

Resource Management for Vehicular See-through Service Provisioning

by

Ruoxu Wang

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Master of Applied Science

in

Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2024

© Ruoxu Wang 2024

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Vehicular see-through service is one of the driver assistance services expected to be provided by smart vehicles in the future. It is envisioned that this service will expand the vision of a smart-vehicle driver by timely alerting the driver to obstacles in the front blind spots. In this study, we investigate cooperative perception and resource allocation in the provision of the vehicular see-through service to ensure that drivers are able to obtain accurate and timely environment information about their front blind spots. To cope with the consumption of communication and computing resources associated with the transmission and processing of multi-view point clouds, we propose a cooperative sensor data collection and processing scheme. In this scheme, smart vehicles cooperatively collect point cloud data for each object and complete the point cloud processing at the on-board computing unit. For each object, we select a set of collectors to collect the point cloud data and an analyzer to process the data. To address the tradeoff between classification accuracy and resource efficiency, we investigate the impact of the location of cooperative vehicles on object classification accuracy and propose an indicator to assess the quality of the collector sets. We develop a collector set pre-selection algorithm that identifies all collector sets for each object, to satisfy the classification accuracy and have minimal data redundancy. To address the tradeoff between the service requirements and the overall resource consumption of the system, we treat each potential collector set and its analyzer of an object as a worker pair and select the worker pair for each object based on its resource consumption. Taking into account the resource competition among worker pairs, we develop a joint vehicle selection and resource allocation algorithm based on ant colony optimization to minimize the overall resource consumption, while satisfying the delay requirements of the

service. Simulation results demonstrate that our proposed service provisioning scheme outperforms benchmark schemes in terms of resource efficiency, task completion rate and request completion rate.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Professor Weihua Zhuang, for her professional guidance, continuous support, and invaluable encouragement throughout my research. Her rigorous academic attitude and insightful thoughts on research problems have profoundly cultivated my ability for in-depth thinking. The methodologies she introduced and the high standards she upheld have enhanced my research skills and will continue to benefit me throughout my professional and personal life.

I would like to extend my deepest gratitude to Professor Xuemin (Sherman) Shen for his invaluable guidance and great support. His insights into both research and life have laid a solid foundation for my work and have enriched my personal growth. I will always remember his teachings, which will continue to guide and influence my journey long into the future.

I would like to express my special thanks to Professor Wei Song for her great help. She generously devoted much of her time to discussing key issues with me, providing selfless help that enabled me to overcome numerous challenges.

I would also like to thank Prof. Zhou Wang and Prof. Patrick Mitran for serving on my thesis committee. I would like to thank Prof. Guang Gong, Prof. Jun Liu and Prof. Sagar Naik for serving on my background comprehensive exam committee. Their insightful comments and valuable questions have significantly improved the quality of my research.

Over the past four years, the experience and memories I have gained in the BBCR group have become my greatest treasures. I would like to extend my sincere thanks to Dr. Kaige Qu, Prof. Dongxiao Liu and Yannan Wei for their substantial support in both my

research and personal life. I am also grateful to all my colleagues at the BBCR group. Their expertise and advice have greatly contributed to my growth as a researcher and have enriched my understanding of the field.

Finally, I would like to express my profound gratitude to my parents, Sili Wang and Changmei Yang, for their unwavering love and support. Their belief in my abilities and continuous encouragement have been my pillars of strength throughout this journey.

Dedication

The thesis is dedicated to my beloved parents, Sili Wang and Changmei Yang.

Table of Contents

List of Figures	xi
List of Abbreviations	xiii
List of Symbols	xv
1 Introduction	1
1.1 Vehicular See-Through Service	2
1.2 Service Provision Schemes	5
1.3 Motivations and Objectives	8
1.4 Outline of the Thesis	14
2 Background and Literature Survey	15
2.1 Service Provisioning Schemes	15
2.1.1 Early Fusion Scheme	16

2.1.2	Mid Fusion Scheme	17
2.1.3	Late Fusion Scheme	18
2.2	Participant Selection for Service Provisioning	19
2.3	Summary	20
3	System Model	22
3.1	Service Provision Scenario	22
3.2	Object Classification Task Model	26
3.3	Point Cloud Quality Model	28
3.3.1	LiDAR Sensor Model	28
3.3.2	Point Cloud Quality Indicator	30
3.4	Communication Model	35
3.5	Computing Model	38
3.6	Summary	39
4	Problem Formulation and Solution	40
4.1	Problem Formulation	40
4.2	Problem Solution	44
4.2.1	Classification Accuracy Fulfillment	44
4.2.2	Joint Vehicle Selection and Resource Allocation	50
4.3	Summary	58

5	Performance Evaluation	60
5.1	Simulation Setup	60
5.2	Simulation Results	66
5.2.1	Performance under Moderate Traffic Conditions	66
5.2.2	Performance under Different Traffic Conditions	71
5.3	Summary	74
6	Conclusion and Future Work	75
6.1	Conclusion	75
6.2	Future Work	76
	References	78

List of Figures

1.1	Typical driving experience degradation caused by front blind spots.	3
3.1	Service region.	23
3.2	The front blind spots of a smart vehicle.	24
3.3	Mechanical multi-line LiDAR sensor structure.	30
3.4	An illustration of collectors' coverage areas.	32
3.5	An illustration of collector j 's coverage area.	33
4.1	Relationship between point cloud quality and classification accuracy.	46
5.1	Simulation environment.	61
5.2	The structure of the feature learning network [1].	62
5.3	The structure of the voxel feature encoding layer [1].	63
5.4	The structure of the Voxnet [2].	64
5.5	Average task completion rate for moderate traffic conditions.	68

5.6	Overall resource consumption versus number of tasks.	69
5.7	Communication and computing resource consumption versus number of tasks.	70
5.8	Average request completion rate versus number of traffic participants. . . .	72
5.9	Resource consumption per task versus number of traffic participants. . . .	73

List of Abbreviations

3D	3-Dimensional
ACO	Ant Colony Optimization
AP	Average Precision
BN	Batch Normalization
BS	Base Station
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DNN	Deep Neural Network
FLN	Feature Learning Network
FLOP	Floating Point Operation
HUD	head-up display
IoV	Internet-of-vehicle
LiDAR	Light Detection And Ranging
LoS	Line-of-sight
NMS	Non-maximal Suppression

OFDMA	Orthogonal Frequency-division Multiplexing Access
ReLU	Rectified Linear Unit
ROI	Region of Interest
SGD	Stochastic Gradient Descent
V2N	Vehicle-to-network
V2V	Vehicle-to-vehicle
VFE	Voxel Feature Encoding

List of Symbols

a	The azimuth resolution of a LiDAR
$A_{i,1}^4$	The left endpoint of the segment $A_{i,1}^4 B_{i,1}^4$
b	The elevation resolution of a LiDAR
B	The available bandwidth in the service region
$\bar{B}_{i,s}$	The bandwidth occupied by all of the task i 's collectors to transmit point cloud to analyzer k
$\bar{B}_{i,s}^j$	The bandwidth occupied by the collector j to transmit its point cloud to analyzer k
$B_{i,1}^4$	The right endpoint of the segment $A_{i,1}^4 B_{i,1}^4$
c	The by the feature learning network to process one point
c_0	The computation resource consumed in the classification step of a task
c_i	The computation resource consumed by task i
c_i^f	The computation resource consumed in the fusion step of task i
d_0	The reference distance
d_ν	The length of the ν -th edge of the rectangle
d_s	The radius of the sensing dead zone of a collector

