进行计算，OTORA 方案逐步减少本地计算的数据比例，倾向于卸载带来的传输延时降低，逐步将任务数据卸载到 RSU 和服务车辆。RDO 方案则倾向于减少向服务车辆的卸载数据比例，增加向 RSU 的卸载比例。FDO 方案数据比例保持较稳定，计算资源的合理分配进而减少任务完成时延。GTO 方案的性能仅次于 OTORA 方案，原因在于仅考虑算力资源而缺少对数据占比的合理分配。

4.5 本章小结

本章针对网联自动驾驶协同感知场景，构建了协同感知任务卸载模型，考虑任务数据阶段性、任务组合性，分析不同数据级别情形下协作任务计算完成时延，在此基础上形式化定义最小化自主车辆协同感知任务完成时延，设计双层最优任务卸载与资源分配算法，最外层确定离散变量求解，内层将优化问题进一步转换为非线性优化问题，对连续变量进行求解。最后，进行全面的性能评估，证明所提任务卸载与资源分配算法的优越性。

第五章　面向网联自动驾驶的协同位姿优化及应用

5.1 引言

自动驾驶协同感知技术近年来受到研究学者的广泛关注，多个交通参与者的终端共享感知信息，对多个终端视角的感知信息进行融合，进行补盲和扩大感知范围，这一技术能够突破单一视角下盲区和远距离感知的固有限制。然而车辆的高移动性导致的位姿信息不准确，这会降低协作质量，通常是由于多种误差信息源以及通信时延的影响。协同感知技术通过共享丰富的信息，建立了明确的约束关系，这些关系有助于优化算法降低多种误差源的影响，并改善车辆的自身定位及其相对位姿。研究面向网联自动驾驶协同位姿优化是对现有高精度定位技术和协同感知技术的冗余补充，当定位系统传感器失效或可靠性降低，在城市峡谷或隧道等限制性很高或信号干扰很强的环境下，存在一定的局限性，通过协同位姿优化，充分利用环境中的其他车辆信息，能够改善协同感知的质量，提高数据资源的合理有效利用，提升协同感知任务的准确性。

协同感知中多源信息融合能够提高对目标和自身状态感知的精度，但现有工作未能对车辆位姿形成有效约束。KIM H 等人通过三种位置信息的测量，采用交替方向乘子法优化自动驾驶的协作定位性能，后续为进一步降低计算负载，考虑节点相对位置，并构建新的算法框架以完全去中心化的方式工作，提高了协作定位精度[182]。SOATTI G 等人通过车载网络增强定位性能，利用非合作特征作为参考点完善位置估计，感知的特征信息通过 V2V 链路和共识算法融合，提高车辆定位精度[183]。YANG P 等人为配备 GPS、惯性测量单元和集成传感系统的自动驾驶车辆开发了多传感器多车辆定位和移动跟踪框架，通过局部过滤和全局过滤获取目标的全局估计，提高了估计的准确性[184]。MENG W 等人开发 V2V 通信辅助协作定位算法，利用位置相关信息和车辆接收的多路径无线电信号特征，推导基于特征的同步定位与建图（Simultaneous Localization and Mapping，SLAM）的贝叶斯模型，使用协作置信传播算法基于因子图联合进行定位，相比非 V2V 协作场景具有更好的定位性能[185]。FANG S 等人提出一种分散式框架的准确鲁棒的迭代分裂协方差交集滤波器的协同定位策略，利用相邻车辆相互共享的信息获得协作相对位姿估计[186]。上述研究基于不同的模型假设和角度利用协同共享信息提升定位性能，本节基于协同感知场景，充分利用定位和感知信息提升协同感知准确性。

研究网联自动驾驶协同位姿优化利用多个车辆之间的感知数据和定位信息，通过算法融合和优化，提高自动驾驶系统的定位精度和稳定性。这对于实现更安全、可靠的自动驾驶技术至关重要。在城市环境、高速公路等复杂场景中，自动驾驶系统面临着多路径效应、信号遮挡、定位误差等挑战。通过网联协同位姿优化，可以有效应对这些挑战，提高系统在复杂环境中的适应能力。传统的自动驾驶系统通常依赖于昂贵的传感器设备和高精度的定位系统，成本和资源消耗较高。通过网联协同位姿优化，可以降低对昂贵设备的依赖，减少系统成本和资源消耗。

本节首先对网联协同自动驾驶交通场景中的参与特征进行状态描述，考虑高精地图、抽象的自车和他车传感器信息，建立关于车-车、车-特征、车-障碍物以及高精地图，导航定位信息的因子图模型，基于不同的状态估计算法对自车及感知目标进行位姿优化。随后对不同级别协同感知融合策略进行仿真实验，开发了目标级协同感知原型系统，并进行了相应的时延测试和性能评估。

本章内容安排如下：5.1 节是本章的引言，介绍协同位姿优化研究现状和目前研究的不足以及本章的主要贡献。5.2 节描述了网联自动驾驶协同位姿优化框架及其仿真性能。5.3 节搭建了基于 C-V2X 设备和计算单元的硬件在环试验平台，设计了目标级协同感知系统，并对系统性能进行了可行性和有效性验证。5.4 节总结了本章的研究工作。

5.2 网联协同自动驾驶协同位姿优化框架

5.2.1 系统状态描述

本节考虑网联自动驾驶交通场景，其交通要素包括智能网联汽车、静态特征，比如灯杆、交通标志、信号灯等，动态目标，其包括非网联车、行人等。同时，网联自动驾驶汽车的地图信息包含了高精度的静态特征信息等。考虑所有交通要素位置随时间实时变化，联网车辆集合为 $\mathcal{V}=\{1,2,...,N_v\}$，$N_v$ 为场景中联网车辆的数量。时隙集合为 $\mathcal{T}=\{\tau,...,T\}$，则网联车 i 在时隙 t 的状态为 $\mathbf{x}_{i,t}^{(V)}$，简化为位置和航向角，如下式

$$\mathbf{x}_{i,t}^{(V)}=\left[\mathbf{p}_{i,t}^{(V)},\theta_{i,t}^{(V)}\right]=[x_{i,t}^{(V)},y_{i,t}^{(V)},\theta_{i,t}^{(V)}]^T, i\in\mathcal{V}, t\in\mathcal{T} \tag{5.1}$$

所有网联车辆的状态集合 $\boldsymbol{X}_t^{(V)}$ 定义为

$$\boldsymbol{X}_t^{(V)}=[\mathbf{x}_{1,t}^{(V)}\dots\mathbf{x}_{N_v,t}^{(V)}], i\in\mathcal{V}, t\in\mathcal{T} \tag{5.2}$$

静态定位特征集合为 $\mathcal{F}=\{1,2,...,N_f\}$，静态特征 j 在时隙 t 的状态 $\mathbf{x}_{j,t}^{(F)}$ 为

$$\mathbf{x}_{j,t}^{(F)}=[x_{j,t}^{(F)},y_{j,t}^{(F)}]^T,\ j\in\mathcal{F},t\in\mathcal{T} \tag{5.3}$$

则所有静态定位特征的状态集合 $\mathbf{X}_t^{(F)}$ 定义为

$$\mathbf{X}_t^{(F)}=[\mathbf{x}_{1,t}^{(F)}\dots\mathbf{x}_{N_f,t}^{(F)}],\ j\in\mathcal{F},t\in\mathcal{T} \tag{5.4}$$

动态目标集合为 $\mathcal{O}=\{1,2,...,N_o\}$，动态目标 k 在时隙 t 的状态 $\mathbf{x}_{i,t}^{(O)}$ 为

$$\mathbf{x}_{k,t}^{(O)}=\left[\mathbf{p}_{k,t}^{(O)},\theta_{k,t}^{(O)}\right]=[x_{k,t}^{(O)},y_{k,t}^{(O)},\theta_{k,t}^{(O)}]^T,k\in\mathcal{O},t\in\mathcal{T} \tag{5.5}$$

考虑某一传感器观测数据的生成过程，该过程本身携带了一定程度的随机误差。用系统的观测方程来描述这一过程。时隙 t 自车感知传感器的观测方程为

$$\mathbf{z}_{i,j,t}^{(S)}=h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)},\mathbf{x}_{i,t}^{(V)}\right)+\mathbf{v}_{i,j,t}^{(S)},\mathbf{v}_{i,j,t}^{(S)}\sim\mathcal{N}(0,\mathbf{R}_{i,j,t}^{(S)}) \tag{5.6}$$

$\mathbf{v}_{i,j,t}^{(S)}$ 是测量噪声，反映了观测过程中的不确定性。$\boldsymbol{R}_{i,j,t}^{(S)}$ 是噪声的协方差矩阵，表示为

$$\boldsymbol{R}_{i,j,t}^{(S)}=\begin{bmatrix}\delta_{sensor,x}^2 & 0 & 0\\ 0 & \delta_{sensor,y}^2 & 0\\ 0 & 0 & \delta_{sensor,\theta}^2\end{bmatrix} \tag{5.7}$$

$h^{(S)}\left(\mathbf{x}_{i,t}^{(V)},\mathbf{x}_{j,t}^{(V)}\right)$ 是关于状态量的函数，描述了不同状态量的空间坐标系转换，表示为

$$h^{(S)}\left(\mathbf{x}_{i,t}^{(V)},\mathbf{x}_{j,t}^{(V)}\right)=\begin{bmatrix}\cos\left(\theta_{i,t}^{(V)}\right) & -\sin\left(\theta_{i,t}^{(V)}\right) & 0\\ \sin\left(\theta_{i,t}^{(V)}\right) & \cos\left(\theta_{i,t}^{(V)}\right) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_{j,t}^{(V)}-x_{i,t}^{(V)}\\ y_{j,t}^{(V)}-y_{i,t}^{(V)}\\ \theta_{j,t}^{(V)}-\theta_{i,t}^{(V)}\end{bmatrix} \tag{5.8}$$

状态量和观测量之间的似然概率及各变量概率之间的依赖关系可以借助因子图进行分析，如图5-1所示，因子图包括两类节点，系统状态的变量节点和表示观测值及变量之间条件概率的函数节点。地图的观测只与定位特征的状态有关，车载组合定位只与车辆状态有关，为单边函数节点。车辆基于车载传感器对外部的测量只与车辆自身状态、定位特征状态和动态目标状态有关，车-车，车-动态目标，车-定位特征的观测对应双边函数节点。

时隙 t 内自车传感器观测值的概率分布如下

$$P(\mathbf{z}_{i,j,t}^{(S)}|\mathbf{x}_{j,t}^{(\Xi)},\mathbf{x}_{i,t}^{(V)})\sim\mathcal{N}(h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)},\mathbf{x}_{i,t}^{(V)}\right),\mathbf{R}_{i,j,t}^{(S)}) \tag{5.9}$$

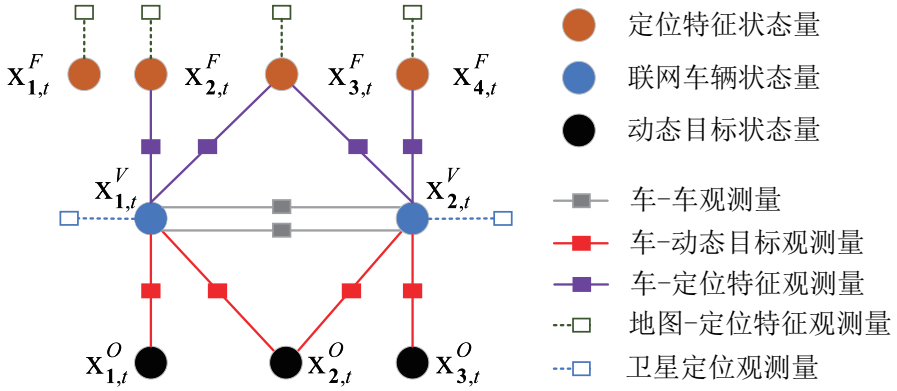


图 5-1 感知系统因子图

Figure 5-1 Factor graph of perception system

其中 $\Xi \in \{V, F, O\}$。

时隙 t 内，自车全球导航卫星系统（Global Navigation Satellite System，GNSS）传感器观测方程为

$$\mathbf{z}_{i,t}^{(\mathrm{G})} = h^{(G)}\left(\mathbf{x}_{i,t}^{(\mathrm{V})}\right) + \mathbf{v}_{i,t}^{(G)}, \mathbf{v}_{i,t}^{(G)} \sim \mathcal{N}(0, \mathbf{R}_{i,t}^{(G)}) \tag{5.10}$$

$\mathbf{v}_{i,t}^{(G)}$ 是 GNSS 传感器的测量噪声，$\mathbf{R}_{i,t}^{(G)}$ 是噪声的协方差矩阵，表示为

$$\boldsymbol{R}_{i,t}^{(G)} = \begin{bmatrix} \delta_{GNSS,x}^2 & 0 & 0 \\ 0 & \delta_{GNSS,y}^2 & 0 \\ 0 & 0 & \delta_{GNSS,\theta}^2 \end{bmatrix} \tag{5.11}$$

给定自车状态参数下，自车 GNSS 观测量的概率分布为

$$P(\mathbf{z}_{i,t}^{(\mathrm{G})} | \mathbf{x}_{i,t}^{(\mathrm{V})}) \sim \mathcal{N}(h^{(G)}\left(\mathbf{x}_{i,t}^{(\mathrm{V})}\right), \mathbf{R}_{i,t}^{(G)}) \tag{5.12}$$

时隙 t 内，高精地图对定位特征的观测方程为

$$\mathbf{z}_{j,t}^{(\mathrm{M})} = h^{(M)}\left(\mathbf{p}_{j,t}^{(\mathrm{F})}\right) + \mathbf{v}_{j,t}^{(M)}, \mathbf{v}_{j,t}^{(M)} \sim \mathcal{N}(0, \mathbf{R}_{j,t}^{(M)}) \tag{5.13}$$

$\mathbf{v}_{j,t}^{(M)}$ 是高精地图传感器测量噪声，$\mathbf{R}_{j,t}^{(M)}$ 是噪声的协方差矩阵，表示为

$$\boldsymbol{R}_{i,t}^{(M)} = \begin{bmatrix} \delta_{map,x}^2 & 0 \\ 0 & \delta_{map,y}^2 \end{bmatrix} \tag{5.14}$$

时隙 t 内，高精地图观测量的概率分布如下

$$P(\mathbf{z}_{j,t}^{(\mathrm{M})} | \mathbf{p}_{j,t}^{(\mathrm{F})}) \sim \mathcal{N}(h^{(M)}\left(\mathbf{p}_{j,t}^{(\mathrm{F})}\right), \mathbf{R}_{j,t}^{(M)}) \tag{5.15}$$

5.2.2 状态估计器

在车联网协同环境下，时隙 t 内，车辆能够通过所有传感器获取到的观测量集合 $\boldsymbol{Z}_t$，考虑系统运动方程和观测方程，对所有状态量集合 $\boldsymbol{X}_t$ 进行估计，假设观测量中各传感器各时隙的测量概率相互独立，整个系统的似然概率密度函数为所有观测量条件概率乘积，表示如下

$$\begin{aligned} P(\boldsymbol{Z}_t|\boldsymbol{X}_t) &= P\left(\boldsymbol{X}_t^{(V)}, \boldsymbol{X}_t^{(F)}, \boldsymbol{X}_t^{(O)} \middle| \boldsymbol{X}_t\right) \\ &= \prod_{i,j} P(\mathbf{z}_{i,j,t}^{(\mathrm{S})}|\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)}) \prod_{i} P(\mathbf{z}_{i,t}^{(\mathrm{G})}|\mathbf{x}_{i,t}^{(V)}) \prod_{j} P(\mathbf{z}_{j,t}^{(\mathbf{M})}|\mathbf{p}_{j,t}^{(\mathrm{F})}) \end{aligned} \tag{5.16}$$

本节的状态估计器即对以状态量集合为条件的观测量的概率进行最大化估计，即

$$\boldsymbol{X}_t^* = \arg\max P(\boldsymbol{Z}_t|\boldsymbol{X}_t) \tag{5.17}$$