$d_{i,j}^\nu$	The length of line segment $A_{i,1}^4 B_{i,1}^4$
$d_{j,k}$	The distance between collector j and analyzer k
e	The number of lines of a laser array
$E_{j,k}$	The spectrum efficiency of the link established between collector j and analyzer k
f_k	The available computation resources of the smart vehicle k
$\bar{f}_{i,s}$	The CPU frequency occupied by the analyzer k to process task i 's point cloud data
$g_{j,k}$	The channel gain of the communication link between vehicle j and vehicle k
h_i	The height of the object i
h_j	The height of the optical center of collector j 's LiDAR
I	The total number of unknown objects in the service region
\mathcal{I}	The set of all unknown objects in the service region
\mathcal{I}_n	The set of all unknown objects of smart vehicle n 's driver
k_i	The analyzer of task i
K	The channel gain per unit distance
l	The beam array index
l_i	The length of the object i
$L_{i,j}^\nu$	The set of vertical laser beam arrays emitted by collector j that hit the coverage area $q_{i,j}^\nu$
M	The maximum number of categories that can be recognized by the classification neural network

n_i	The total number of points received by the analyzer for task i
$n_{i,j}$	The number of points in the point cloud data provided by collector j
N	The total number of smart vehicles in the service region
N_i^c	The total number of valid collector sets of task i
$N_{i,j}^\nu$	The number of points collected by collector j in the ν -th face of object i
$N_{i,j}^{\nu,l}$	The number of points in the set $P_{i,j}^{\nu,l}$
\mathcal{N}	The set of smart vehicles in the service region
\mathcal{N}_i	The set of selected collectors for task i
\mathcal{N}_i^c	The set of all collector sets for object i that satisfy the accuracy requirement and have the minimal data redundancy
\mathcal{N}_i^l	The set of smart vehicles in task i 's l_i -th valid collector set
\mathcal{N}_i^t	The current set in Algorithm 1
\mathcal{N}^r	The remaining smart vehicle set in Algorithm 1
o_i	The bounding box of object i
O	The overall resource consumption for service provisioning
p	The point index of the set $P_{i,j}^{\nu,l}$
p_j	The transmission power of collector j
$p_{i,j}(t)$	The probability that the worker pair s is selected for task i
P_i	The M-dimensional prediction vector of task i
P_l	The intersection of the vertical laser beam array l and the face ν in the top view
$P_{i,m}$	The predicted probability that the object i belongs to category m

$P_{i,j}^{\nu,l}$	The set of points emitted by the vertical laser beam array l that hit the object i
\tilde{q}_0	The point cloud quality threshold value
$q_i(\mathcal{N}_i)$	The overall quality of the fused point cloud for object i
$\tilde{q}_i(\mathcal{N}_i)$	The normalized quality of the fused point cloud for object i
\tilde{q}_i^t	The current quality in Algorithm 1
q_i^ν	The coverage area of the fused point cloud on the ν -th face of the object i
$q_i^\nu(\mathcal{N}_i)$	The point cloud quality for the ν -th face of object i
$q_{i,j}^\nu$	The coverage area of collector j on the ν -th face of the object i
$r_{j,k}^i$	The transmission rate of the link between collector j and analyzer k for task i
s	The worker pair index
s_i	The worker pair selected for task i
$s_{i,j}$	The data size of the point cloud data provided by collector j for task i
\mathbf{S}	The worker pair selection decision vector
$\mathbf{S}_g(t)$	The worker pair selection decisions made by ant g in t -th iteration
t_c	The average reaction time of human drivers
t_i	The moment at which task i is completed
t_i^P	The task processing time of the task i
t_i^T	The point cloud transmission time of the task i
$t_{i,j}^{T,k}$	The time it takes for collector j to send the sensor data of object i to analyzer k

\mathbf{U}	The collector selection decision matrix
v_n	The speed of smart vehicle i
\mathbf{V}	The analyzer selection decision matrix
w	The weighting factor
w_i	The width of the object i
x_a	The x-axis coordinate of the left endpoint of the segment $A_{i,1}^4 B_{i,1}^4$
x_b	The x-axis coordinate of the right endpoint of the segment $A_{i,1}^4 B_{i,1}^4$
x_i	The x-axis coordinate of the geometric center of the object i
x_l	The x-axis coordinate of the point P_l
y_a	The y-axis coordinate of the left endpoint of the segment $A_{i,1}^4 B_{i,1}^4$
y_b	The y-axis coordinate of the right endpoint of the segment $A_{i,1}^4 B_{i,1}^4$
y_i	The y-axis coordinate of the geometric center of the object i
y_l	The y-axis coordinate of the point P_l
y_{AB}	The line connecting the two points $A_{i,1}^4(x_a, y_a)$ and $B_{i,1}^4(x_b, y_b)$
z_i	The z-axis coordinate of the geometric center of the object i
$\alpha_{k,i}$	The ratio of the computation resources allocated to the task i to the total available resources f_k
$\boldsymbol{\alpha}$	The computation resources allocation decision matrix
$\beta_{j,i}$	The available bandwidth in the service region
$\boldsymbol{\beta}$	The communication resources allocation decision matrix
ϵ	The data size of one point in the point cloud data
η_i	The normalized information entropy of the M -dimensional output vector
$\eta_{i,s}$	The selection priority of a worker pair s

ℓ_n	The length of smart vehicle i 's ROI
ν	The face index of an bounding box
ϕ	The upper bound of the vertical sensing range
ϕ_i	The rotation of object i around x-axis
ψ_i	The rotation of object i around z-axis
γ	The large-scale fading factor
ρ_0	The point cloud density threshold
$\rho'_{i,j}$	The resolution of the point cloud collected by collector j on the ν -th face of object i
σ^2	The receiving noise power at the smart vehicle's receiver
τ	The service period
θ	The lower bound of the vertical sensing range
θ_i	The rotation of object i around y-axis
μ_0	The accuracy requirement of the classification task
μ_i	The confidence level of the classification result of task i
$\xi_{i,s}(t)$	The pheromone value of the worker pair s of task i

Chapter 1

Introduction

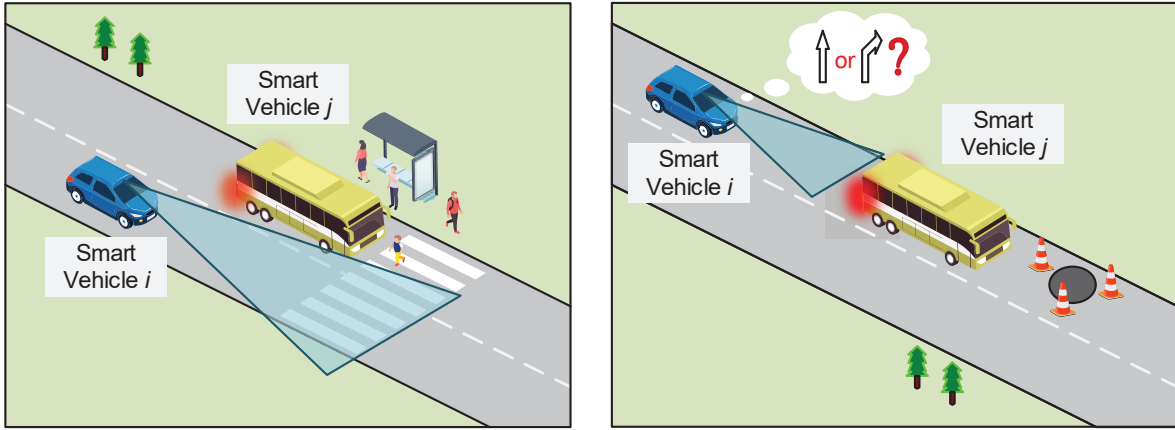
The growing demand for improved driving safety and comfort has catalyzed the continuous development of driver assistance technologies. Nowadays, a large number of driver assistance technologies have been invented and integrated into newly manufactured vehicles. Vehicular see-through service is envisioned as a key component of future driver assistance systems [3]. It is anticipated to expand the vision of a vehicle driver by offering real-time environment information about obstructed frontal areas [4]. In contrast to traditional driver assistance technologies such as lane keeping assistance and adaptive cruise control, the vehicular see-through service relies on cooperation between vehicles. It requires rapid inter-vehicle sensor data sharing and data processing to detect and classify objects in the environment [5]. Due to limited sensing, communication and computing resources, it is challenging for vehicles to provide accurate environment information to the driver in a very short time. Therefore, a key issue in the service provision is how to efficiently manage the available resources of the vehicles so that the stringent requirements for service delay

and accuracy in environment information can be met.

1.1 Vehicular See-Through Service

Vehicles featuring driver assistance technologies are often called smart or intelligent vehicles [6]. These vehicles are equipped with advanced sensing, communication, and computing systems. Sensors installed in the front, back, and sides of a smart vehicle can provide a 360-degree view of the surroundings, enabling the smart vehicle to analyze the environment and provide real-time environment information to the driver. This information helps the driver better understand the surroundings, facilitating safer and more comfortable driving decisions [7]. In addition, smart vehicles can share state and decision information via Internet-of-vehicle (IoV). This interconnectivity allows them to work together to reduce congestion and collision risks, thereby leading to a more safer and efficient driving experience [8].

Despite their advanced sensing capabilities, smart vehicles may encounter limitations in obtaining information of all relevant areas for safe and comfortable driving. This is because the sensing range of a single smart vehicle can be restricted by obstructions from other vehicles or large objects on the road [9]. Two instances illustrating such obstructions are depicted in Figure 1.1. In Figure 1.1(a), smart vehicle i 's sensors cannot detect a pedestrian crossing the road in front of vehicle j , which is engaged in passenger drop-off activities. This obstruction presents a risk of potential collision as vehicle i moves alongside vehicle j . Figure 1.1(b) presents a situation where smart vehicle i is approaching a halted vehicle, j . Since its view is obstructed by vehicle j , it is difficult for the driver to make a timely



(a) Pedestrian going across the road.

(b) Leading vehicle stopping on the road.

Figure 1.1: Typical driving experience degradation caused by front blind spots.

decision about changing lanes or waiting for vehicle j to move. The above view-obstructed areas in front of the smart vehicle are known as front blind spots [9]. These blind spots significantly limit the vehicle's ability to provide complete environment information to the driver, thereby impairing road safety or degrading the quality of driving experience. This issue becomes particularly critical in urban areas, where the complex road environment frequently leads to view obstruction [10].

A promising driver assistance technology to address front blind spots in urban environments is the vehicular see-through service [11]. The idea is to gather environment information from nearby smart vehicles with an unobstructed view of these blind spots in real-time and project this information onto a head-up display (HUD) [12]. As the driver observes the HUD, a virtual road representing the current environment is rendered on the screen, with 3-dimensional (3D) models of road elements such as vehicles, pedestrians, and roadblocks accurately overlaid at their real-world locations on the virtual road. This

visualization allows the driver to effectively “see-through” obstructing objects, thereby significantly enhancing situational awareness and promoting more informed driving decisions [4].

To provide the vehicular see-through service to a smart-vehicle driver, the smart vehicle needs to acquire environment information about the front blind spots from one or more nearby smart vehicles to achieve the full coverage of all the front blind spots [13]. The environment information includes the existence, location, and category of objects within the front blind spots. To keep pace with the dynamic nature of the environment, the smart vehicle needs to acquire this information frequently. The frequency at which the environment information is acquired is generally the sampling frequency of the on-board sensors of nearby vehicles.

Advanced smart vehicles use light detection and ranging (LiDAR) to scan their surroundings and store the sensor data as a point cloud. Specifically, a LiDAR sensor includes multiple laser transmitters and receivers. It determines the distance to surrounding objects by emitting laser beams to the environment and measuring the time it takes for the laser beams to be reflected back. Each reflected laser beam provides information on the distance to objects detected in a specific direction. By conducting measurements in different directions, the LiDAR sensor records the location of surrounding objects and stores the information into a 3D point cloud.

In each information acquisition period, the nearby smart vehicles collect sensor data from their LiDAR sensors. Then, each of them independently performs an environment perception task to obtain environment information about their surroundings from the point cloud. Once the environment information is obtained, the nearby smart vehicles send

the information back to the smart vehicle that requests the service. The environment information will be integrated at the smart vehicle that requests the service and finally be projected onto the screen.

The vehicular see-through service has strict requirements for the completeness, accuracy and timeliness of environment information about the front blind spots [11]. However, these requirements present considerable challenges for smart vehicles with limited resources. On one hand, the LiDAR sensors have limited sensing capabilities [14, 15]. When the front blind spot is obstructed or the point cloud resolution is low, it is difficult for an environment perception algorithm to obtain accurate environment information from the point cloud data. On the other hand, advanced environment perception algorithms adopt deep neural networks (DNNs) to detect and classify surrounding objects [16, 1]. These operations are computation-intensive, which can potentially hinder the ability of resource-constrained smart vehicles to fulfill tasks within the required time period [17]. Therefore, a key issue in service provisioning is how to efficiently manage the available resources of smart vehicles so that the service requirements can be met.

1.2 Service Provision Schemes

To meet the above requirements, previous studies select one of the surrounding vehicles to provide the environment information for each smart vehicle with a service request [18, 19]. At the beginning of each information acquisition cycle, the smart vehicle with an obstructed view generates a service request and selects one of the smart vehicles in its surroundings to acquire the environment information in the front blind spots. The view-obstructed smart

vehicle is referred to as the requester, and the smart vehicle that provides environment information is referred to as the provider. After receiving the service request, the provider obtains point cloud data of its surroundings from the onboard LiDAR sensor and then processes an environment perception task to obtain information about the existence and category of all the objects in its surroundings. The provider then sends this information back to the requester.

However, the LiDAR sensor has the characteristic of line-of-sight (LoS) sensing. If there are large objects located around the sensor, the sensor's view can be obstructed, which can lead to the sensor data not fully covering the front blind spot of the requester [20]. In addition, due to the geometric perspective property of the LiDAR sensor, the resolution of the point cloud data in the front blind spot gradually decreases as the distance between the provider and the front blind spot increases [21]. A lower point cloud resolution will result in a lower level of detection and classification accuracy of the environment perception algorithm [22]. Therefore, in urban roads where smart vehicles coexist with traditional vehicles and pedestrians, it is difficult to find a suitable provider for each requester.