式(5.16)中车辆的感知传感器在 t 时隙的观测值概率分布

$$\begin{aligned} &P(\mathbf{z}_{i,j,t}^{(S)}|\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)})) = \mathcal{N}\left(h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)}\right), \mathbf{R}_{i,j,t}^{(S)}\right) \\ &= \frac{1}{\sqrt{(2\pi)^N \det(\mathbf{R}_{i,j,t}^{(S)})}} \exp\left(-\frac{1}{2}\left(\mathbf{z}_{i,j,t}^{\Xi} - h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)}\right)\right)^{\mathrm{T}} \mathbf{R}_{i,j,t}^{\Xi\,-1}\left(\mathbf{z}_{i,j,t}^{\Xi} - h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)}\right)\right)\right) \end{aligned} \tag{5.18}$$

两侧取负对数得

$$\begin{aligned} P(\mathbf{z}_{i,j,t}^{(S)}|\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)})) = &-\frac{1}{2}\ln\left((2\pi)^N \det(\mathbf{R}_{i,j,t}^{\Xi})\right) + \frac{1}{2}\left(\mathbf{z}_{i,j,t}^{\Xi} - h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)}\right)\right)^{\mathrm{T}} \\ &\mathbf{R}_{i,j,t}^{\Xi\,-1}\left(\mathbf{z}_{i,j,t}^{\Xi} - h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)}\right)\right) \end{aligned} \tag{5.19}$$

其中右侧第一项为常数，求解式(5.17)的优化问题可以总结为

$$\begin{aligned} \boldsymbol{X}_t^* = \arg\min &\sum_{i,j}\left(\mathbf{z}_{i,j,t}^{(S)} - h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)}\right)\right)^{\mathrm{T}} \mathbf{R}_{i,j,t}^{\Xi\,-1}\left(\mathbf{z}_{i,j,t}^{(S)} - h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)}\right)\right) \\ &+ \sum_{i}\left(\mathbf{z}_{i,t}^{G} - h^{(G)}\left(\mathbf{x}_{i,t}^{(V)}\right)\right)^{\mathrm{T}} \mathbf{R}_{i,t}^{G\,-1}\left(\mathbf{z}_{i,t}^{G} - h^{(S)}\left(\mathbf{x}_{i,t}^{(V)}\right)\right) \\ &+ \sum_{j}\left(\mathbf{z}_{j,t}^{M} - h^{(M)}\left(\mathbf{x}_{j,t}^{(V)}\right)\right)^{\mathrm{T}} \mathbf{R}_{j,t}^{M\,-1}\left(\mathbf{z}_{j,t}^{M} - h^{(M)}\left(\mathbf{x}_{j,t}^{(V)}\right)\right) \end{aligned} \tag{5.20}$$

定义传感器观测残差

$$\mathbf{e}_{i,j,t}^{(S)} = \left(\mathbf{z}_{i,j,t}^{(S)} - h^{(S)}\left(\mathbf{x}_{j,t}^{(\Xi)}, \mathbf{x}_{i,t}^{(V)}\right)\right) \tag{5.21}$$

定义 GNSS 传感器观测残差

$$\mathbf{e}_{i,t}^{(G)} = \left(\mathbf{z}_{i,t}^{(G)} - h^{(G)}\left(\mathbf{x}_{i,t}^{(V)}\right)\right) \tag{5.22}$$

定义地图传感器观测残差

$$\mathbf{e}_{j,t}^{(M)} = \left(\mathbf{z}_{j,t}^{(M)} - h^{(M)}\left(\mathbf{x}_{j,t}^{(V)}\right)\right) \tag{5.23}$$

观测残差反映了不同传感器的实际观测值和预测值之差。基于定义的观测残差，式(5.20)等价为

$$\begin{aligned} \boldsymbol{X}_t^* = \arg\min & \sum_{i,j}\left(\mathbf{e}_{i,j,t}^{(S)}\right)^{\mathrm{T}} {\mathbf{R}_{i,j,t}^{\Xi}}^{-1}\left(\mathbf{e}_{i,j,t}^{(S)}\right)\sum_{i}\left(\mathbf{e}_{i,t}^{(G)}\right)^{\mathrm{T}}\mathbf{R}_{i,t}^{G\,-1}\left(\mathbf{e}_{i,t}^{(G)}\right) \\ & + \sum_{j}\left(\mathbf{e}_{j,t}^{(M)}\right)^{\mathrm{T}}\mathbf{R}_{j,t}^{M\,-1}\left(\mathbf{e}_{j,t}^{(M)}\right) \end{aligned} \tag{5.24}$$

算法 5-1: 列文伯格-马夸尔特法

输入: 调整因子 λ，阈值 Δ，最大迭代次数 i_{max}，初始值 $\boldsymbol{X}_t^0$

输出: $X_t{}^*$

```
while i < i_max do
    计算优化函数梯度 ∇f(X_t^i)
    计算优化函数海森矩阵 H(X_t^i)
    计算更新参数 X_t^{i+1} = X_t^i − (H(X_t^i) + λI)^{-1} ∇f(X_t^i)
    计算更新后的目标函数 f(X_t^{i+1})
    if |f(X_t^{i+1}) − f(X_t^i)| < Δ then
        X_t* = X_t^{i+1}
        break
    else
        λ = λ / 10
        i = i + 1
    end
end
```

该问题为非线性最小二乘问题，$\boldsymbol{X}_t^*$ 是需要优化的参数，$\mathbf{e}_{i,j,t}^{(S)}$，$\mathbf{e}_{i,t}^{(G)}$ 和 $\mathbf{e}_{j,t}^{(M)}$ 是误差向量，${\mathbf{R}_{i,j,t}^{\Xi}}^{-1}$，$\mathbf{R}_{i,t}^{G\,-1}$ 和 $\mathbf{R}_{j,t}^{M\,-1}$ 是对应的权重矩阵。采用广泛使用的列文伯格-马夸尔特法（Levenberg-Marquardt method）求解该问题，见算法 5-1。该方法结合了梯度下降法

和高斯-牛顿法的特点，基本思路是通过迭代的方式逐步优化目标函数，首先利用梯度下降法找到一个局部最小值附近的解，然后利用高斯-牛顿法来进一步精细调整以接近全局最小值[187]。

5.2.3 实验仿真及结果分析

本节对多车协同感知状态估计进行仿真和分析，首先对仿真场景及参数和仿真方案进行设计，然后从不同设置参数对车辆状态估计性能进行评估。

仿真场景针对智能网联汽车面临的特征密集场景和过渡场景，如图5-2所示，在Vissim 仿真软件中构建了两条双向七车道的道路，每条道路长 380 m，每条车道宽 3.5 m，组成一个十字路口，十字路口放置若干灯杆、交通灯、交通指示牌等；行人在非机动车道行走及穿行过马路。外围区域为过渡场景，不放置任何障碍物。网联车辆和非网联车辆从过渡场景的四个路口进入，在十字路口相遇，随后直行或转向行驶，特征密集场景道路长 180 m。由于不同车辆生成时间不同，其达到十字路口时间不同，大体是在仿真中间时刻到达特征密集场景区域，仿真的起始和结束阶段为过渡场景。

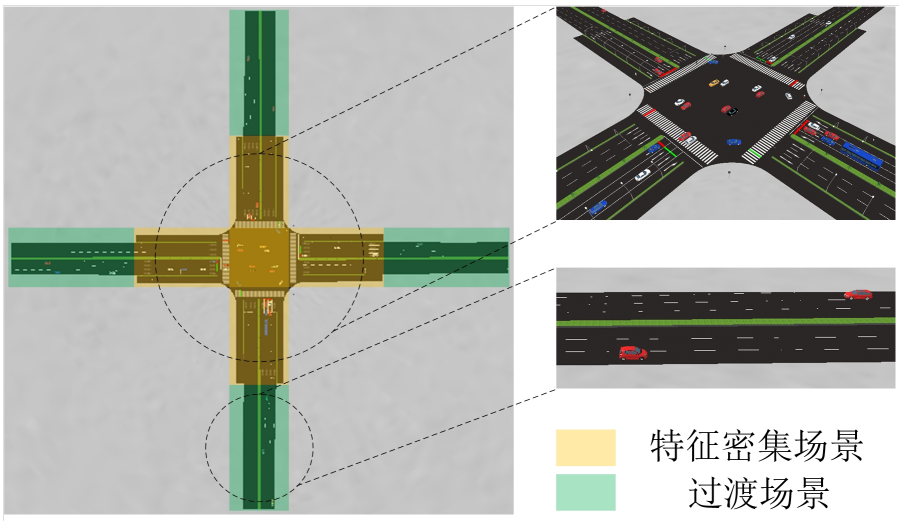


图 5-2 交通仿真场景

Figure 5-2 Traffic simulation scenario

智能网联车辆的车载传感器种类繁多，为不失一般性，对单车感知系统进行抽象，如图5-3所示，单车的感知需求范围为尺寸为 W_d 和 $l_r + l_f$ 的矩形，其中 l_f 为需要感知的前向距离，l_r 为需要感知的后向距离，W_d 为需要感知的宽度。车辆的传感器感知能力假设为半径为 R_s 的前向扇形，视场角为 θ_{FOV}。同时单车会存在盲区或遮挡问题，

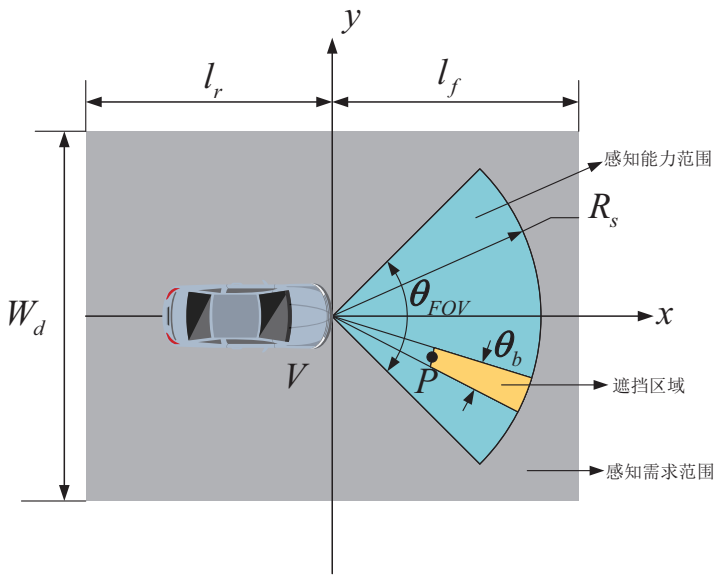


图 5-3 单车感知范围示意图

Figure 5-3 Illustration of single vehicle perception range

即图中由于 P 点的存在导致形成角度为 θ_b 的黄色区域。场景中参数设置为 $\theta_{FOV}=70$ m，$R_s=80$ m，$\theta_b=2°$，$L_f=100$ m，$L_r=30$ m，$W_d=60$ m。

在上述典型场景下对状态估计器的性能进行评估，重复进行 100 次实验，独立对观测量添加噪声。对所有网联车的姿态误差进行统计分析，定义自车位姿全局均方根误差（Root Mean Square Error，RMSE）：

$$e^V_{RMSE,t}=\sqrt{\frac{1}{MN_v}\sum_{j=1}^{M}\sum_{i=1}^{N_v}\left(\left\|\hat{\mathbf{p}}^V_{i,j,t}-\mathbf{p}^V_{i,j,t}\right\|_2^2+(\hat{\theta}^V_{i,j,t}-\theta^V_{i,j,t})^2\right)} \tag{5.25}$$

其中 M 为实验次数，$\hat{\mathbf{p}}^V_{i,j,t}$ 为时隙 t 车辆 i 在第 j 次仿真中的定位结果，$\mathbf{p}^V_{i,j,t}$ 为位置真值，$\hat{\theta}^V_{i,j,t}$ 为时隙 t 车辆 i 在第 j 次仿真中的方向角结果，$\theta^V_{i,j,t}$ 为方向角的真值。

本节提出的基于因子图的状态估计方法（Factor Graph-based State Estimation Method，FGSEM）与以下基准方案进行对比：

1）隐式协同定位（Implicit Cooperative Positioning，ICP）方案：利用非合作的物理特征（行人、交通灯等），通过消息传递进行共识融合来提高车辆定位精度[188]。

2）无高精地图（No High-Definition Map，NHDM）方案：高精地图由于特定区域、成本和更新速度原因无法全域覆盖，此方案模拟无高精地图的情况[183]。

3）V2V 通信断联（V2V Communication Disconnection，V2V-CD）方案：模拟 V2V 通信存在一定概率断联的情形[189]。

图5-4表示不同方案网联车辆的平均自定位性能对比，仿真参数设置 $N_v = 4$，$N_f = 6$，$N_o = 4$，$\delta_{sensor,p} = 0.1$ m，$\delta_{sensor,\theta} = 0.1$ rad，$\delta_{map,p} = 0.05$ m，$\delta_{GNSS,p} = 2.5$ m，$\delta_{GNSS,\theta} = 0.1$ rad。仿真时长为 250 s，0~80 s 和 180~250 s 处于特征稀疏区域，80~180 s 进入十字路口，处于特征密集区域。基于因子图的状态估计方法，融合自车与他车的传感器信息及高精地图数据，全局定位精度提升明显。仿真开始时，车辆位于过渡区域，无法形成车-车，车-特征或车-障碍物等约束，随着车辆逐渐进入特征密集区域，如图5-5所示，约束的数量逐渐增加，有效利用这些观测约束以及高精地图对定位特征的先验约束，车辆自定位误差逐渐降低，在特征密集区域定位精度在 0.3 m 以内。仅利用 GNSS 定位的误差较大，ICP 方案和 NHDM 方案在特征较密集区域定位性能有所下降，ICP 方案没有有效利用车车之间的观测约束，NHDM 方案缺少对车和特征之间的先验约束。FGSEM 方案充分利用了定义中的所有约束，其约束总数平均最高达 50 次左右，当 V2V 通信发生一定概率断联时，导致 V2V 约束减少，影响了部分时刻的定位性能。

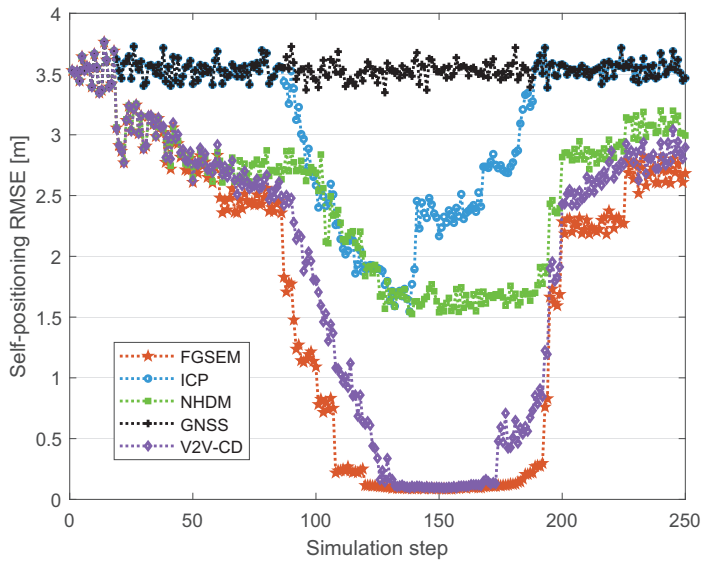


图 5-4 不同方案自定位性能对比

Figure 5-4 Comparison of self-localization performance across different schemes

图5-6表示不同网联车辆数量变化对 FGSEM 方案的性能影响，仿真参数设置 $N_f = 6$，$N_o = 2$，$\delta_{sensor,p} = 0.25$ m，$\delta_{sensor,\theta} = 0.25$ rad，$\delta_{map,p} = 0.05$ m，$\delta_{GNSS,p} = 2.5$ m，$\delta_{GNSS,\theta} = 0.1$ rad。可以看到，即使只有少量联网车，在特征密集区域，FGSEM 方案的定位性能仍有大幅提升。联网车辆数量越多，所有车辆的平均自定位精度越高，在过

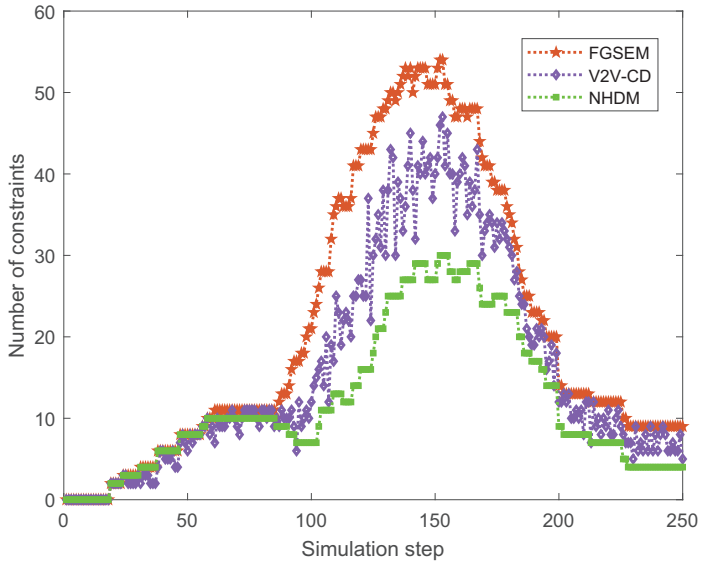


图 5-5 不同方案的约束数量

Figure 5-5 Number of constraints in different schemes

渡场景，定位性能改善源于车车之间的约束，过渡区域中联网车辆数量更加重要，而在特征密集区域，存在的特征数量本身以及高精地图的加持形成了较强的约束。

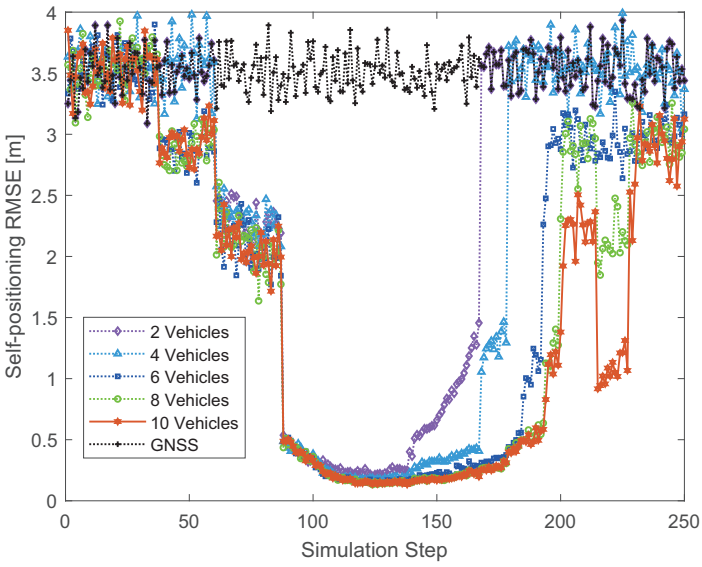


图 5-6 网联车辆数量变化对 FGSEM 方案性能的影响

Figure 5-6 The impact of changes in the number of connected vehicles on the performance of FGSEM scheme

图5-7表示不同特征数量变化对 FGSEM 方案的性能影响，仿真参数设置 $N_v = 4$，$N_o = 2$，$\delta_{sensor,p} = 0.25$ m，$\delta_{sensor,\theta} = 0.25$ rad，$\delta_{map,p} = 0.05$ m，$\delta_{GNSS,p} = 2.5$ m，$\delta_{GNSS,\theta} = 0.1$ rad。在特征稀疏区域，主要是 V2V 约束提高了平均定位性能。在特征密

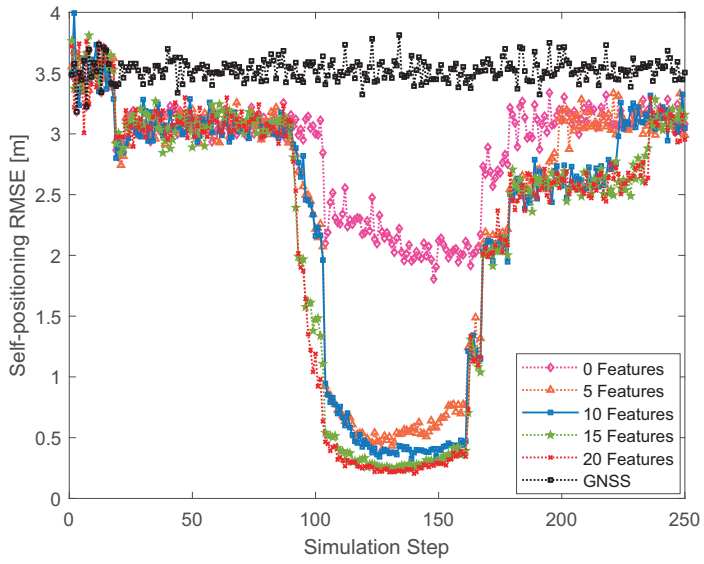


图 5-7 特征数量变化对 FGSEM 方案性能的影响

Figure 5-7 The impact of changes in the number of features on the performance of FGSEM scheme

集区域，无任何特征即紧靠 V2V 约束其定位精度达到 2 m 左右，随着特征数量的增加，引入了更多的定位特征约束以及地图的先验信息约束，使得定位误差显著降低。因此高精地图应该包含更多的特征信息有助于协同定位性能提升。

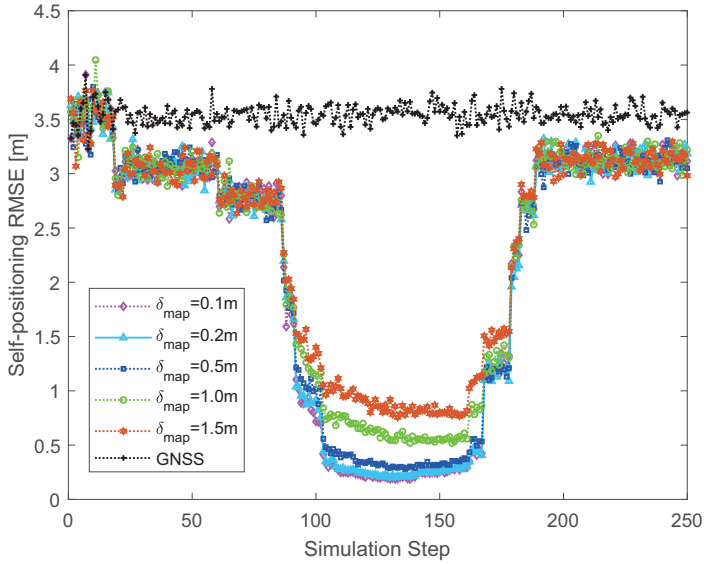


图 5-8 地图精度变化对 FGSEM 方案性能的影响

Figure 5-8 The impact of map accuracy variation on the performance of FGSEM scheme

图5-8表示地图精度变化对 FGSEM 方案的性能影响，仿真参数设置 $N_v = 4, N_f = 6$, $N_o = 4$，$\delta_{sensor,p} = 0.25$ m，$\delta_{sensor,\theta} = 0.25$ rad，$\delta_{GNSS,p} = 2.5$ m，$\delta_{GNSS,\theta} = 0.1$ rad。地图数据各维度标准差设置在 0.1 m 到 1.5 m 之间，在过渡区域，主要是 V2V 约束影响了

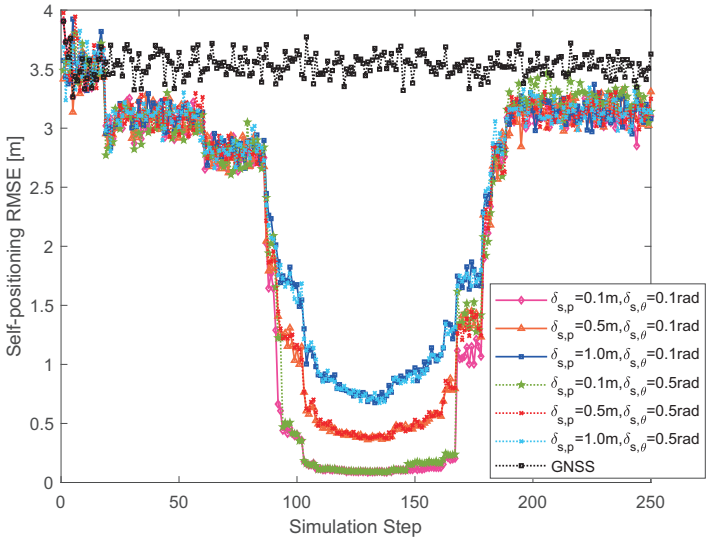


图 5-9 传感器感知精度变化对 FGSEM 方案性能的影响

Figure 5-9 The impact of sensor perception accuracy variation on the performance of FGSEM scheme

定位性能，进入特征密集区域，可以看到地图精度误差越小，能够增强协同定位效果。

图5-9表示车载传感器感知精度对 FGSEM 方案的性能影响，仿真参数设置 $N_v = 4$，$N_f = 6$，$N_o = 4$，$\delta_{map,p} = 0.05$ m，$\delta_{GNSS,p} = 2.5$ m，$\delta_{GNSS,\theta} = 0.1$ rad。在特征稀疏区域，随传感器感知精度增加，车辆平均定位精度性能提升无明显变化，随着车辆进入特征密集区域，对网联车、定位特征、动态目标等感知需求增加，形成的约束耦合性越强，此时传感器感知精度会影响到协同定位性能，感知越准确，定位精度越高。

5.3 目标级协同感知应用

协同感知技术利用车辆之间的通信和协作，实现感知数据的共享和集成处理，为自动驾驶系统提供更全面、准确和及时的环境感知。本节基于边缘计算平台和车载单元实现了 V2V 目标级协同感知算法，并对模块时延进行性能评估。5.2 节和此节内容的共同目标是提升自动驾驶系统的感知和决策能力，两者技术上互补，协同位姿优化框架提供准确位姿，协同感知系统增强感知能力，提升环境感知的整体性能。

5.3.1 仿真实验

车联网协同感知技术目前已开源仿真数据集 OPV2V[103]，V2XSet[104]，V2X-SIM[105]，DOLPHINS[106] 和实车采集数据集 V2V4Real[107]，DAIR-V2X[108]。仿真框架目前已开源 OpenCOOD[103]，OpenCDA[111, 112]，HEAL[113]。协同感知仿真的一般框架涉及配置模块、数据加载模块、前处理模块、后处理模块和融合模型，如图5-10，配

置系统对数据接口、训练模式和模型的训练参数进行配置。数据加载模块涉及不同融合策略，早期融合聚合所有原始点云信息，中期融合聚合中间的深度特征。前处理模块将原始点云转换为体素和鸟瞰特征，后处理模块将目标的边界框转化为正确的格式，模型的输出转换为正确的格式进行评估和可视化。融合模型即不同融合策略的网络架构。

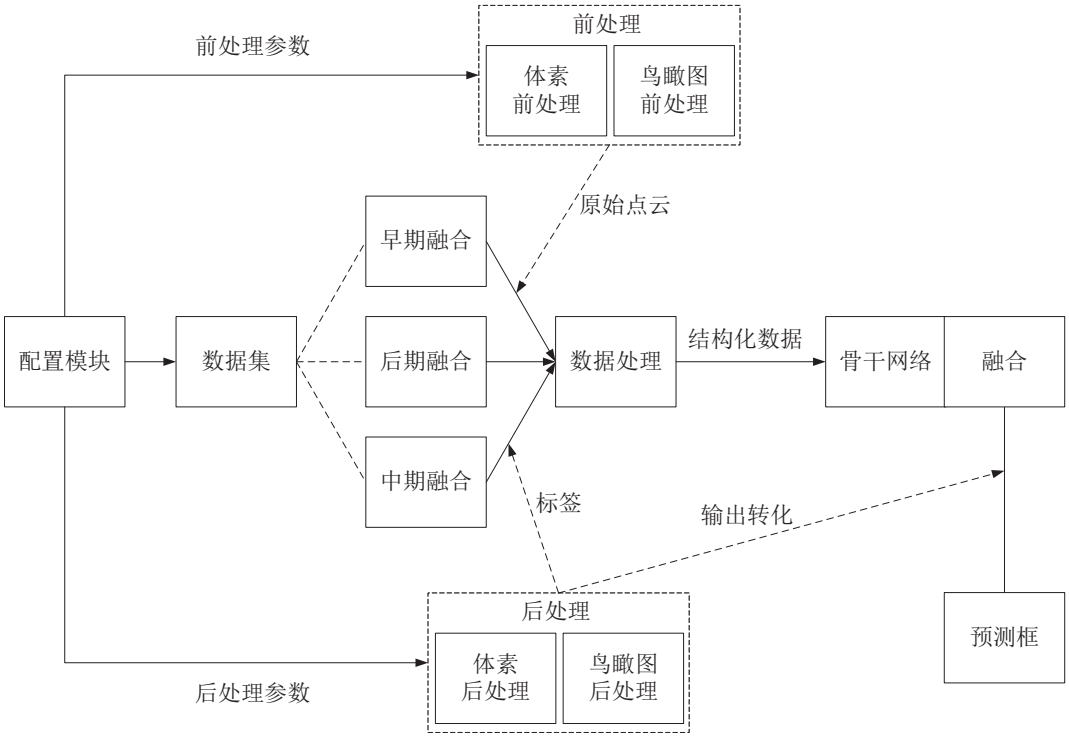


图 5-10 协同感知仿真框架

Figure 5-10 Simulation framework for collaborative perception

本节基于 OpenCOOD 框架，选择基于激光雷达的 3D 目标检测器 SECOND[152]、VoxelNet[151]、PIXOR[156] 和 PointPillars[157] 用于协同感知研究。VoxelNet 将点云数据表示为三维体素化形式，并利用卷积神经网络提取特征，其数据表示比较低效，3D 卷积计算量大。SECOND 用稀疏 3D 卷积代替了 VoxelNet 中的 3D 卷积层，避免空白区域的无效计算，提高了检测速度和内存使用。PIXOR 提出手工设计的方式，将 3D 体素压缩到 2D 的像素，避免 3D 卷积，但损失了高度方向上的信息。PointPillar 的思路是将 3D 点云量化到 2D 平面网格，网格内的点堆叠在一起，称为柱子，随后进行特征提取得到伪图片化数据，避免了 VoxelNet 中的 3D 卷积和空白区域的无效计算，也避免了 PIOXR 中手工设计特征导致信息丢失和网络适应性不强的问题。Cooper[114] 是首

表 5-1 不同融合策略目标检测性能

Table 5-1 Performance of object detection with different fusion strategies

骨干网络	融合策略	AP@IoU	
		0.5	0.7
PointPillars	早期融合	**0.913**	**0.839**
PointPillars	后期融合	0.815	0.747
PointPillars	Attentive Fusion (中期融合)	0.871	0.811
PointPillars	Where2comm (中期融合)	0.843	0.705
PointPillars	中期融合	0.879	0.811
SECOND	早期融合	0.877	0.813
SECOND	后期融合	0.846	0.775
SECOND	中期融合	**0.903**	**0.856**
VoxelNet	Cooper (早期融合)	0.852	0.758
VoxelNet	后期融合	0.801	0.738
VoxelNet	F-Cooper (中期融合)	**0.906**	**0.864**
PIXOR	早期融合	0.810	0.678
PIXOR	后期融合	0.769	0.578
PIXOR	中期融合	**0.815**	**0.687**

个基于原始点云数据融合的协同感知策略，提出可适应低密度点云的方法，经过预处理、体素特征提取、稀疏卷积中间层，随后进行特征图拼接输入区域提案网络（Region Proposal Network，RPN）进行目标检测。F-Cooper[115] 设计了体素特征融合和空间特征融合方案，空间特征融合可动态调整特征图的大小，随后将融合的特征图输入区域提案网络进行目标检测。Attentive Fusion[103] 提出了一个注意力中间融合管道来获取通信车辆的特征之间的交互，帮助网络关注重要的传感器数据信息。Where2comm[118] 构建空间置信度图，决定与谁通信并聚合谁的信息，多次沟通并补充信息，实现高效协作。本节基于 OpenCOOD 框架在 OPV2V 数据集进行仿真测试不同融合策略的检测性能，见表5-1。两辆自动驾驶汽车在不同场景下分别采集并分阶段处理感知数据，进行独立的训练和测试。这批数据包含了 73 个多样化的场景、6 种道路类型及 9 座城市的特征，确保了数据的广泛代表性和典型性。常用于评价目标检测模型性能的指标为平