In recent years, the emergence of cooperative perception has provided a new solution to the service provisioning problem. Cooperative perception allows each requester to select multiple providers [23]. The providers located in different locations have different viewing angles and distances to the front blind spots. By fusing environment information from multiple providers, the information complementarity in the front blind spots can be realized. Thus, the problem of limited coverage by a single provider can be addressed. For each object located within a front blind spot, different providers can provide information with different viewing angles. By fusing this information, the environment perception al-

gorithm can have a more comprehensive understanding of the object, which can improve the detection and classification accuracy of low-resolution objects [24].

Depending on the data fusion methods, three cooperative methods have been proposed, i.e., early fusion, mid fusion and late fusion [25]. Early fusion forms a new point cloud by stitching the point cloud data from all the providers and then utilizes an environment perception algorithm to detect and classify objects in the stitched point cloud. Since all the point cloud data collected by the providers is utilized, this method has the highest object detection and classification accuracy [23]. Mid fusion requires each provider to perform feature extraction independently and then fuse these features for environment perception. Since some information is lost in the feature extraction, the detection and classification accuracy of this method is lower than that of early fusion [24]. Late fusion obtains environment information by combining the independent environment perception results from each provider. Since the complementary information of the original point cloud data is not fully utilized, the detection and classification accuracy of late fusion is lower than that of early and mid fusion [26].

Recently, there have been several studies applying early fusion to cooperative perception systems and have achieved high environment perception accuracy [23, 27]. However, the transmission and processing of point cloud data requires a large number of resources, which hinders the practical deployment of the early fusion-based cooperative perception. First, the size of the point cloud data collected is very large. For example, a single frame of point cloud generated by a 32-line LiDAR contains the location and reflectivity information of more than 100,000 reflection points [28]. It will take a long time for the smart vehicle to transmit the point cloud data to the data processing node. Second, the stitching of

the point cloud expands the processing area of the environment perception algorithm [23]. Since the point cloud data contains many non-blind regions, processing the stitched point cloud data can lead to unnecessary consumption of computing resources. Third, the sensing regions of multiple providers may overlap each other. The sensing data of the overlapping regions will be processed repeatedly, leading to considerable computational redundancy [29]. Therefore, recent studies are focusing on reducing the size of the point cloud to be transmitted and processed without compromising the environment perception accuracy [30, 31].

1.3 Motivations and Objectives

Recently, some researchers have proposed a cooperative scheme based on a two-stage environment perception algorithm to improve resource efficiency [18, 27]. In this scheme, cooperative perception is divided into two stages: object detection and object classification. In the object detection stage, all smart vehicles independently employ non-compute intensive semantic segmentation or shape-fitting algorithms to determine the location of surrounding objects and predict their bounding boxes. In the object classification stage, a controller deployed at the base station (BS) selects a group of smart vehicles to provide point cloud data for each detected object. The selected smart vehicles are required to send the point cloud data of the objects to the controller. Subsequently, the controller fuses the received point cloud data and utilizes a compute-intensive classification neural network to classify the detected objects from the fused point cloud data.

In this scheme, instead of transmitting the entire point cloud data to the controller, the

smart vehicle transmits only the point cloud data of the object. Therefore, the two-stage cooperative scheme consumes fewer communication resources than the traditional early fusion method. In addition, since the fused point cloud data does not contain irrelevant data such as non-blind spot regions and the background of the object, the consumption of computing resources of the two-stage cooperative scheme is much less than that of the traditional early fusion approach. Therefore, applying this method to the vehicular see-through service can potentially improve resource efficiency. However, the following two key issues must be addressed for the practical application of the two-stage cooperative scheme to the vehicular see-through service provisioning:

- In the object classification stage, each detected object is allowed to select one or more smart vehicles to provide point cloud data. However, there may be many smart vehicles on the road. How to select a set of smart vehicles to provide point cloud data for each object is a new problem. The difficulty of the problem is that the point cloud data provided by the selected smart vehicles should meet the required classification accuracy after data fusion and have high resource efficiency. However, it remains unclear how classification accuracy is affected by point cloud data collected from different vehicles. To obtain the classification accuracy of the fused point cloud data, a classification neural network should be used to process the point cloud data. The classification accuracy is then calculated based on the output of the classification neural network. However, traversing all possible combinations to find the most resource-efficient one is not feasible in a practical deployment of the service. For example, a 300-meter-long section of a two-way four-lane urban roadway contains an average of 18 vehicles under moderate traffic conditions [32]. Assuming that

half of these vehicles are smart vehicles, there are up to 512 possible smart-vehicle combinations, i.e., 2^9 , for any given object. Traversing these combinations with a classification neural network can lead to significant consumption of communication and computing resources.

Recent studies have shown that the viewing angle of different sensors has a significant impact on the accuracy of cooperative perception algorithms in classifying an object [33, 34]. Specifically, smart vehicles with similar locations have similar viewing angles. Fusing the sensor data of these vehicles is not helpful for neural networks to better understand the shape of the object. On the contrary, some smart vehicles have large differences in viewing angles, and they can collect sensor data from different parts of the object. Fusing the sensor data of these vehicles can improve classification accuracy. Based on this observation, recent studies have attempted to use the object coverage ratio to assess the quality of the fused sensor data [35, 36]. By iteratively selecting the sensor that maximizes the coverage ratio to be added to the system, these methods reduce the number of sensors participating in the cooperation while satisfying the data quality requirements.

It is challenging to comprehensively assess the impact of the sensor data selection on the classification accuracy. This is because there are many parameters that may affect the classification accuracy of a classification neural network. Existing studies have shown that classification accuracy is related to both the coverage area of the object and the resolution of the sensor data [22, 33]. As the LiDAR moves farther away from the object, the amount of point cloud data it collects from the object gradually decreases, making it more difficult for the classification neural network to

understand the shape of the object. In addition, data redundancy can occur when LiDAR sensors collect point cloud data from similar locations and viewing angles. The redundant point cloud data is not helpful for the classification neural network to understand the shape of the object. Previous studies on object coverage ratio have established an assessment indicator for the sensor data selection [35, 36].

In this study, we want to further investigate the relationship between sensor data selection and classification accuracy, to establish an accuracy estimation model. Based on the estimation model, the controller can quickly identify the combinations of smart vehicles that meet the accuracy requirement for each object.

- For each detected object, the acquisition of its category information requires processing a compute-intensive classification neural network, which can be regarded as a computing task. In previous studies, the classification neural network is placed at a controller with sufficient computing resources deployed at the BS [27, 31]. Considering that smart vehicles are equipped with advanced communication and computing modules, some studies have suggested that smart vehicles can utilize their free computing resources to process the classification neural network [37, 38]. This local processing scheme helps prevent service failures due to inadequate infrastructure and also improves the resource efficiency of smart vehicles. In this study, we consider that classification neural networks are deployed at smart vehicles. Therefore, smart vehicles are categorized into three roles: the requester that requests services, the collector that provides sensor data, and the analyzer that processes the classification neural network. For each detected object, a set of collectors and an analyzer are required to be selected for sensor data collection and processing. In addition, communica-

tion and computing resources are required to be allocated for data transmission and processing.

It is challenging to simultaneously achieve the stringent accuracy and delay requirements and maintain low resource consumption for these services because there exists a tradeoff between service requirements and resource consumption. Specifically, selecting more collectors for an object can improve classification accuracy. However, more collectors means that more data needs to be transmitted and processed, leading to more resource consumption. In contrast, selecting collectors with smaller data sizes can reduce the computing resource consumed by data processing. When these collectors are far away from other smart vehicles with sufficient computing resources, the transmission and processing of point cloud data may require more resources to ensure that the delay requirements are satisfied.