均精度（Average Precision，AP）和交并比（Intersection over Union，IoU），IoU 反映标注框和预测框的重合程度，即两框重叠部分的面积与两框总共部分面积之比。AP 即精度和召回率曲线与坐标轴围成的面积，AP 越高，精度和召回率越高。表展示了 IoU 阈值为 0.5 和 0.7 时的平均精度，早期融合由于保留了更多的信息，其融合效果优于后期融合。基于 SECOND、VoxelNet 和 PIXOR 骨干网络的中期融合模式取得了更好的性能，得益于引入的压缩、共享和注意力模块。中期融合策略是平衡性能和传输带宽的一种选择，在实测数据集中，早期融合每帧传输点云的平均数据大小约为 1382275.75 字节，后期融合每帧传输目标检测框的平均数据大小约为 336.16 字节[108]。基于现有自动驾驶车辆的感知系统，最先容易实现的是后期融合，其数据准备程度高，现阶段的通信设备能够满足其传输带宽，不需要对自动驾驶汽车的检测模型进行调整，只需要在现有检测模型的输出结果后面进行融合，能够便利地兼容现有系统，升级成本低。

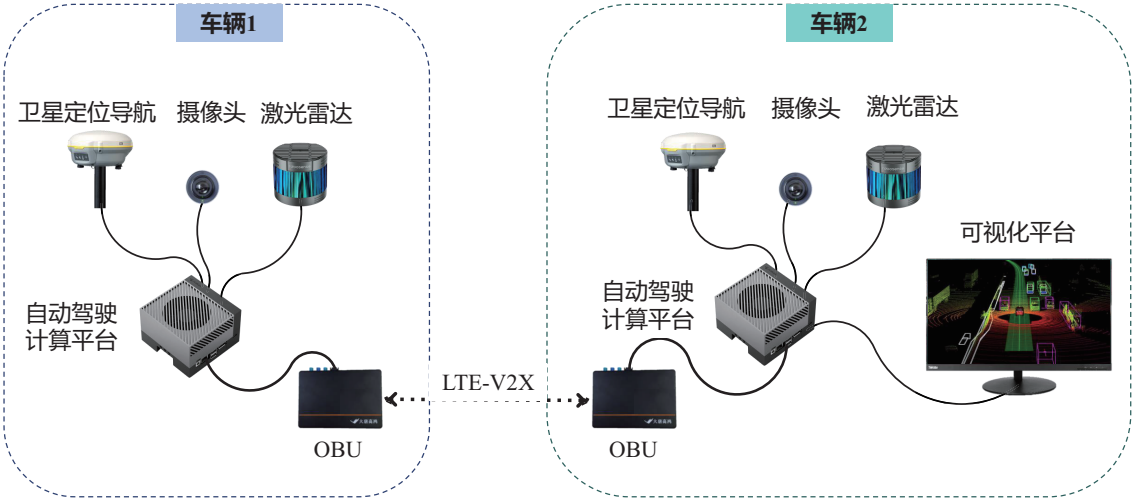


图 5-11 硬件在环平台框架

Figure 5-11 Hardware-in-loop platform framework

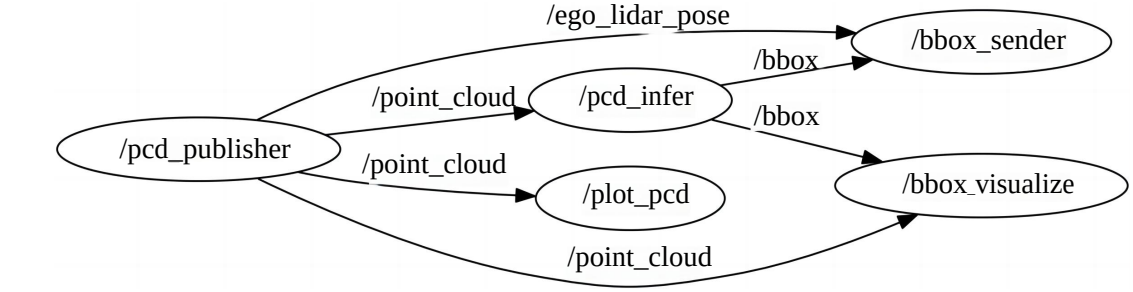
5.3.2 硬件在环实测平台

本节基于 C-V2X 通信设备搭建目标级协同感知在环试验平台，实现了后期融合算法，并对传输链路时延、丢包率等进行统计和分析，验证原型系统的可行性。基于 C-V2X 设备的硬件在环试验平台框架如图5-11所示。本系统考虑两辆车，并配备大唐的支持 LTE-V2X OBU 设备，支持 3GPP Release 14 PC5 直连通信，工作频段 5905-5925 MHz，工作带宽 10 MHz/20 MHz，发送功率最大 23±2 dBm，具有 GNSS 天线，接收卫星信号。在室内场景难以接收卫星信号，因此选择靠窗位置进行实验。每辆车的自动

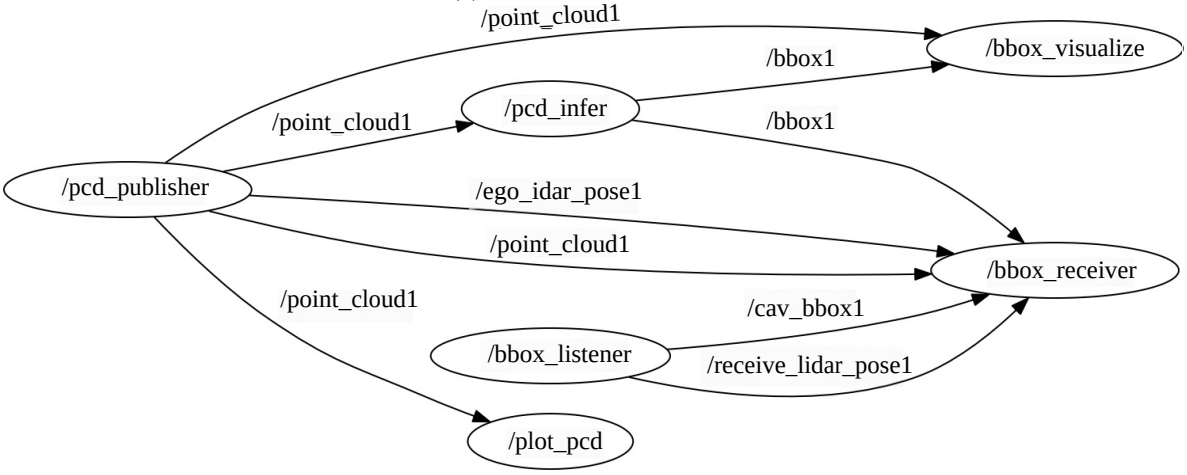
驾驶计算平台配备 NVIDIA 的 Jetson AGX Orin 32 GB，算力高达 200 TOPS，CPU 最大频率 2.2 GHz，GPU 最大频率 930 MHz。计算平台负责处理传感器收集的数据，部署检测模型，并与其下位机 OBU 进行通信。由于已有公开的协同感知数据集，摄像头、激光雷达等传感器采集数据通过直接读取现有协同感知数据集的数据替代，适用于室内测试。每辆车都配备有显示屏用于可视化感知数据，车辆 1 作为发送端，车辆 2 作为接收端，车辆 1 通过传感器采集数据，经过计算平台检测模型输出，并在显示屏可视化激光雷达的原始点云数据和检测框的数据，同时将检测框、雷达位姿的数据打包为用户数据报协议（User Datagram Protocol，UDP）报文转发到下位机 OBU 相应的端口，OBU 通过数据透传模式设定相应的传输配置参数，随后经过 LTE-V PC5 协议栈模块处理将数据发送出去。接收端的 OBU 接收到车辆 1 的检测框数据将其转发到制定的地址和端口，计算平台侦听并读取车辆 1 的检测框数据，车辆 2 的显示屏将自车的点云数据、自车检测框数据以及与车辆 1 检测框融合后的数据进行可视化。

目标级协同感知平台开发基于机器人操作系统（Robot Operating System，ROS），是一个分布式通信框架，帮助程序进程之间更方便地通信。ROS 安装在 Linux 操作系统运行，其作用是连接真正的操作系统和使用者开发的 ROS 应用程序（比如自动驾驶的感知、规划、决策等模块），建立沟通的桥梁。ROS 中可执行程序的基本单位叫节点，节点是可执行程序，节点之间通过消息机制进行通信，消息收发机制分为话题、服务和动作。ROS 利用节点将代码和功能解耦，提高了系统的容错性和可维护性。目标级协同感知平台的发送端和接收端的 ROS 节点图如图5-12所示，发送端共有 5 个 ROS2 节点（使用的 ROS2 版本），分别代表 5 个功能模块，共有三种消息类型，其中/point_cloud 即代表将点云数据封装为 ROS2 的消息类型，/ego_lidar_pose 代表自车激光雷达的位姿数据封装为 ROS2 的消息类型，/bbox 代表 3D 检测框封装为 ROS2 的消息类型。接收端共有 6 个 ROS2 节点，分别代表 6 个功能模块，/point_cloud1 代表接收端读取的点云数据 ROS2 消息类型，/ego_lidar_pose1 代表接收端车辆激光雷达的位姿数据 ROS2 消息类型，/bbox1 代表接收端 3D 检测框 ROS2 消息类型，/cav_bbox1 代表接收端侦听到发送端发送的检测框数据 ROS2 消息类型，/receive_lidar_pose1 代表接收端侦听到发送端发送的激光雷达位姿数据 ROS2 消息类型。发送端和接收端 ROS2 节点的功能描述见表5-2。接收端和发送端均包括点云数据读取与发布模块、点云数据可视化模块、点云

数据推理模块、点云数据及检测框可视化模块，发送端需要对检测框及雷达位姿数据进行封装，接收端则需要对数据进行侦听，并与接收端自身的数据进行融合及可视化。



(a) ROS2 发送端节点图



(b) ROS2 接收端节点图

图 5-12 ROS2 节点图

Figure 5-12 ROS2 node graph

表 5-2 ROS2 节点功能说明

Table 5-2 Functional description of ROS2 nodes

ROS2 发送端节点	功能	ROS2 接收端节点	功能
/pcd_publisher	点云数据读取及发布	/pcd_publisher	点云数据读取及发布
/plot_pcd	点云数据可视化	/plot_pcd	点云可视化
/pcd_infer	点云数据推理	/pcd_infer	点云数据推理
/bbox_visualize	点云数据及检测框可视化	/bbox_visualize	点云数据及检测框可视化
/bbox_sender	检测框数据封装	/bbox_listener	数据侦听模块
-	-	/bbox_receiver	数据融合及可视化

发送端和接收端的点云数据推理模块使用 NVIDIA 的推理部署工具 TAO（Train,

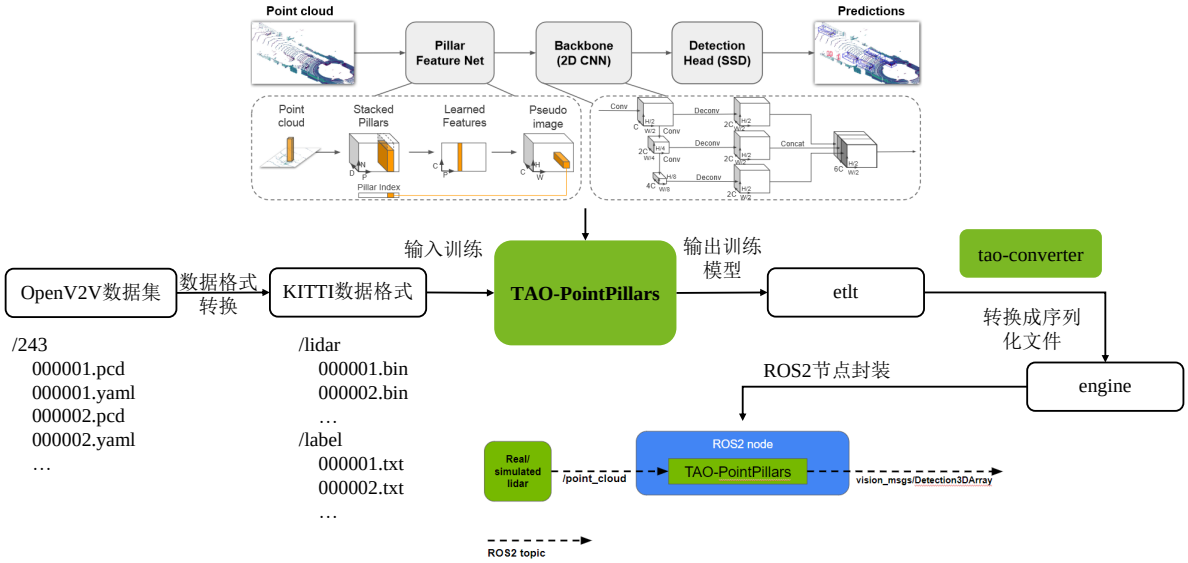


图 5-13 ROS2 推理节点部署流程

Figure 5-13 Deployment process for ROS2 inference nodes

Adapt，Optimize）进行模型训练并转换为可部署的序列文件。推理检测模型使用能够平衡检测速度和检测精度的 PointPillars 模型，在 TAO 工具包中训练 PointPillars 包括准备数据集、创建规范文件、训练模型、评估模型、运行推理、修建和重新训练、导出模型以及部署模型。激光雷达生成的扫描点组合在一起就是点云数据，点的特征可以表示为 (x, y, z, r)，分别表示 X 坐标、Y 坐标、Z 坐标和反射率（强度）。3D 检测框可以表示为 $(x, y, z, dx, dy, dz, yaw)$，分别表示检测框的中心点 X 坐标、Y 坐标、Z 坐标、长度（X 方向）、宽度（Y 方向）、高度（Z 方向）和朝向角。在准备数据集时，由于 OPV2V 数据集是左手坐标系，需要进行转换为右手坐标系，同时将点云数据格式转化为 KITTI[190] 的数据格式，随后创建规范实验文件进行模型训练，经过评估后将模型导出为 etlt 的部署格式，随后需要将可部署的模型封装为 ROS2 节点，使用 tao-converter 将 etlt 部署格式文件转化为 TensorRT engine 文件，将其部署到针对 TAO-PointPillars 开源的 ROS2 节点包即可实现点云数据推理功能。图5-13展示了 ROS2 推理节点的部署流程，PointPillar 对点云数据的处理首先在 X、Y 轴（不考虑 Z 轴）将数据划分为一个个的网格，落入网格的点云数据视为处在一个 Pillar 中，每个点云用 $D = 9$ 维的向量表示，分别为 $(x, y, z, r, x_c, y_c, z_c, x_p, y_p)$，其中 x, y, z, r 反映点云真实坐标信息和反射强度，x_c, y_c, z_c 为该点云所处 Pillar 中所有点的几何中心，$x_p = x - x_c$，$y_p = y - y_c$ 反映点与几何中心的相对位置，随后对样本维度进行统一，实现点云数据的张量化，进一步进

行特征提取，得到伪图片特征。骨干网络使用 2D CNN 进一步提取图片特征，获得三个不同分辨率的特征图，另一个网络进行采样至相同大小然后进行连接。网络输出的检测框用 7 维向量表示 $(x, y, z, \omega, h, l, \theta)$，$(x, y, z)$ 为中心点坐标，ω, l, h 为尺寸数据，θ 为方向角。检测框回归任务要学习的参数为 7 个变量的偏移量，预测检测框（Anchor Box）与真值（Ground Truth）之间的残差定义为

$$\Delta_x = \frac{x^{gt} - x^a}{d^a} \tag{5.26}$$

$$\Delta_y = \frac{y^{gt} - y^a}{d^a} \tag{5.27}$$

$$\Delta_z = \frac{z^{gt} - z^a}{h^a} \tag{5.28}$$

$$\Delta_\omega = log\frac{\omega^{gt}}{\omega^a} \tag{5.29}$$

$$\Delta_l = log\frac{l^{gt}}{l^a} \tag{5.30}$$

$$\Delta_h = log\frac{h^{gt}}{h^a} \tag{5.31}$$

$$\Delta_\theta = sin(\theta^{gt} - \theta^a) \tag{5.32}$$

其中 $d^a = \sqrt{(\omega^a)^2 + (l^a)^2}$，总的定位损失通过 Smooth L1 损失函数计算[143]：

$$\mathcal{L}_{loc} = \sum_{b\in(x,y,z,\omega,l,h,\theta)} L_{1,smooth}(\Delta b) \tag{5.33}$$