It can be seen from the above discussion that treating collector set selection and analyzer selection as two separate problems cannot achieve high resource efficiency. In this study, we want to propose a joint collector set and analyzer selection scheme and allocate appropriate communication and computing resources for data transmission and processing. The proposed scheme should consider the overall resources consumed by the selected vehicles for each task so as to avoid additional resource consumption due to the over-selection of collectors and the mismatch of collector sets and analyzers.

In summary, we focus on the resource management problem in the classification stage of the two-stage cooperative scheme. In this scheme, a controller deployed at the BS generates a classification task for each detected object. We design a solution to select a set of collectors and an analyzer for each task and allocate appropriate resources to the data

transmission and processing such that the accuracy and delay requirements of the service are satisfied and the overall resource consumption is minimized. The contributions of this research are summarized as follows:

- A cooperative sensor data collection and processing scheme is proposed for the classification stage to improve the resource efficiency of the vehicular see-through service. By selecting smart vehicles with complementary viewing angles and high point cloud resolutions to provide point cloud data, the proposed scheme reduces the additional resource consumption caused by redundant sensor data. The proposed scheme also allows smart vehicles to utilize their own computing resources to fuse and process point cloud data, which improves the resource efficiency of smart vehicles.
- A classification accuracy model is proposed for characterizing the relationship between collector selection and object classification accuracy. Specifically, we consider three key parameters that affect the classification accuracy, i.e., the coverage ratio of each face of the object, the resolution of the point cloud data, and the redundancy of the view angles. Based on the parameters, we propose a new indicator for assessing the quality of the fused point cloud and find the relationship between the indicator and the object classification accuracy. By comprehensively considering the key parameters affecting the classification accuracy, the proposed accuracy model can estimate the classification accuracy of the fused point cloud data.
- A joint collector set and analyzer selection scheme with resource allocation is proposed to 1) select a set of collectors for each task to collect the point cloud data; 2) select an analyzer for each task to process the data; 3) allocate communication resources for the data transmission between the collectors and the analyzers; 4) allocate

computing resources for each task. To ensure that each task achieves the required classification accuracy, we design a backtracking algorithm based on the proposed accuracy model to find all the potential collector sets that satisfy the accuracy requirement with minimal data redundancy. To address the problem of the inherent difficulty in simultaneously meeting stringent service requirements and low resource consumption demands, we treat each potential collector set-analyzer pair of a task as a single unit and assign a selection priority based on its resource consumption per unit time. Then, the vehicle selection and resource allocation problem is transformed into a pair selection and resource allocation problem, and we propose an ant colony optimization (ACO) algorithm-based solution to minimize the overall resource consumption of the system.

1.4 Outline of the Thesis

The rest of this thesis is organized as follows. A literature review is presented in Chapter 2 and the system model is described in Chapter 3. A vehicle selection and resource allocation problem is formulated in Chapter 4, in which a classification accuracy model and a joint collector set and analyzer selection scheme with resource allocation are also presented. Simulation results are presented in Chapter 5 to evaluate the performance of proposed solutions, and conclusions are drawn in Chapter 6.

Chapter 2

Background and Literature Survey

2.1 Service Provisioning Schemes

There are two types of schemes that have been proposed for the service provisioning of vehicular see-through services, i.e., the single-vehicle scheme and the cooperative scheme. In the single-vehicle scheme, each smart vehicle that generates a service request selects a neighboring vehicle to provide environment information [18, 19]. In the cooperative scheme, more than one neighboring vehicle can be selected by the smart vehicle to provide environment information. As introduced in the previous chapter, the cooperative scheme can effectively address the limitations of single-sensor perception by fusing point cloud data from multiple LiDAR sensors. In this section, we will focus on the cooperative scheme.

Compared to the single-vehicle schemes, the cooperative scheme requires the transmission of more point clouds, which results in the consumption of more communication resources. Therefore, a key issue in the cooperative scheme is to design efficient coopera-

tive perception methods to strike a balance between perception performance and resource consumption. Recent studies have investigated this problem from the perspective of point cloud data fusion. From the perspective of data fusion methods, the existing studies can be categorized into the following three types: early fusion scheme, mid fusion scheme, and late fusion scheme [25].

2.1.1 Early Fusion Scheme

Early fusion, also known as raw data fusion, forms a new point cloud by stitching together two or more point cloud data. Since point clouds record the location information of light-impermeable entities in space, point cloud data can be directly stitched together. Existing studies generally assume that the LiDAR sensors involved in cooperative perception are equipped with highly accurate localization systems. Therefore, by mapping the point clouds to a global coordinate system, these sensors can quickly obtain a point cloud stitching result. Cooper is one of the early fusion methods with high perception accuracy [23]. It maps all point clouds collected by LiDAR sensors in the system to a global coordinate system to achieve point cloud stitching. The method then uses VoxelNet [1] to detect and recognize objects in the stitched point cloud. Since the method makes full use of the collected point cloud data, the perception of this method is very high. To reduce the communication resources consumed by raw data transmission, three early fusion frameworks, namely Autocast, AVR and EMP, are proposed. Specifically, Autocast treats the object as the smallest unit [27]. For each object, Autocast requires all sensors to provide data. AVR is similar to Autocast. It adds a motion detection module before the point cloud transmission to control the perception performance of objects with different motion states

[30]. By detecting the motion states of objects in the environment, the AVR selectively transmits the point clouds of highly dynamic objects. In this way, the consumption of communication resources can be reduced and the perception accuracy of dynamic objects can be guaranteed. Unlike AVR, EMP divides the sensing area into non-overlapping sub-areas based on sensor locations [31]. Each sensor collects the point cloud of a subarea and shares it with the surrounding sensors. This greatly reduces the size of the point cloud data to be transmitted.

2.1.2 Mid Fusion Scheme

To further reduce the consumption of communication resources for point cloud transmission, recent studies have discussed mid fusion schemes. Mid fusion, also known as feature fusion, obtains comprehensive information about environmental features by fusing the feature maps of point clouds from different sensors. In this method, sensors are required to perform both data preprocessing and feature extraction steps individually, and then share the feature maps of the point cloud with others. To improve the perception performance, three mid fusion methods have been proposed. The first method is to maxout [24] or sum [39] the overlapping elements one by one after mapping the feature maps to the global coordinate system. To further improve the perception performance of mid fusion, some studies have modeled the influence of sensor characteristics on perception accuracy and designed a weight-based feature fusion method. DiscoNet is one of the representative studies [40]. This method considers the effect of sensor positions on perception accuracy. It models the complementary relationship between sensors in the cooperative system as a cooperative graph, where each sensor is a node and the weights of the edges reflect the strength of

complementarity between these sensors. To obtain the weights of each edge, this method designs a teacher-student training framework and trains a knowledge distillation network by using the knowledge from early fusion. When fusing sensor data, this method replaces the maxout operation in [24] with a proportional summation operation. In contrast, V2X-VIT considers the effect of sensor deployment locations such as vehicle roofs and light poles on the sensing accuracy [33]. Similar to DiscoNet, this method models the complementary relationships between the same and different sensors as a collaborative graph. By reducing the size of the graph, this method reduces the overall complexity of the system. In addition, there are similar studies such as V2VNet [34] and OPV2V [26]. The third method takes into account the multichannel nature of the feature graph. This method considers that each channel of a feature map contains a type of information. To make full use of the information of different channels, Slim-FCP designs an attention mechanism to select the most valuable channels from the multiple channels of the feature map and share them with other sensors [41]. This method can further reduce the consumption of communication resources and improve the perception performance.