为避免方向判别错误，引入 Softmax 损失函数学习目标的方向，记为 $\mathcal{L}_{dir}$。对于分类损失，使用 Facal Loss 函数[191]，定义为

$$\mathcal{L}_{cls} = -\alpha(1 - p^a)^\gamma log p^a \tag{5.34}$$

其中 p^a 表示预测检测框的类别概率。

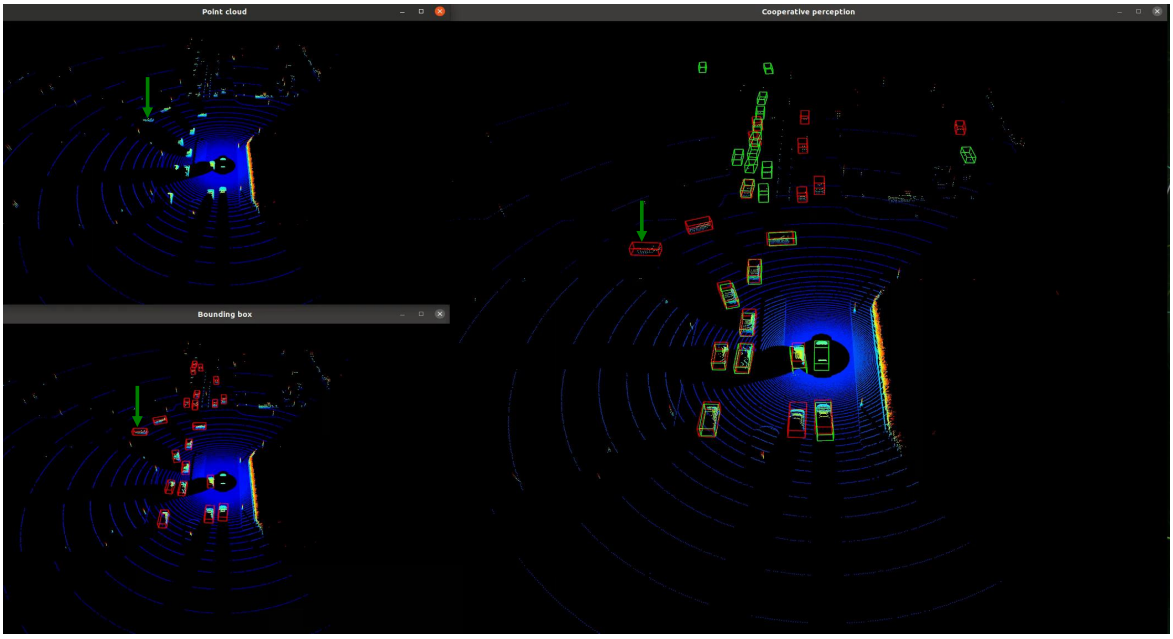
总的损失函数定义如下

$$\mathcal{L} = \frac{1}{N_{pos}}(\beta_{loc}\mathcal{L}_{loc} + \beta_{cls}\mathcal{L}_{cls} + \beta_{dir}\mathcal{L}_{dir}) \tag{5.35}$$

其中 N_{pos} 表示真阳性检测框的数量。

协同感知可视化界面如图5-14所示，图5-14(a)左边表示发送端的点云数据可视化界面，对应/plot_pcd 节点，右边表示点云数据、检测框一起可视化的界面，对应/bbox_visualize 节点，其中红色箭头所指的为接收端车辆的位置，也能看到发送端

(a) 发送端可视化界面



(b) 接收端可视化界面

图 5-14 目标级协同感知可视化界面

Figure 5-14 Visual interface for object-level cooperative perception

的推理模块预测的检测框在这一帧出现了漏检。图5-14(b)左上表示接收端的点云数据可视化界面，对应/plot_pcd 节点，左下表示接收端点云数据、检测框一起可视化的界面，对应/bbox_visualize 节点，右侧界面表示目标级协同感知融合可视化界面，其中绿色箭头所指的车辆即发送端车辆，由于协同感知，接收端车辆扩大了感知范围。

5.3.3 性能评估

协同感知系统的链路时延对于确保多车辆之间实时数据交换和决策至关重要，为评估目标级协同感知系统的链路时延，对各功能节点的处理时延和节点之间的通信时延进行定义，首先是发送端节点的时延定义，点云数据发布节点/pcd_publisher 到点云

推理节点/pcd_infer 之间的通信时延定义为 $d^{sender}_{pcd\rightarrow infer}$，推理节点进行推理并输出检测框的时延定义为 d^{sender}_{infer}，推理节点到检测框数据封装节点/bbox_sender 的通信时延定义为 $d^{sender}_{infer\rightarrow sender}$，检测框数据封装节点的处理时延为 $d^{sender}_{prepare}$，随后发送给下位机 OBU 处理后进行广播，接收端 OBU 接收到并转发到接收端的侦听节点/bbox_listener，定义为 d_{tran}，包含了发送端和接收端 OBU 的处理数据时延以及无线通信时延。接下来是接收端各节点的时延定义，接收端数据侦听模块的处理时延定义为 $d^{receiver}_{listener}$，数据侦听节点到数据融合节点/bbox_receiver 之间的通信时延定义为 $d^{receiver}_{lis\rightarrow rec}$，数据融合处理时延定义为 $d^{receiver}_{fusion}$，接收端的推理节点推理时延为 $d^{receiver}_{infer}$，推理节点到可视化节点/bbox_visualize 的通信时延为 $d^{receiver}_{infer\rightarrow vis}$，可视化节点的处理时延为 $d^{receiver}_{vis}$。所有链路时延定义见图5-15。

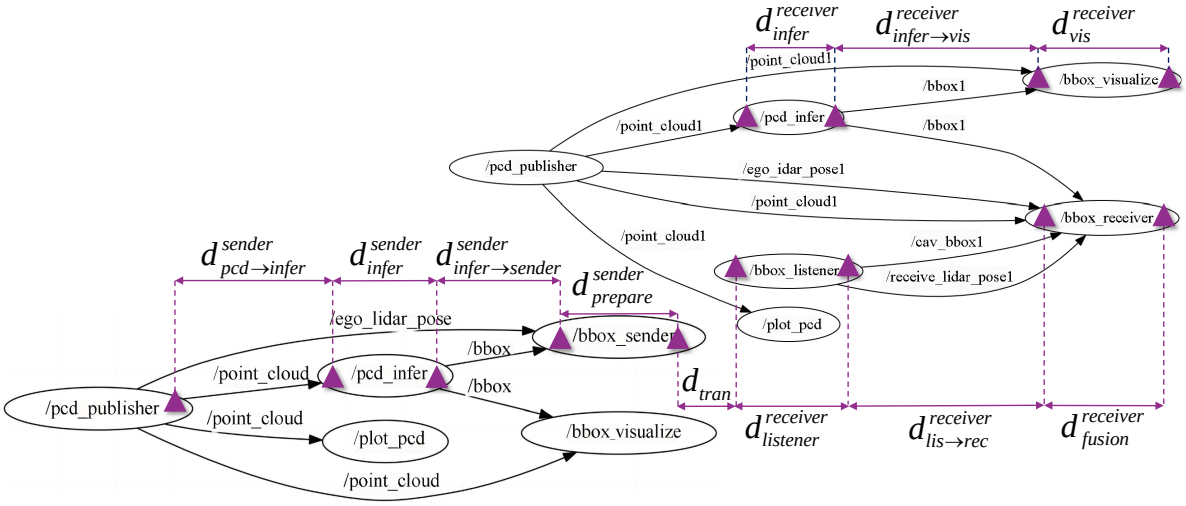


图 5-15 链路时延示意图

Figure 5-15 Illustration of link latency

本节进行多次实验对链路各模块时延进行测试，首先是接收端推理模块及可视化模块等链路时延测试，见表5-3，其中 S^{av}_{pcd} 表示点云数据的平均大小，单位为 MB，S^{av}_{bbox} 表示检测框数据的平均大小，单位为 B，时延的单位为 ms。点云数据平均大小为 0.85 MB，检测框数据平均大小为 1051 B 左右，OBU 要求发送的数据包大小有限制（1470 B），超过限制需要进行分包，若发送原始点云数据，由于数据过大需要进行分包操作，这会造成较大的延迟，而发送检测框的数据每帧只需要一个数据包就能发送。推理模块的处理时延约 100 ms 左右，ROS2 节点之间的通信时间约 20~50 ms，可视化时间约 50 ms，可视化模块的逻辑是，点云数据传输过来先进行缓存，待推理模块输出的检

表 5-3 接收端推理链路时延

Table 5-3 Inference link latency of the receiver

测试数据	数据包总数	$S_{pcd}^{av}\ (MB)$	$S_{bbox}^{av}\ (B)$	$d_{infer}^{receiver}\ (ms)$	$d_{infer\rightarrow vis}^{receiver}(ms)$	$d_{vis}^{receiver}(ms)$
测试一	2151	0.85	1053.37	112.56	34.70	50.01
测试二	1981	0.85	1051.77	99.33	48.32	49.33
测试三	4986	0.85	1052.34	122.74	23.11	49.60
测试四	4242	0.85	1052.93	99.38	47.11	49.62
测试五	4211	0.85	1052.35	105.30	40.43	49.25

表 5-4 协同感知链路时延

Table 5-4 Latency of cooperative perception links

测试数据	丢包率	$d_{pcd\rightarrow infer}^{sender}$	d_{infer}^{sender}	$d_{infer\rightarrow sender}^{sender}$	$d_{prepare}^{sender}$	d_{tran}	$d_{listener}^{receiver}$	$d_{lis\rightarrow rec}^{receiver}$	$d_{fusion}^{receiver}$
测试一	0.092%	5.66	108.39	18.97	20.29	18.02	4.06	8.23	99.72
测试二	0	5.63	109.23	18.82	20.30	32.28	4.16	8.47	100.55
测试三	0	5.75	109.02	18.75	20.28	25.87	3.95	18.48	101.84
测试四	0.047%	5.83	102.71	18.28	20.25	26.19	4.18	8.65	99.85
测试五	0.023%	5.75	108.79	18.91	20.29	35.71	3.99	8.31	101.73

表 5-5 接收端推理链路时延（交换场景数据）

Table 5-5 Inference link latency of the receiver (exchange scene data)

测试数据	数据包总数	$S_{pcd}^{av}\ (MB)$	$S_{bbox}^{av}\ (B)$	$d_{infer}^{receiver}\ (ms)$	$d_{infer\rightarrow vis}^{receiver}(ms)$	$d_{vis}^{receiver}(ms)$
测试六	3270	0.86	1014.83	101.70	46.23	50.89
测试七	3185	0.86	1015.04	118.20	33.69	51.49
测试八	4716	0.86	1015.74	101.72	48.71	51.64

测框传输过来后开始可视化。此次测试中发送端到接收端的协同链路时延见表5-4，两表的测试数据相同，其中两次测试没有产生丢包。发送端的推理模块时延约 100 ms 左右，与接收端的推理模块时延一致，原因在于发送端和接收端的计算设备相同且功率一致。不同 ROS2 节点之间的通信时延不同，主要取决于传输的消息类型，发送端的 $d_{pcd\rightarrow infer}^{sender}$ 与 $d_{infer\rightarrow sender}^{sender}$ 与接收端的 $d_{lis\rightarrow rec}^{receiver}$ 时延相差较大，ROS2 节点中的消息传输

表 5-6 协同感知链路时延（交换场景数据）

Table 5-6 Latency of cooperative perception links (exchange scene data)

测试数据	丢包率	$d_{pcd\rightarrow infer}^{sender}$	d_{infer}^{sender}	$d_{infer\rightarrow sender}^{sender}$	$d_{prepare}^{sender}$	d_{tran}	$d_{listener}^{receiver}$	$d_{lis\rightarrow rec}^{receiver}$	$d_{fusion}^{receiver}$
测试六	0	5.64	107.65	13.57	20.27	17.92	3.88	9.59	102.96
测试七	0	5.83	102.46	13.24	20.26	22.98	4.03	8.03	100.96
测试八	0	5.77	107.29	13.85	20.27	22.83	3.15	19.31	102.56

时延均小于 20 ms。发送端检测框封装到接收端解封装的时延 d_{tran} 小于 40 ms，这其中包括了 OBU 之间无线传输的时延，OBU 数据包的发送间隔最小设定为 10 ms。接收端的协同感知融合时延 $d_{fusion}^{receiver}$ 达到了 100 ms 左右，融合模块涉及到了接收端车辆的点云数据、检测框数据需要预先进行缓存，待侦听到发送端的检测框和雷达位姿数据后，进行坐标系的转换以及可视化，点云和检测框可视化的模块约 50 ms 左右，加上缓存等待、计算等造成融合模块的处理时延较高。协同链路总平均时延约 300 ms，包括了从发送端处理点云数据开始，经过推理、封装、发送，接收端侦听接收，融合过程。测试过程发送端和接收端计算单元的运行功率设置为 50 W，其 GPU 负载达到 99%，若设置低功耗模式，则会增加推理过程产生的时延。将发送端和接收端运行的场景数据进行交换后看测试结果是否会有较大变化，得到接收端的推理链路时延见表5-5，协同感知链路时延见表5-6。可以看到，交换场景数据后，链路各模块处理时延没有较大变化，运行稳定，得出的结论与未交换前一致。根据基于 C-V2X 的智能化网联化融合发展路线图（征求意见稿）[125] 中对网联化技术等级的描述，车辆智能化网联化融合将由深入浅的经历三个发展阶段，车路云一体化提醒预警、车路云一体化的辅助驾驶、车路云一体化的自动驾驶，在第一阶段，对协同感知的网联性能指标描述为，应用层端到端时延要求较低、消息可靠性要求较低、感知精度要求较低，此阶段无协同决策和协同控制。目标级协同感知系统能够实现第一阶段描述的典型场景即感知数据共享、弱势交通参与者安全通行场景。目标级协同感知以提醒、预警方式在人机交互界面显示，弥补人类驾驶员和单车智能在感知能力上的局限。

5.4 本章小结

本章研究了网联自动驾驶协同位姿优化及协同感知系统，协同位姿优化基于协同的感知信息，两者互相促进，提升系统的综合性能。协同位姿优化建立了基于因子图的约束模型框架，建立车-车、车-特征、车-目标之间最小化残差的状态估计问题，使用列文伯格-马夸尔特算法求解最小二乘问题，随后在特征稀疏和密集场景进行仿真，分析不同约束条件变化时对协同位姿性能的影响。协同位姿优化基于协同感知结果，尤其是存在感知误差时能够进行位姿优化。进一步对不同融合策略的协同感知进行仿真，分析不同层级融合的性能优劣，搭建硬件在环平台，开发对应的软件环境，用于测试和评估目标级协同感知系统性能。

结论与展望

研究结论

车联网技术具备“非视距、全天候”、“上帝视角”、“可协同”等特点，融合车联网技术能够有效弥补单车智能的驾驶自动化功能不足，对单车及基础设施进行赋能，实现能力增强和能力拓展。随着智能网联汽车渗透率的不断提高，由此产生的车载信息娱乐、传感器、车路协同、地图等数据传输、处理、存储的需求极大地增加了网络负荷，并对网络时延、带宽、可靠性提出了更高的需求。车载边缘计算系统与车联网技术融合能够减少数据传输路由长度，降低端到端的通信时延，同时作为本地服务托管环境，提供强大的计算、存储资源进行协同计算，处理各种复杂需求场景。本文主要围绕程序关联性任务和协同感知任务代表性需求场景，面临资源供应不确定性和动态变化性挑战，探究移动环境中通信、计算、存储的作用机理，从系统宏观层面的资源调度分配到系统局部的实时资源优化，进一步聚焦到具体的协作感知任务效果上，进行位姿优化和开发目标级协同感知系统原型，资源分配提供合理的计算环境，协同感知进一步提升协同质量，提供案例支撑并相互迭代，提升车载边缘计算系统中多类型任务场景需求下资源分配合理性和自适应性。本文的主要研究内容总结如下:

（1）面向程序关联性任务的车载边缘计算系统资源配置优化。车载边缘计算系统中资源供应端计算、存储差异性较大且供应呈现不确定性和动态变化性，程序关联性任务的缓存与卸载决策耦合性以及长期联合决策的需求增加了进行资源分配的复杂性。为应对以上挑战，首先建立程序关联性任务计算卸载流程，充分利用车端、边缘服务器端和云端之间的协作，根据任务卸载流程建立六种情形下的服务缓存以及对应的卸载模型，对不同情形下的计算时延、传输时延、能量消耗进行推导，并分析单次决策下不同情形任务完成时延和能量消耗的极值。在此基础上形式化定义了最小化长期累计任务处理时延的协同资源优化问题。进一步将该问题描述为马尔可夫决策过程，提出基于深度确定性策略梯度的联合服务缓存与计算卸载方案，并设计演员与评论家的网络结构。仿真实验评估了所提方案与基线方案性能的收敛性，车辆密度、任务大小、计算频率等对方案性能的影响。结果表明所提的联合服务缓存与计算卸载方案在边缘计算系统中能够尽量避免单次做出最大任务处理时延的缓存决策，有效减少任务计算完成时延。

（2）面向网联自动驾驶的协同感知任务资源配置优化。自动驾驶感知任务对资源需求较高且时延容忍严格，协同感知任务面临感知模型差异性、算力差异性、协同策略差异性以及任务组合性。为应对以上挑战，提出车载边缘计算系统下的协同感知任务需求与计算服务场景，建立自车进行协同的感兴趣域任务计算、卸载、结果反馈系统模型，推导任务计算完成时延，考虑容忍时延、算力以及卸载约束，形式化定义了最小化协同感知任务完成时延优化问题。为求解混合整数非线性规划问题，将离散变量和连续变量解耦，提出双层最优任务卸载与资源分配算法，外层通过搜索和可行域检查减少优化问题的非凸性并确定资源分配变量的范围，内层为凸优化问题通过非线性优化器求解得到最优卸载决策与资源分配方案。仿真实验评估了车辆计算资源、任务车辆数量、路侧单元计算资源和无线通信带宽对所提方案与基线方案的性能影响，结果表明所提算法在车载边缘计算系统中能够适应不同需求差异性，为保障协同感知任务完成计算提供最优资源环境。

（3）面向网联自动驾驶的协同位姿优化及应用。车联网协同感知任务依赖准确的位姿关系，实际会面临非理想通信环境和存在各种传感器误差的情况，导致位姿估计错误，降低协作性能。为应对该挑战，建立车-车、车-特征、车-动态目标、地图、全球卫星导航定位系统的因子图约束模型，形式化定义了最小化状态观测量与预测量的残差优化问题，采用列文伯格-马夸尔特法求解状态估计问题，仿真实验评估了各约束条件和传感器精度对状态估计性能的影响，协同感知形成的约束越多，能够增强自车与目标的协同定位性能，提升协同感知效果。为推动自动驾驶协同感知研究向实际应用迈进，基于协同感知仿真框架评估了不同融合策略的性能优劣，由于目标级协同感知具备良好的兼容性和可拓展性，基于 C-V2X 通信设备和边缘计算单元搭建目标级协同感知在环试验平台，并开发配套的软件环境部署检测模型和融合算法，对各功能模块实验并对链路时延进行测试，其性能满足车辆智能化网联化融合的第一发展阶段要求，实现提醒预警，弥补人类驾驶员和单车智能在感知能力上的局限。

未来研究展望

本文主要针对车载边缘计算中的资源分配和协同感知技术开展了研究，并取得了一定的成果。车载边缘计算系统融合 C-V2X 网络，为车路云一体化方案提供多连接、全场景、低时延、高可靠的服务保障，目前尚处于早期探索阶段，本文工作无法解决所

有挑战，有待进一步探索和解决。总结本文的研究成果和经验，作者认为未来的研究可以从以下几个方面展开：

（1）在面向程序关联性任务的车载边缘计算系统资源配置优化研究中，仅考虑了多用户单任务动态请求计算需求，实际场景中存在多类型任务需求且满足不同指标要求。此外，未考虑通信之间的干扰，基于深度强化学习的服务缓存与计算卸载方案采用集中式训练和集中式决策，基于多智能体的分布式训练和分布式决策可能更加适用车路云一体化系统。基于深度强化学习的服务缓存与计算卸载方案网络结构的可拓展性和鲁棒性还值得进一步研究。

（2）在面向网联自动驾驶的协同感知任务资源配置优化研究中，对车辆感知能力的假设较为简单，实际场景中感知检测有效范围可能是动态变化的，并且需要考虑检测精度。基于特定的感兴趣区域选择协同车辆进行资源配置，能够进一步提高协同感知的效率和资源使用率。感知结果的量化能够为计算资源分配提供更多的决策信息。

（3）在面向网联自动驾驶的协同位姿优化研究中，未考虑时序异步的情况，考虑时序异步情形需要进一步研究相应的滤波或跟踪算法处理数据。车车协同时，进一步考虑共同感知目标，引入位姿图优化是一个值得探索的方向。在协同感知案例实践中，仅考虑了兼容性和可拓展性好的目标级协同感知，此外，各模块的功能代码还需要进一步精简和优化，考虑真实上车场景中存在的各种问题进行功能拓展。协同感知实际场景下还存在很多值得研究的课题，考虑定位误差、通信时延、隐私、异构传感器模态或感知模型等问题下的协同感知。原始数据级和特征数据级协同感知需要更多实践案例支撑，不同级别融合算法均需要进行实车测试和性能验证。

参考文献

[1] 陈山枝. 蜂窝车联网 (C-V2X) 及其赋能智能网联汽车发展的辩思与建议[J]. 电信科学, 2022, 38(7): 1-17.

[2] IMT-2020(5G) 推进组. C-V2X 与单车智能融合功能及应用白皮书[R/OL]. (2023-11-07)[2024-01-09]. http://v2x.caict.ac.cn/achievementdetail.html?id=51category=1.

[3] 陈山枝, 胡金玲, 赵丽, 等. 蜂窝车联网 (C-V2X)[M]. 北京: 人民邮电出版社, 2023.

[4] CHEN S, HU J, ZHAO L, et al. Cellular vehicle-to-everything (C-V2X)[M]. London: Springer Nature, 2023.

[5] CHEN S, HU J, SHI Y, et al. LTE-V: A TD-LTE-based V2X solution for future vehicular network[J]. IEEE Internet of Things Journal, 2016, 3(6): 997-1005.

[6] CHEN S, HU J, SHI Y, et al. A vision of C-V2X: Technologies, field testing, and challenges with chinese development[J]. IEEE Internet of Things Journal, 2020, 7(5): 3872-3881.

[7] 工业和信息化部. 车联网（智能网联汽车）直连通信使用 5905-5925MHz 频段管理规定（暂行）[R/OL]. (2023-10-25)[2024-01-11]. https://www.gov.cn/zhengce/zhengceku/2018-12/31/content_5442658.htm.

[8] IMT-2020(5G) 推进组 C-V2X 工作组. 车联网典型应用案例集 (2023 年)[R/OL]. (2023-11-07)[2024-01-11]. http://v2x.caict.ac.cn/achievementdetail.html?id=52&category=2.

[9] 中国智能网联汽车产业创新联盟. 车路云一体化系统白皮书[R/OL]. (2023-01-11)[2024-01-11]. http://www.caicv.org.cn/index.php/newsInfo?id=936.

[10] 中国通信学会. C-V2X 车联网技术发展与产业实践白皮书[R/OL]. (2022-07-06)[2024-01-11]. https://www.china-cic.cn/upload/202212/11/336cb372567c4013964be4ef7d639d6b.pdf.

[11] 中国信息通信研究院. 车联网白皮书 (2023 年)[R/OL]. (2023-12-03)[2024-01-11]. http://www.caict.ac.cn/kxyj/qwfb/bps/202312/P020231221344824574866.pdf.

[12] 清华大学智能产业研究院、百度 Apollo. 面向自动驾驶的车路协同关键技术与展望 2.0[R/OL]. (2022-12-28)[2024-01-11]. https://air.tsinghua.edu.cn/info/1007/1917

.htm.

[13] 丁飞, 张楠, 李升波, 等. 智能网联车路云协同系统架构与关键技术研究综述[J]. 自动化学报, 2022, 48(12): 2863-2885.

[14] LI H, CHEN Y, LI K, et al. Transportation internet: A sustainable solution for intelligent transportation systems[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(12): 15818-15829.

[15] XU Q, LI K, WANG J, et al. The status, challenges, and trends: An interpretation of technology roadmap of intelligent and connected vehicles in china (2020)[J]. Journal of Intelligent and Connected Vehicles, 2022, 5(1): 1-7.

[16] HUANG Z, BAI L, SUN M, et al. A mixed-bouncing based non-stationarity and consistency 6G V2V channel model with continuously arbitrary trajectory[J]. IEEE Transactions on Wireless Communications, 2023, 23(2): 1634-1650.

[17] CHENG X, DUAN D, GAO S, et al. Integrated sensing and communications (ISAC) for vehicular communication networks (VCN)[J]. IEEE Internet of Things Journal, 2022, 9(23): 23441-23451.

[18] 程翔, 张浩天, 李思江, 等. 机器联觉: 通信与多模态感知的智能融合[J]. 模式识别与人工智能, 2023, 36(11): 967-986.

[19] FANG Y, MIN H, WU X, et al. On-ramp merging strategies of connected and automated vehicles considering communication delay[J]. IEEE Transactions on Intelligent Transportation Systems, 2022, 23(9): 15298-15312.

[20] YUE W, LI C, MAO G, et al. Evolution of road traffic congestion control: A survey from perspective of sensing, communication, and computation[J]. China Communications, 2021, 18(12): 151-177.

[21] MAO G, HUI Y, REN X, et al. The internet of things for smart roads: A road map from present to future road infrastructure[J]. IEEE Intelligent Transportation Systems Magazine, 2022, 14(6): 66-76.

[22] SU C, GUO J, DONG Y, et al. Strategy-proof computational resource reservation based on dynamic matching for vehicular edge computing[J]. IEEE Internet of Things Journal,

2023, 11(9): 15602-15615.

[23] LANG P, TIAN D, DUAN X, et al. Blockchain-based cooperative computation offloading and secure handover in vehicular edge computing networks[J]. IEEE Transactions on Intelligent Vehicles, 2023, 8(7): 3839-3853.

[24] JU Y, CHEN Y, CAO Z, et al. Joint secure offloading and resource allocation for vehicular edge computing network: A multi-agent deep reinforcement learning approach [J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(5): 5555-5569.

[25] LANG P, TIAN D, DUAN X, et al. Cooperative computation offloading in blockchain-based vehicular edge computing networks[J]. IEEE Transactions on Intelligent Vehicles, 2022, 7(3): 783-798.

[26] JIA Y, MAO R, SUN Y, et al. MASS: Mobility-aware sensor scheduling of cooperative perception for connected automated driving[J]. IEEE Transactions on Vehicular Technology, 2023, 72(11): 14962-14977.

[27] NAN Z, ZHOU S, JIA Y, et al. Joint task offloading and resource allocation for vehicular edge computing with result feedback delay[J]. IEEE Transactions on Wireless Communications, 2023, 22(10): 6547-6561.

[28] SUN Y, XIE B, ZHOU S, et al. MEET: Mobility-enhanced edge intelligence for smart and green 6G networks[J]. IEEE Communications Magazine, 2023, 61(1): 64-70.

[29] SUN Y, ZHOU S, NIU Z. Distributed task replication for vehicular edge computing: Performance analysis and learning-based algorithm[J]. IEEE Transactions on Wireless Communications, 2021, 20(2): 1138-1151.

[30] XU X, LIU K, DAI P, et al. Cooperative sensing and heterogeneous information fusion in VCPS: A multi-agent deep reinforcement learning approach[J]. IEEE Transactions on Intelligent Transportation Systems, 2023: 1-16.

[31] LIU K, XIAO K, DAI P, et al. Fog computing empowered data dissemination in software defined heterogeneous VANETs[J]. IEEE Transactions on Mobile Computing, 2021, 20 (11): 3181-3193.

[32] REN H, LIU K, YAN G, et al. A memetic algorithm for cooperative complex task

offloading in heterogeneous vehicular networks[J]. IEEE Transactions on Network Science and Engineering, 2023, 10(1): 189-204.

[33] HUANG N, DOU C, WU Y, et al. Mobile edge computing aided integrated sensing and communication with short-packet transmissions[J]. IEEE Transactions on Wireless Communications, 2023: 1-18.

[34] YE X, QU K, ZHUANG W, et al. Accuracy-aware cooperative sensing and computing for connected autonomous vehicles[J]. IEEE Transactions on Mobile Computing, 2023: 1-15.

[35] SHI W, CAO J, ZHANG Q, et al. Edge computing: Vision and challenges[J]. IEEE Internet of Things Journal, 2016, 3(5): 637-646.

[36] XIAO H, ZHAO J, FENG J, et al. Joint optimization of security strength and resource allocation for computation offloading in vehicular edge computing[J]. IEEE Transactions on Wireless Communications, 2023, 22(12): 8751-8765.

[37] XIAO H, CAI L, FENG J, et al. Resource optimization of MAB-based reputation management for data trading in vehicular edge computing[J]. IEEE Transactions on Wireless Communications, 2023, 22(8): 5278-5290.

[38] LUO Q, LI C, LUAN T H, et al. Minimizing the delay and cost of computation offloading for vehicular edge computing[J]. IEEE Transactions on Services Computing, 2022, 15 (5): 2897-2909.

[39] ABBAS N, ZHANG Y, TAHERKORDI A, et al. Mobile edge computing: A survey[J]. IEEE Internet of Things Journal, 2018, 5(1): 450-465.

[40] WANG X, ZHU H, NING Z, et al. Blockchain intelligence for internet of vehicles: Challenges and solutions[J]. IEEE Communications Surveys & Tutorials, 2023, 25(4): 2325-2355.

[41] CAI G, FAN B, DONG Y, et al. Task-efficiency oriented V2X communications: Digital twin meets mobile edge computing[J]. IEEE Wireless Communications, 2023, 31(2): 149-155.

[42] ZHANG K, CAO J, ZHANG Y. Adaptive digital twin and multiagent deep reinforce-

ment learning for vehicular edge computing and networks[J]. IEEE Transactions on Industrial Informatics, 2022, 18(2): 1405-1413.

[43] GU H, ZHAO L, HAN Z, et al. AI-Enhanced Cloud-Edge-Terminal collaborative network: Survey, applications, and future directions[J]. IEEE Communications Surveys & Tutorials, 2023: 1-64.

[44] ZHANG S, LI J, SHI L, et al. Federated learning in intelligent transportation systems: Recent applications and open problems[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 25(5): 3259-3285.

[45] FANG C, HU Z, MENG X, et al. DRL-driven joint task offloading and resource allocation for energy-efficient content delivery in cloud-edge cooperation networks[J]. IEEE Transactions on Vehicular Technology, 2023, 72(12): 16195-16207.

[46] AHMED M, MIRZA M A, RAZA S, et al. Vehicular communication network enabled CAV data offloading: A review[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(8): 7869-7897.

[47] ZHOU Y, YU F R, REN M, et al. Adaptive data transmission and computing for vehicles in the Internet-of-Intelligence[J]. IEEE Transactions on Vehicular Technology, 2023, 73(2): 2533-2548.

[48] LIU J, LI Y, SUN R, et al. SDSS: Secure data sharing scheme for edge enabled iov networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(11): 12038-12049.

[49] GONG Y, WEI Y, FENG Z, et al. Resource allocation for integrated sensing and communication in digital twin enabled internet of vehicles[J]. IEEE Transactions on Vehicular Technology, 2023, 72(4): 4510-4524.

[50] MAO B, LIU Y, LIU J, et al. AI-assisted edge caching for metaverse of connected and automated vehicles: Proposal, challenges, and future perspectives[J]. IEEE Vehicular Technology Magazine, 2023, 18(4): 66-74.

[51] ZHU Y, MAO B, KATO N. Intelligent reflecting surface in 6G vehicular communications: A survey[J]. IEEE Open Journal of Vehicular Technology, 2022, 3: 266-277.

[52] 吕品, 许嘉, 李陶深, 等. 面向自动驾驶的边缘计算技术研究综述[J]. 通信学报, 2021, 42(3): 190-208.

[53] NADEAU T D, GRAY K. SDN: Software defined networks: An authoritative review of network programmability technologies[M]. O'Reilly Media, Inc., 2013.

[54] CHIANG Y, ZHANG Y, LUO H, et al. Management and orchestration of edge computing for IoT: A comprehensive survey[J]. IEEE Internet of Things Journal, 2023, 10 (16): 14307-14331.