2.1.3 Late Fusion Scheme

Although mid fusion can greatly reduce the consumption of communication resources, the data size of the feature maps is not negligible. In highly dynamic environments such as autonomous driving, this method may not be applicable due to the more stringent delay requirements for environment sensing. To deal with this problem, late fusion schemes have been proposed. In this method, sensors first independently detect and recognize objects in their environment and share the objects' bounding boxes, categories, and estimated proba-

bilities of belonging to each category. After receiving information from other sensors, each sensor fuses the bounding box of each object by using a non-maximal suppression (NMS) method and estimates the category to which the object belongs by using Bayesian estimation [42]. Since the amount of data to be transmitted is relatively small, the communication resource consumption and transmission delay of this method is negligible.

2.2 Participant Selection for Service Provisioning

Another issue in the provision of vehicular see-through services is the selection of cooperative participants. Since LiDAR sensors are based on LoS sensing, their sensing ability is often limited by the obstructions caused by surrounding objects and the low resolution caused by the large distance between the LiDAR sensor and the region of interest (ROI). In order to fully cover the ROI and achieve the required perception accuracy requirements, existing studies have investigated the issue of participant selection by proposing several participant-assessing indicators. The recent studies can be divided into two categories. One category uses the coverage of the interested region as the assessment indicator. These studies considered the influence of locations and visible regions of the sensors and calculated the coverage region of each sensor [43, 44]. By greedily selecting participants from the service region, these studies maximize the coverage ratio of the interested region while minimizing the number of selected sensors [45] or minimizing the resources consumed by these sensors [18].

Another category focuses on view quality related to object detection and classification accuracy and proposes several assessment indicators. In [19] and [40], the average precision

(AP) of the environment perception result is used to evaluate the selected sensor data providers. Since the AP value cannot be obtained before the environmental perception task has been processed, these studies assume that the environment does not change in a short period of time. To obtain the relationship between the AP value and the selected sensor data provider, these studies collect sensor data from all sensors in the system and train a neural network to profile the relationship [33]. To reduce the impact of changes in the environment, some general assessment indicators have been proposed in recent studies. For example, the coverage ratio of object aspects proposed in [35] and [46] is a well-known one. This indicator represents how much of the full 360-degree view around the object is covered by the selected sensor data providers. Real-world experiments conducted in [36] have proven that the higher the coverage ratio of object aspects, the better the result of environment perception. Recently, a new vehicle assessment method has been proposed in [38]. This method models an object as a cubic bounding box and divides it into several subregions. By counting the number of data points within each subregion, the method designs a new assessment indicator. Then, it trained a neural network to obtain the relationship between this indicator and the actual classification accuracy. By developing a fine-grained model of the object, this method can achieve good accuracy estimation performance.

2.3 Summary

In this chapter, we first introduce two schemes for the provisioning of vehicular see-through services: the single-vehicle scheme and the cooperative scheme. Then, we introduce three fusion schemes proposed by existing work that address the tradeoff between perception per-

formance and resource consumption in service provisioning. For the participant selection problem in service provisioning, we summarize the main idea of existing work and introduce two types of participant assessment indicators. The above work helps us to better understand the resource management problem in the provisioning of vehicular see-through services.

Chapter 3

System Model

3.1 Service Provision Scenario

Consider a multi-lane urban road covered by multiple BSs. As shown in Figure 3.1, the road segment covered by each BS is a service region. A controller is deployed on the BS to manage the provisioning of vehicular see-through services. In this study, we focus on resource management within a service region.

The service region under consideration includes N smart vehicles and a variety of objects, such as traditional vehicles, pedestrians, and obstacles such as roadside litter and roadblocks. Let $\mathcal{N} = \{1, 2, \dots, N\}$ denote the set of smart vehicles in the service region. All smart vehicles in the service region are equipped with LiDAR sensors, computing platforms, and communication modules. They can exchange information with each other to determine each other's location. Let M denote the number of object categories that can be recognized by smart vehicles. All objects within the service region can be classified

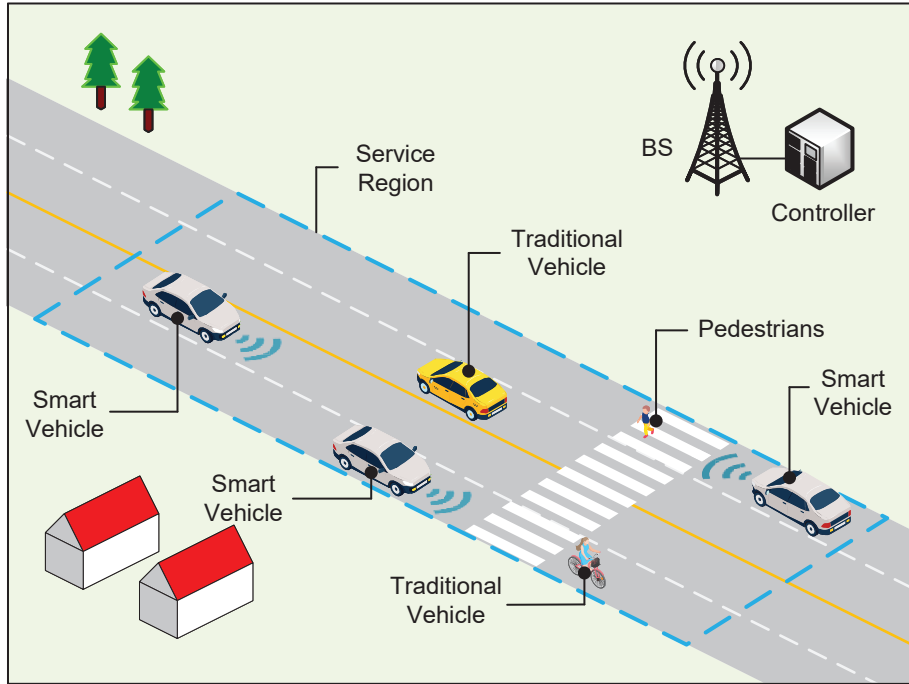


Figure 3.1: Service region.

into these categories. Since the objects are not equipped with the above modules, they cannot report their location and category information to smart vehicles. If these objects are located outside the view of the smart-vehicle driver, the driver cannot identify their existence.