[55] MENEGUETTE R, DE GRANDE R, UEYAMA J, et al. Vehicular edge computing: Architecture, resource management, security, and challenges[J]. ACM Computing Surveys, 2021, 55(1): 1-46.

[56] SUN F, HOU F, CHENG N, et al. Cooperative task scheduling for computation offloading in vehicular cloud[J]. IEEE Transactions on Vehicular Technology, 2018, 67(11): 11049-11061.

[57] ZHOU Z, LIU P, FENG J, et al. Computation resource allocation and task assignment optimization in vehicular fog computing: A contract-matching approach[J]. IEEE Transactions on Vehicular Technology, 2019, 68(4): 3113-3125.

[58] NING Z, HUANG J, WANG X. Vehicular fog computing: Enabling real-time traffic management for smart cities[J]. IEEE Wireless Communications, 2019, 26(1): 87-93.

[59] QIAO G, LENG S, ZHANG K, et al. Collaborative task offloading in vehicular edge multi-access networks[J]. IEEE Communications Magazine, 2018, 56(8): 48-54.

[60] YU R, HUANG X, KANG J, et al. Cooperative resource management in cloud-enabled vehicular networks[J]. IEEE Transactions on Industrial Electronics, 2015, 62(12): 7938-7951.

[61] NOOR-A-RAHIM M, LIU Z, LEE H, et al. A survey on resource allocation in vehicular networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 23(2): 701-721.

[62] JIANG X, YU F R, SONG T, et al. Resource allocation of video streaming over vehicular networks: a survey, some research issues and challenges[J]. IEEE Transactions on

Intelligent Transportation Systems, 2021, 23(7): 5955-5975.

[63] GUO C, LIANG L, LI G Y. Resource allocation for vehicular communications with low latency and high reliability[J]. IEEE Transactions on Wireless Communications, 2019, 18(8): 3887-3902.

[64] LIANG L, YE H, LI G Y. Spectrum sharing in vehicular networks based on multi-agent reinforcement learning[J]. IEEE Journal on Selected Areas in Communications, 2019, 37(10): 2282-2292.

[65] WU W, LIU R, YANG Q, et al. Robust resource allocation for vehicular communications with imperfect CSI[J]. IEEE Transactions on Wireless Communications, 2021, 20(9): 5883-5897.

[66] YANG H, ZHANG K, ZHENG K, et al. Joint frame design and resource allocation for ultra-reliable and low-latency vehicular networks[J]. IEEE Transactions on Wireless Communications, 2020, 19(5): 3607-3622.

[67] CHEN J, WU H, LYU F, et al. Adaptive resource allocation for diverse safety message transmissions in vehicular networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2021, 23(8): 13482-13497.

[68] HE Y, WANG Y, LIN Q, et al. Meta-hierarchical reinforcement learning (MHRL)-based dynamic resource allocation for dynamic vehicular networks[J]. IEEE Transactions on Vehicular Technology, 2022, 71(4): 3495-3506.

[69] DAI Y, XU D, ZHANG K, et al. Deep reinforcement learning and permissioned blockchain for content caching in vehicular edge computing and networks[J]. IEEE Transactions on Vehicular Technology, 2020, 69(4): 4312-4324.

[70] CHEN J, WU H, YANG P, et al. Cooperative edge caching with location-based and popular contents for vehicular networks[J]. IEEE Transactions on Vehicular Technology, 2020, 69(9): 10291-10305.

[71] XUE Z, LIU Y, HAN G, et al. Two-layer distributed content caching for infotainment applications in VANETs[J]. IEEE Internet of Things Journal, 2021, 9(3): 1696-1711.

[72] GUPTA D, RANI S, SINGH A, et al. ICN based efficient content caching scheme for

vehicular networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2022.

[73] YANG P, ZHANG N, ZHANG S, et al. Content popularity prediction towards location-aware mobile edge caching[J]. IEEE Transactions on Multimedia, 2018, 21(4): 915-929.

[74] TIAN H, XU X, QI L, et al. CoPace: Edge computation offloading and caching for self-driving with deep reinforcement learning[J]. IEEE Transactions on Vehicular Technology, 2021, 70(12): 13281-13293.

[75] WU J, WANG J, CHEN Q, et al. Resource allocation for delay-sensitive vehicle-to-multi-edges (V2Es) communications in vehicular networks: A multi-agent deep reinforcement learning approach[J]. IEEE Transactions on Network Science and Engineering, 2021, 8(2): 1873-1886.

[76] TANG C, ZHU C, WU H, et al. Toward response time minimization considering energy consumption in caching-assisted vehicular edge computing[J]. IEEE Internet of Things Journal, 2021, 9(7): 5051-5064.

[77] LAN D, TAHERKORDI A, ELIASSEN F, et al. Deep reinforcement learning for computation offloading and caching in fog-based vehicular networks[C]//IEEE International Conference on Mobile Ad Hoc and Sensor Systems. IEEE, 2020: 622-630.

[78] LI Z, YANG C, HUANG X, et al. CoOR: Collaborative task offloading and service caching replacement for vehicular edge computing networks[J]. IEEE Transactions on Vehicular Technology, 2023, 72(7): 9676-9681.

[79] LIU J, AHMED M, MIRZA M A, et al. RL/DRL meets vehicular task offloading using edge and vehicular cloudlet: A survey[J]. IEEE Internet of Things Journal, 2022, 9(11): 8315-8338.

[80] ZHU X, LUO Y, LIU A, et al. Multiagent deep reinforcement learning for vehicular computation offloading in IoT[J]. IEEE Internet of Things Journal, 2020, 8(12): 9763-9773.

[81] QI Q, WANG J, MA Z, et al. Knowledge-driven service offloading decision for vehicular edge computing: A deep reinforcement learning approach[J]. IEEE Transactions on

Vehicular Technology, 2019, 68(5): 4192-4203.

[82] SHANG B, LIU L, TIAN Z. Deep learning-assisted energy-efficient task offloading in vehicular edge computing systems[J]. IEEE Transactions on Vehicular Technology, 2021, 70(9): 9619-9624.

[83] KAZMI S A, OTOUM S, HUSSAIN R, et al. A novel deep reinforcement learning-based approach for task-offloading in vehicular networks[C]//IEEE Global Communications Conference. IEEE, 2021: 1-6.

[84] 许世琳. 车联网中基于深度强化学习的计算任务卸载策略研究[D]. 北京: 北京邮电大学, 2021.

[85] 王妍聪. 边缘计算与车联网融合的应用技术和算法研究[D]. 吉林: 吉林大学, 2023.

[86] FAN W, SU Y, LIU J, et al. Joint task offloading and resource allocation for vehicular edge computing based on V2I and V2V modes[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(4): 4277-4292.

[87] 刘杰. 面向车联网边缘计算的联合任务卸载与资源管理技术研究[D]. 北京: 北京邮电大学, 2024: 11-34.

[88] FAN W, ZHANG Y, ZHOU G, et al. Deep reinforcement learning-based task offloading for vehicular edge computing with flexible RSU-RSU cooperation[J]. IEEE Transactions on Intelligent Transportation Systems, 2024: 1-14.

[89] NAN Z, ZHOU S, JIA Y, et al. Joint task offloading and resource allocation for vehicular edge computing with result feedback delay[J]. IEEE Transactions on Wireless Communications, 2023, 22(10): 6547-6561.

[90] FAN X, GU W, LONG C, et al. Optimizing task offloading and resource allocation in vehicular edge computing based on heterogeneous cellular networks[J]. IEEE Transactions on Vehicular Technology, 2023, 73(5): 7175-7187.

[91] ZHANG J, WU Y, MIN G, et al. Neural network-based game theory for scalable offloading in vehicular edge computing: A transfer learning approach[J]. IEEE Transactions on Intelligent Transportation Systems, 2024: 1-14.

[92] LIN Z, CHEN X, HE X, et al. Energy-efficient cooperative task offloading in NOMA-

enabled vehicular fog computing[J]. IEEE Transactions on Intelligent Transportation Systems, 2024: 1-14.

[93] YIN L, LUO J, QIU C, et al. Joint task offloading and resources allocation for hybrid vehicle edge computing systems[J]. IEEE Transactions on Intelligent Transportation Systems, 2024: 1-14.

[94] ZHU H, WU Q, WU X J, et al. Decentralized power allocation for MIMO-NOMA vehicular edge computing based on deep reinforcement learning[J]. IEEE Internet of Things Journal, 2021, 9(14): 12770-12782.

[95] XU X, LIU K, DAI P, et al. Joint task offloading and resource optimization in NOMA-based vehicular edge computing: A game-theoretic DRL approach[J]. Journal of Systems Architecture, 2023, 134: 102780.

[96] WANG Z, MU X, LIU Y, et al. NOMA-aided joint communication, sensing, and multi-tier computing systems[J]. IEEE Journal on Selected Areas in Communications, 2022, 41(3): 574-588.

[97] HU S, FANG Z, DENG Y, et al. Collaborative perception for connected and autonomous driving: Challenges, possible solutions and opportunities[J]. arXiv preprint arXiv:2401.01544, 2024.

[98] HUANG T, LIU J, ZHOU X, et al. V2X cooperative perception for autonomous driving: Recent advances and challenges[J]. arXiv preprint arXiv:2310.03525, 2023.

[99] LIU S, GAO C, CHEN Y, et al. Towards Vehicle-to-Everything autonomous driving: A survey on collaborative perception[J]. arXiv preprint arXiv:2308.16714, 2023.

[100] HAN Y, ZHANG H, LI H, et al. Collaborative perception in autonomous driving: Methods, datasets and challenges[J]. arXiv preprint arXiv:2301.06262, 2023.

[101] BAI Z, WU G, BARTH M J, et al. A survey and framework of cooperative perception: From heterogeneous singleton to hierarchical cooperation[J]. arXiv preprint arXiv:2208.10590, 2022.

[102] 毛瑞清, 贾宇宽, 孙宇璇, 等. 定位与通信受限的网联协同感知算法[J]. 模式识别与人工智能, 2023, 36(11): 1019-1028.

[103] XU R, XIANG H, XIA X, et al. OPV2V: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication[C]//International Conference on Robotics and Automation. IEEE, 2022: 2583-2589.

[104] XU R, XIANG H, TU Z, et al. V2X-VIT: Vehicle-to-everything cooperative perception with vision transformer[C]//European Conference on Computer Vision. Springer, 2022: 107-124.

[105] LI Y, MA D, AN Z, et al. V2X-Sim: Multi-agent collaborative perception dataset and benchmark for autonomous driving[J]. IEEE Robotics and Automation Letters, 2022, 7 (4): 10914-10921.

[106] MAO R, GUO J, JIA Y, et al. DOLPHINS: Dataset for collaborative perception enabled harmonious and interconnected self-driving[C]//Asian Conference on Computer Vision. 2022: 4361-4377.

[107] XU R, XIA X, LI J, et al. V2V4Real: A real-world large-scale dataset for Vehicle-to-Vehicle cooperative perception[C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 13712-13722.

[108] YU H, LUO Y, SHU M, et al. DAIR-V2X: A large-scale dataset for vehicle-infrastructure cooperative 3D object detection[C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 21361-21370.

[109] XIANG H, ZHENG Z, XIA X, et al. V2X-Real: a largs-scale dataset for vehicle-to-everything cooperative perception[J]. arXiv preprint arXiv:2403.16034, 2024.

[110] CHENG X, HUANG Z, BAI L, et al. M3SC: A generic dataset for mixed multi-modal (MMM) sensing and communication integration[J]. China Communications, 2023, 20 (11): 13-29.

[111] XU R, GUO Y, HAN X, et al. OpenCDA: an open cooperative driving automation framework integrated with co-simulation[C]//IEEE International Intelligent Transportation Systems Conference. IEEE, 2021: 1155-1162.

[112] XU R, XIANG H, HAN X, et al. The OpenCDA open-source ecosystem for cooperative driving automation research[J]. IEEE Transactions on Intelligent Vehicles, 2023, 8(4):

2698-2711.

[113] LU Y, HU Y, ZHONG Y, et al. An extensible framework for open heterogeneous collaborative perception[C]//The Twelfth International Conference on Learning Representations. 2024.

[114] CHEN Q, TANG S, YANG Q, et al. Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds[C]//IEEE International Conference on Distributed Computing Systems. IEEE, 2019: 514-524.

[115] CHEN Q, MA X, TANG S, et al. F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds[C]//ACM/IEEE Symposium on Edge Computing. 2019: 88-100.

[116] WANG T H, MANIVASAGAM S, LIANG M, et al. V2VNet: Vehicle-to-Vehicle communication for joint perception and prediction[C]//European Conference on Computer Vision. Springer, 2020: 605-621.

[117] LI Y, REN S, WU P, et al. Learning distilled collaboration graph for multi-agent perception[J]. Advances in Neural Information Processing Systems, 2021, 34: 29541-29552.

[118] HU Y, FANG S, LEI Z, et al. Where2comm: Communication-efficient collaborative perception via spatial confidence maps[J]. Advances in Neural Information Processing Systems, 2022, 35: 4874-4886.

[119] LIU Y C, TIAN J, GLASER N, et al. When2com: Multi-agent perception via communication graph grouping[C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 4106-4115.

[120] YANG K, YANG D, ZHANG J, et al. What2comm: Towards communication-efficient collaborative perception via feature decoupling[C]//ACM International Conference on Multimedia. 2023: 7686-7695.

[121] YANG D, YANG K, WANG Y, et al. How2comm: Communication-efficient and collaboration-pragmatic multi-agent perception[J]. Advances in Neural Information Processing Systems, 2024, 36.

[122] LU Y, LI Q, LIU B, et al. Robust collaborative 3D object detection in presence of pose

errors[C]//IEEE International Conference on Robotics and Automation. IEEE, 2023: 4812-4818.

[123] LUO G, SHAO C, CHENG N, et al. EdgeCooper: Network-aware cooperative LiDAR perception for enhanced vehicular awareness[J]. IEEE Journal on Selected Areas in Communications, 2024, 42(1): 207-222.

[124] 中国公路学会. 车路协同自动驾驶系统（车路云一体化系统）协同发展框架[R/OL]. (2023-01-07)[2024-01-11]. https://www.china-cic.cn/upload/202303/24/1908679ab41a4463aa7e077170aa1c80.pdf.

[125] 中国智能网联汽车产业创新联盟. 基于 C-V2X 的智能化网联化融合发展路线图（征求意见稿）[R/OL]. (2023-09-27)[2024-01-11]. http://www.caicv.org.cn/index.php/newsInfo?id=1496.

[126] 雷凯茹, 余冰雁. 面向车联网的 MEC 跨域协同技术研究[J]. 信息通信技术与政策, 2024, 50(3): 18.

[127] 中国联通. 运营商赋能车联网能力白皮书[R/OL]. (2022-12-11)[2024-01-11]. http://221.179.172.81/images/20221222/87901671694922944.pdf.

[128] 周镖华. 基于博弈论的车联网边缘计算任务卸载策略研究[D]. 广州: 广东工业大学, 2023: 12-15.

[129] AHMED M, RAZA S, MIRZA M A, et al. A survey on vehicular task offloading: Classification, issues, and challenges[J]. Journal of King Saud University-Computer and Information Sciences, 2022, 34(7): 4135-4162.

[130] 王琦，杨毅远，江季. Easy RL：强化学习教程[M/OL]. 北京: 人民邮电出版社, 2022. https://github.com/datawhalechina/easy-rl.

[131] 王树森，黎域君，张志华. 深度强化学习[M/OL]. 北京: 人民邮电出版社, 2022. https://github.com/wangshusen/DRL/blob/master/Notes_CN/DRL.pdf.

[132] WATKINS C J, DAYAN P. Q-learning[J]. Machine learning, 1992, 8: 279-292.

[133] SUTTON R S. Generalization in reinforcement learning: Successful examples using sparse coarse coding[J]. Advances in Neural Information Processing Systems, 1995, 8.

[134] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Playing atari with deep reinforcement

learning[J]. arXiv preprint arXiv:1312.5602, 2013.

[135] HASSELT H. Double q-learning[J]. Advances in Neural Information Processing Systems, 2010, 23.

[136] WILLIAMS R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning[J]. Machine Learning, 1992, 8: 229-256.

[137] BARTO A G, SUTTON R S, ANDERSON C W. Neuronlike adaptive elements that can solve difficult learning control problems[J]. IEEE Transactions on Systems, Man, and Cybernetics, 1983(5): 834-846.

[138] SILVER D, LEVER G, HEESS N, et al. Deterministic policy gradient algorithms[C]// International Conference on Machine Learning. PMLR, 2014: 387-395.

[139] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning[J]. arXiv preprint arXiv:1509.02971, 2015.

[140] FUJIMOTO S, HOOF H, MEGER D. Addressing function approximation error in actor-critic methods[C]//International Conference on Machine Learning. PMLR, 2018: 1587-1596.

[141] 伊笑莹, 芮一康, 冉斌, 等. 车路协同感知技术研究进展及展望[J]. 中国工程科学, 2024, 26(1): 178-189.

[142] GIRSHICK R, DONAHUE J, DARRELL T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]//IEEE Conference on Computer Vision and Pattern Recognition. 2014: 580-587.

[143] GIRSHICK R. Fast R-CNN[C]//IEEE International Conference on Computer Vision. 2015: 1440-1448.

[144] LIU W, ANGUELOV D, ERHAN D, et al. SSD: Single shot multibox detector[C]// European Conference on Computer Vision. Springer, 2016: 21-37.

[145] REDMON J, DIVVALA S, GIRSHICK R, et al. You only look once: Unified, real-time object detection[C]//IEEE Conference on Computer Vision and Pattern Recognition. 2016: 779-788.

[146] KONG T, SUN F, LIU H, et al. FoveaBox: Beyound anchor-based object detection[J].

IEEE Transactions on Image Processing, 2020, 29: 7389-7398.

[147] LAW H, DENG J. CornerNet: Detecting objects as paired keypoints[C]//European Conference on Computer Vision. 2018: 734-750.

[148] ZHOU X, ZHUO J, KRAHENBUHL P. Bottom-up object detection by grouping extreme and center points[C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 850-859.

[149] DUAN K, BAI S, XIE L, et al. CenterNet: Keypoint triplets for object detection[C]//IEEE/CVF International Conference on Computer Vision. 2019: 6569-6578.

[150] 全国汽车标准化技术委员会. 智能网联汽车感知训练数据集标准化需求研究报告[R/OL]. (2023-11-06)[2024-01-11]. http://www.catarc.org.cn/upload/202401/24/202401241353334269.pdf.

[151] ZHOU Y, TUZEL O. VoxelNet: End-to-End learning for point cloud based 3D object detection[C]//IEEE Conference on Computer Vision and Pattern Recognition. 2018: 4490-4499.

[152] YAN Y, MAO Y, LI B. SECOND: Sparsely embedded convolutional detection[J]. Sensors, 2018, 18(10): 3337.

[153] QI C R, SU H, MO K, et al. PointNet: Deep learning on point sets for 3D classification and segmentation[C]//IEEE Conference on Computer Vision and Pattern Recognition. 2017: 652-660.

[154] 霍威乐, 荆涛, 任爽. 面向自动驾驶的三维目标检测综述[J]. 计算机科学, 2023, 50(7): 107-118.

[155] MEYER G P, LADDHA A, KEE E, et al. LaserNet: An efficient probabilistic 3D object detector for autonomous driving[C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 12677-12686.

[156] YANG B, LUO W, URTASUN R. PIXOR: Real-time 3D object detection from point clouds[C]//IEEE Conference on Computer Vision and Pattern Recognition. 2018: 7652-7660.

[157] LANG A H, VORA S, CAESAR H, et al. PointPillars: Fast encoders for object de-

tection from point clouds[C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 12697-12705.

[158] SHI S, GUO C, JIANG L, et al. PV-RCNN: Point-Voxel feature set abstraction for 3D object detection[C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 10529-10538.

[159] ZHANG L, SUN Y, CHEN Z, et al. Communications-caching-computing resource allocation for bidirectional data computation in mobile edge networks[J]. IEEE Transactions on Communications, 2020, 69(3): 1496-1509.

[160] TOUT H, MOURAD A, KARA N, et al. Multi-persona mobility: Joint cost-effective and resource-aware mobile-edge computation offloading[J]. IEEE/ACM Transactions on Networking, 2021, 29(3): 1408-1421.

[161] TRAN T X, POMPILI D. Joint task offloading and resource allocation for multi-server mobile-edge computing networks[J]. IEEE Transactions on Vehicular Technology, 2018, 68(1): 856-868.

[162] ELGENDY I A, ZHANG W Z, ZENG Y, et al. Efficient and secure multi-user multi-task computation offloading for mobile-edge computing in mobile IoT networks[J]. IEEE Transactions on Network and Service Management, 2020, 17(4): 2410-2422.

[163] YAN J, BI S, DUAN L, et al. Pricing-driven service caching and task offloading in mobile edge computing[J]. IEEE Transactions on Wireless Communications, 2021, 20 (7): 4495-4512.

[164] KO S W, KIM S J, JUNG H, et al. Computation offloading and service caching for mobile edge computing under personalized service preference[J]. IEEE Transactions on Wireless Communications, 2022, 21(8): 6568-6583.

[165] BI S, HUANG L, ZHANG Y J A. Joint optimization of service caching placement and computation offloading in mobile edge computing systems[J]. IEEE Transactions on Wireless Communications, 2020, 19(7): 4947-4963.

[166] XU J, CHEN L, ZHOU P. Joint service caching and task offloading for mobile edge computing in dense networks[C]//IEEE Conference on Computer Communications. IEEE,

2018: 207-215.

[167] ZHANG G, ZHANG S, ZHANG W, et al. Joint service caching, computation offloading and resource allocation in mobile edge computing systems[J]. IEEE Transactions on Wireless Communications, 2021, 20(8): 5288-5300.

[168] SHEN Q, HU B J, XIA E. Dependency-aware task offloading and service caching in vehicular edge computing[J]. IEEE Transactions on Vehicular Technology, 2022, 71 (12): 13182-13197.

[169] WANG Y, SHENG M, WANG X, et al. Mobile-edge computing: Partial computation offloading using dynamic voltage scaling[J]. IEEE Transactions on Communications, 2016, 64(10): 4268-4282.

[170] BOYD S P, VANDENBERGHE L. Convex optimization[M]. Cambridge university press, 2004.

[171] FUDENBERG D, TIROLE J. Game theory[M]. MIT press, 1991.

[172] NING Z, ZHANG K, WANG X, et al. Joint computing and caching in 5G-envisioned internet of vehicles: A deep reinforcement learning-based traffic control system[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 22(8): 5201-5212.

[173] SU Z, HUI Y, XU Q, et al. An edge caching scheme to distribute content in vehicular networks[J]. IEEE Transactions on Vehicular Technology, 2018, 67(6): 5346-5356.

[174] 中国移动. 车路协同算力网络白皮书[R/OL]. (2023-10-28)[2024-01-11]. https://aimg8.dlssyht.cn/u/551001/ueditor/file/276/551001/1698108678560886.pdf.

[175] XIAO Z, SHU J, JIANG H, et al. Perception task offloading with collaborative computation for autonomous driving[J]. IEEE Journal on Selected Areas in Communications, 2022, 41(2): 457-473.

[176] LIU L, ZHAO M, YU M, et al. Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2022, 24(2): 2169-2182.

[177] KU Y J, BAIDYA S, DEY S. Uncertainty-aware task offloading for multi-vehicle perception fusion over vehicular edge computing[J]. IEEE Transactions on Vehicular Tech-

nology, 2023, 72(11): 14906-14923.

[178] LV P, XU W, NIE J, et al. Edge computing task offloading for environmental perception of autonomous vehicles in 6G networks[J]. IEEE Transactions on Network Science and Engineering, 2022, 10(3): 1228-1245.

[179] YANG J, CHEN Y, LIN Z, et al. Distributed computation offloading in autonomous driving vehicular networks: A stochastic geometry approach[J]. IEEE Transactions on Intelligent Vehicles, 2024, 9(1): 2701-2713.

[180] QIN P, FU Y, TANG G, et al. Learning based energy efficient task offloading for vehicular collaborative edge computing[J]. IEEE Transactions on Vehicular Technology, 2022, 71(8): 8398-8413.

[181] WANG X, NING Z, GUO S, et al. Imitation learning enabled task scheduling for online vehicular edge computing[J]. IEEE Transactions on Mobile Computing, 2020, 21(2): 598-611.

[182] KIM H, LEE S H, KIM S. Cooperative localization with constraint satisfaction problem in 5G vehicular networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 23(4): 3180-3189.

[183] SOATTI G, NICOLI M, GARCIA N, et al. Implicit cooperative positioning in vehicular networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2018, 19(12): 3964-3980.

[184] YANG P, DUAN D, CHEN C, et al. Multi-sensor multi-vehicle (MSMV) localization and mobility tracking for autonomous driving[J]. IEEE Transactions on Vehicular Technology, 2020, 69(12): 14355-14364.

[185] MENG W, CHU X, LU Z, et al. V2V communication assisted cooperative localization for connected vehicles[C]//IEEE Wireless Communications and Networking Conference. IEEE, 2021: 1-6.

[186] FANG S, LI H, YANG M. LiDAR SLAM based multivehicle cooperative localization using iterated split CIF[J]. IEEE Transactions on Intelligent Transportation Systems, 2022, 23(11): 21137-21147.

[187] RANGANATHAN A. The levenberg-marquardt algorithm[J]. Tutoral on LM algorithm, 2004, 11(1): 101-110.

[188] BRAMBILLA M, NICOLI M, SOATTI G, et al. Augmenting vehicle localization by cooperative sensing of the driving environment: Insight on data association in urban traffic scenarios[J]. IEEE Transactions on Intelligent Transportation Systems, 2019, 21 (4): 1646-1663.

[189] OKELLO K, MWANGI E, ABD EL-MALEK A H. Connectivity probability analysis for vanets with big vehicle shadowing[C]//International Symposium on Networks, Computers and Communications. IEEE, 2023: 1-6.

[190] GEIGER A, LENZ P, URTASUN R. Are we ready for autonomous driving? the KITTI vision benchmark suite[C]//IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012: 3354-3361.

[191] LIN T Y, GOYAL P, GIRSHICK R, et al. Focal loss for dense object detection[C]// IEEE International Conference on Computer Vision. 2017: 2980-2988.

攻读学位期间取得与学位论文相关的成果

发表和投稿与学位论文相关学术论文

[1] **Zheng Xue**, Chang Liu, Canliang Liao, Guojun Han, Zhengguo Sheng. Joint service caching and computation offloading scheme based on deep reinforcement learning in vehicular edge computing systems. IEEE Transactions on Vehicular Technology. 2023, 75(5):6709-6722. (SCI Impact Factor 6.8, WOS:000991849700088)

[2] **Zheng Xue**, Yang Liu, Guojun Han, Ferheen, Ayaz, Zhengguo Sheng, Yonghua Wang. Two-layer distributed content caching for infotainment applications in VANETs. IEEE Internet of Things Journal. 2022, 9(3):1696-1711. (SCI Impact Factor 10.6, WOS:000747462100010)

[3] **Zheng Xue**, Fuxi Wen, Chang Liu, Guojun Han. Task offloading and resource allocation for cooperative perception in vehicular edge computing systems. IEEE Transactions on Vehicular Technology. 2024. (SCI Impact Factor 6.8, 已投稿)

[4] Yang Liu, **Zheng Xue**, Guojun Han, Chang Liu, Canliang Liao. Fuzzy logic based agent selection for failure management in VANETs. International Symposium on Communications and Information Technologies (ISCIT). 2022:166-170.

[5] Chang Liu, **Zheng Xue**, Canliang Liao, Jiawen Kang, Guojun Han. DRL-based vehicular edge caching with dynamic content popularity for complex road intersections. IEEE Transactions on Vehicular Technology. 2024. (SCI Impact Factor 6.8, 已投稿)

[6] Canliang Liao, Chang Liu, **Zheng Xue**, Yang Liu, Guojun Han. Cooperative edge caching for vehicular networks in road intersections. International Symposium on Communications and Information Technologies (ISCIT). 2022:105-110.

[7] Yang Liu, Guojun Han, Yonghua Wang, **Zheng Xue**, Jing Chen, Chang Liu. Received signal strength-based wireless source localization With inaccurate anchor positions. IEEE Sensors Journal. 2022, 22(23):23539-23551. (SCI Impact Factor 4.3, WOS:000893571900123)

[8] **薛拯**, 刘畅, 韩国军. 面向车联网的城市交通目标级协同感知系统研究. 应用科技. 2024.

申请发明专利

[1] 韩国军, 廖灿亮, 翟雄飞, **薛拯**, 刘畅. 一种最佳采样信号识别方法、装置、电子设备及存储介质. 发明专利申请号: CN202211281538.5.

致谢

行文至此，我的博士求学生涯即将画上句号。我读过黄国平博士论文走红致谢中的“把书念下去，然后走出去，不枉活一世”的坚定信念，在当今内卷时代这种信念更加难能可贵。每个人都有自己的选择和理由，人生从来没有完美的答案，选择能承担的，承担所选择的，和时间做朋友，不断完善自己。三年时光短亦长，有过迷茫、焦虑、孤独，有简单的快乐也有现在的不舍。成长路上的点滴进步离不开老师们、同学们的帮助和支持，以及家人们无私的爱。谨以此文向所有关心和帮助我的人们表示最诚挚的感谢和最美好的祝福。

首先我要感谢我的导师韩国军教授，给予了我攻读博士学位的机会。六年前有幸加入您的课题组，一路见证团队和学院的飞速发展，您是我学术道路上的引路人，在科研上提供了优越的环境，您对我的信任和鼓励激发了我不断超越自己的动力。韩老师在学术上严谨的治学态度、渊博深厚的学识和前瞻性的洞察力，在科研课题上为我指明了方向。韩老师为人谦和，平易近人，在为人处事和人生规划方面提供了宝贵的建议。此外，感谢韩老师让我有机会去清华大学智能汽车设计与安全性研究中心访问交流，学习优秀经验。毕业之际，满怀感激，我将永远铭记您对的我的慷慨付出和关心。

同时，衷心感谢信息工程学院的刘畅老师、翟雄飞老师和刘洋老师。刘畅老师在课题推进、英文写作方面提供了非常大的帮助，对论文的研究点反复斟酌，带领车联网课题组稳步向前推进。翟老师在推进敏视公司项目时给予我专业的指导，帮助我快速入门 FPGA。刘洋老师在研究课题上提供了宝贵的建议。老师们丰富的科研经验、严谨负责的科研精神对我产生了深远的影响。

同时，由衷感谢清华大学车辆与运载学院的温福喜老师，在读博士期间有幸在智能汽车设计与安全性研究中心进行了为期一年的访问，温老师严谨求实、敏锐的学术洞察力深深影响着我。感谢一起在科建大厦科研的宋致应、张海梁、施笑寒、谢腾辉、赵成祥、马小涵、陈正先、王力、李雪珂、吴奕佳、刘家欣、李耀宇、徐彦超、杨磊、吴思宇、牟睿、古昕昱、杨胜利、陈炜，与你们在清华相遇是一种缘分，感谢你们的关怀、照顾和陪伴，愿你们在自己的道路上砥砺前行。

另外，感谢实验室的李英阁、海勤达、廖灿亮、孙祎、尹蔺欣、曾文坦、杨伟泽、刘婕颖、童健、尹蔺欣、宁康、耿晓超、叶龙建、张双旺、谢颖辉、叶震亮、罗铮、罗

岚、莫劲洪、邓彩云、李应钊、胡康、王勇亮、曾子锋、魏宏敏、许嘉成、潘雨晨、周宇、曾皓、陈泽豪、陈彦汛、潘俊吉、张玉榕。和你们一起探讨学术、一起吃饭、一起运动和玩耍丰富了我的科研生活，让实验室充满了欢声笑语。感谢韩老师信任让我统筹了课题组 2022 级的招生面试工作，希望你们科研顺利、生活和工作顺心。感谢室友胡海华，一路陪伴、鼓舞和帮助，希望你一切顺利。

特别感谢在背后默默支持和鼓励的父母和其他亲人，你们是我战胜困难的勇气和不断前进的动力，感恩之情无以言表，希望你们健康快乐，让我用余生来报答这份恩情。

最后，由衷感谢论文评审专家和答辩委员会专家对我的论文进行审阅和评价。您们的宝贵意见和建议帮助我完善了论文，并促使我更深入地思考和探索研究问题。谢谢您们的辛勤工作和专业指导！