To make safe and comfortable driving decisions, a smart-vehicle driver needs to be aware of the environment information about a road segment in its front. The road segment is referred to as the vehicle's ROI, the length of which is defined as the product of the vehicle's speed and the average reaction time of human drivers [18, 29]. Let v_n denote the speed of smart vehicle n and t_c denote the average reaction time. The length of smart vehicle n 's ROI, denoted by ℓ_n , is given by $\ell_n = v_n t_c$. We define the front blind spots as the areas

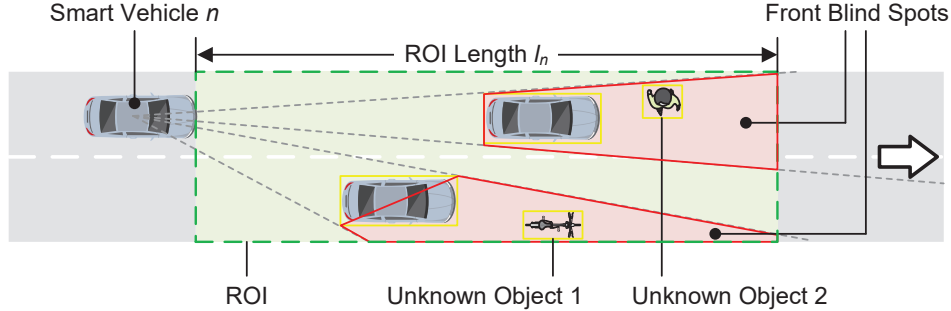


Figure 3.2: The front blind spots of a smart vehicle.

within the smart vehicle's ROI where the driver's view is obstructed. As shown in Figure 3.2, the road segments enclosed by the green dashed line and red solid line are the ROI and front blind spots of smart vehicle n , respectively. Since the bicycle and the pedestrian are located in the front blind spot, they are regarded as unknown objects by the smart vehicle. Let \mathcal{I}_n denote the set of unknown objects of smart vehicle n . The set of all unknown objects in the service region, denoted by \mathcal{I} , is given by $\mathcal{I} = \cup_{n \in \mathcal{N}} \mathcal{I}_n$.

The purpose of the vehicular see-through service is to provide smart-vehicle drivers with real-time environment information about the front blind spots. The environment information includes the existence and categories of all objects located within the front blind spots. To ensure that the smart vehicle driver obtains real-time environment information, we consider a periodic service provision scheme where the smart vehicle periodically provides services to the driver. The service period is short which allows vehicles to continuously update environment information to adapt to rapid environment changes. Let τ be the length of each period. We assume that all smart vehicles have the same service period, and the service starts simultaneously.

In each service period, the two-stage cooperative scheme described in Section 1.3 is

adopted to obtain the environment information about the front blind spots of smart vehicles. The two stages of this scheme are the object detection stage and the object classification stage.

In the object detection stage, all smart vehicles within the service region independently perform an object detection task. The purpose of the object detection task is to identify the existence, locations, and bounding boxes of objects in the environments around each smart vehicle. Then, each smart vehicle sends the detection results to the controller. Based on the received information, the controller determines the front blind spots of each smart vehicle and localizes the objects within these front blind spots. To detect objects in the environment, each smart vehicle scans its surroundings by using a LiDAR sensor. The in-vehicle computation platform then applies a classical Euclidean clustering algorithm to detect the existence and location of surrounding objects [47]. Finally, a shape fitting algorithm, such as minimum-bounding rectangle fitting or L-shape fitting, is used to determine the bounding boxes of the detected objects [48].

In the object classification stage, the controller creates an object classification task for each object located in the front blind spots of the smart vehicles and selects a set of collectors and an analyzer for each task. The selected collectors are required to send the point cloud data within the object's bounding box to the analyzer. Then, the analyzer classifies the object based on the received point clouds and sends the classification result to the smart vehicles that requested the service. Given that the resource consumption of the object classification stage is significantly higher than that of the detection stage, we omit the processing time of the detection stage and focus on resource management during the classification stage.

3.2 Object Classification Task Model

As described in the previous section, there are I unknown objects in the service region. In the case of heavy traffic, an unknown object can be found in the front blind spots of multiple smart vehicles. To avoid repeated processing, the controller generates one object classification task for each unknown object and each task is processed only once. Let \mathcal{I} denote both the set of unknown objects and the corresponding set of tasks, with $i \in \mathcal{I}$ representing both the unknown object index and the task index. Specifically, task i is dedicated to acquiring the category information of unknown object i .

For each task $i \in \mathcal{I}$, the processing flow is as follows: First, all collectors send the point cloud data of the object to the analyzer. Then, the analyzer processes a classification neural network to identify the category of the object. Let t_i^T denote the time consumed by the point cloud transmission and t_i^P denote the time consumed by processing the classification neural network. Denote the beginning moment of each service cycle as moment 0. Assume that the collectors start transmission immediately after receiving the collector selection decision and the analyzer starts processing the classification neural network immediately after receiving all the point cloud data. The moment at which the analyzer starts processing the classification neural network is t_i^T . Let t_i be the moment at which task i is completed, which is given by

$$t_i = t_i^T + t_i^P. \quad (3.1)$$

Let \mathcal{N}_i denote the collector set of task i . For each collector $j \in \mathcal{N}_i$, the point cloud data to be transmitted is the point cloud collected for object i . Let $n_{i,j}$ denote the number of points in the point cloud data. The data size of the point cloud data, denoted by $s_{i,j}$,

is given by $s_{i,j} = \epsilon n_{i,j}$, where ϵ in bits is the data size of one point in the point cloud data. Let n_i denote the total number of points received by the analyzer, which is given by $n_i = \sum_{j \in \mathcal{N}_i} n_{i,j}$.

The processing of the classification neural network can be divided into two steps: point cloud fusion and object classification. In the point cloud fusion step, the neural network concatenates the point cloud data from each collector and extracts the features of each point by using a pre-trained feature learning network [1]. Assume that all smart vehicles in the service region employ the same feature learning network. Let c be the computing resources consumed by the feature learning network to process one point, the size of which is measured by the consumed central processing unit (CPU) cycles. The overall computing resources consumed by the point cloud fusion step, denoted by c_i^f , is given by $c_i^f = c \sum_{j \in \mathcal{N}_i} n_{i,j}$.

In the classification step, a pre-trained convolutional neural networks (CNNs) is used to estimate the probability that the object belongs to each of the M categories [2]. Assume that all smart vehicles in the service region employ the same pre-trained CNNs. Since the pre-trained CNNs has a fixed network structure, the processing will consume a fixed amount of computing resources. Let c_0 denote the computing resources consumed by the CNNs. The overall computing resources consumed by processing the object classification task i , denoted by c_i , is given by

$$c_i = c_i^f + c_0. \quad (3.2)$$

After processing the object classification task, the analyzer outputs an M -dimensional prediction vector $P_i = \{P_{i,1}, \dots, P_{i,M}\}$, where the element $P_{i,m}$ represents the predicted probability that the object belongs to category m . The analyzer then selects the category

with the highest probability as the category of the object. In this study, we adopt the confidence level (also known as confidence score) to assess the accuracy of the classification results [49]. Let η_i denote the normalized information entropy of the M -dimensional output vector, which is given by

$$\eta_i = - \sum_{m=1}^M \frac{P_{i,m} \log_2 P_{i,m}}{\log_2 M}. \quad (3.3)$$

The confidence level of the classification result denoted by μ_i is defined as $\mu_i = 1 - \eta_i$, where μ_i ranges from 0 to 1 [50]. A confidence level close to 1 indicates high classification accuracy, while a confidence level close to 0 indicates low classification accuracy. In the processing of the object classification task, the classification accuracy is closely related to the input data [33, 40]. Here, we denote the classification accuracy of task $i \in \mathcal{I}$ as a function of the set \mathcal{N}_i , which is expressed as

$$\mu_i = U(\mathcal{N}_i). \quad (3.4)$$

The function will be further discussed in Chapter 4.

3.3 Point Cloud Quality Model

3.3.1 LiDAR Sensor Model

In this study, smart vehicles use mechanical multi-line LiDAR sensors to collect sensor data from unknown objects. A LiDAR sensor operates by emitting multiple beams of laser light toward the target object and measuring both the time it takes for each beam to reflect back from the object's surface and the intensity of the reflection. Based on the time-of-flight of

the beam, the LiDAR sensor accurately determines the specific locations where each laser beam hits the object. The location information and the intensity information collected by each beam are then aggregated into a point cloud, where each point represents a specific location on the object’s surface, along with the texture characteristics of that particular location.

Figure 3.3 shows the structure of a mechanical multi-line LiDAR sensor [28, 51]. Specifically, the LiDAR sensor has a top lid that protects against rain and dust, a vertical laser array for emitting and receiving laser beams and a stepped rotating seat for adjusting laser beam orientation. Let e be the number of lines of a laser array. During the sensing operation, the laser array emits e laser beams simultaneously in the vertical direction. As shown in Figure 3.3(a), due to the small size of the laser array, these laser beams can be regarded as emitting from a single point. The point is referred to as the optical center of the LiDAR. Let the interval $[-\theta, \phi]$ denote the vertical sensing range of the laser array, with e laser beams uniformly distributed across this range. The elevation resolution of the LiDAR sensor, denoted by b , is defined as the angle between two adjacent vertical laser beams. For example, the Velodyne ULTRA Puck has a laser array that can simultaneously emit 32 laser beams across the vertical range of $[-25^\circ, 15^\circ]$ with an elevation resolution of $b = 0.4^\circ$ [28]. Each time the laser array completes a sensing operation, the rotating seat moves counterclockwise by a specific angle before initiating the next sensing cycle. This process repeats until a full rotation is achieved. As shown in Figure 3.3(b), the angle of each rotation, denoted as a , is referred to as the azimuth resolution. For example, the Velodyne ULTRA Puck has an azimuth resolution of $a = 0.1^\circ$ [28]. Once the rotation is complete, the LiDAR sensor aggregates the sensor data and stores it as a point cloud.

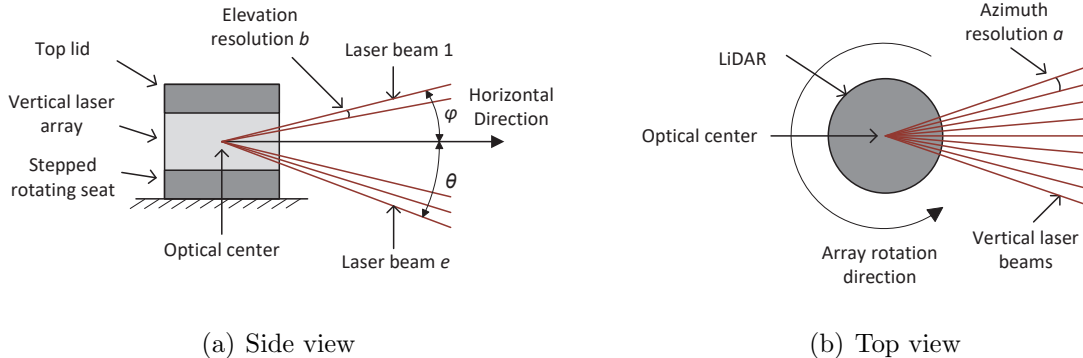


Figure 3.3: Mechanical multi-line LiDAR sensor structure.

During the sensing operation, the LiDAR sensor establishes a three-dimensional coordinate system with its optical center as the origin. The vehicle’s direction of travel determines the x-axis. The x-y plane is parallel to the ground. The z-axis is perpendicular to the ground. Each time the laser array emits e laser beams, the LiDAR sensor records the intensity of each reflected laser beam and the 3D coordinates of each point detected by the laser beam. The intensity, location, and 3D coordinates of the LiDAR sensor are then compiled into a seven-dimensional array. The LiDAR sensor aggregates the seven-dimensional array of all the points collected during the sensing operation and stores the data as a file, which is known as a point cloud [28, 52].

3.3.2 Point Cloud Quality Indicator

Considering that existing object detection and classification algorithms commonly model objects as bounding boxes with cuboid shapes [1, 16], to facilitate integration with these algorithms, we represent both smart vehicles and objects by their bounding boxes. For

object $i \in \mathcal{I}$, its bounding box is defined in the global coordinate system by the following key parameters: the geometric center (x_i, y_i, z_i) , which are the 3D coordinates of the center point of the cube; the dimensional parameters (l_i, w_i, h_i) , which denote the distances from the center of the cube to the long, wide, and high edges, respectively; and the rotational parameters ϕ_i (Roll, rotating around x-axis), θ_i (Pitch, rotation around y-axis) and ψ_i (Yaw, rotation around z-axis), which together determine the orientation of the cube in space. Since the rotation angles in the objects' roll and pitch directions are negligible, we ignore these two parameters. Then the bounding box o_i of object i can be represented by a 7-tuple, i.e. $o_i = \{x_i, y_i, z_i, l_i, w_i, h_i, \psi_i\}$.

Coverage Area

Following existing studies, we deploy LiDAR sensors at the center of the roof of smart vehicles [26, 53]. Due to the low deployment height, part of the laser beams emitted by the LiDAR sensor are obstructed by the roof itself. The remaining beams that reach the object mostly hit its side faces, with only a few reaching the top face. For simplification in our analysis, we assume that the LiDAR sensor has a sensing dead zone with a radius of d_s and we focus on the laser beams that hit the four side faces of an object located outside this dead zone. We define the face directly in front of the object as the first face label the four side faces in counterclockwise order. Let ν denote the face index, where $\nu \in \{1, 2, 3, 4\}$. Since the propagation of the laser beam can be obstructed by surrounding obstacles, we define the coverage area of the collector j on the ν -th face of the object i as the part of the face that can be directly detected by the LiDAR sensor. For further simplification, the coverage area is represented as a rectangular region with the same height as the object.

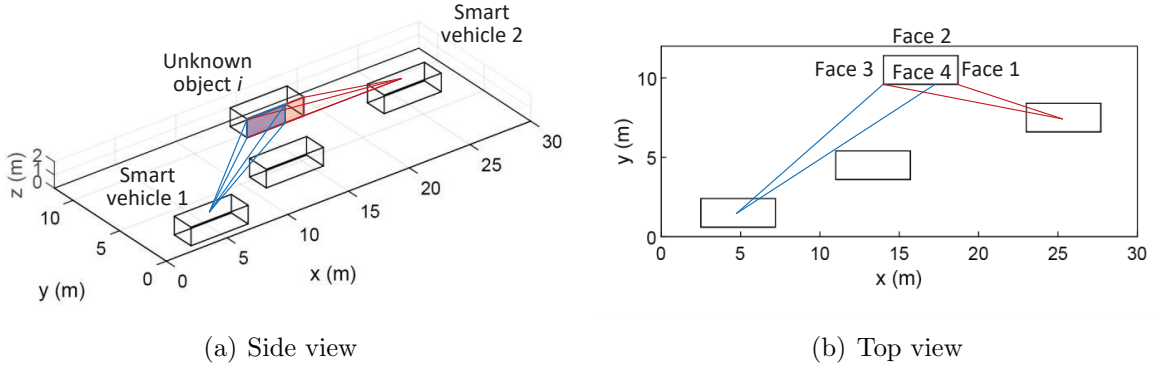


Figure 3.4: An illustration of collectors' coverage areas.

As shown in Figure 3.4, collector 1's coverage area on the 4th face of object i is depicted as a blue rectangle, and collector 2's coverage area is the red one.

In the top view of object i , the bounding box is represented as a rectangle with a length of $2 \times l_i$ and a width of $2 \times w_i$. Let $\nu \in 1, 2, 3, 4$ denote the index of each edge of the rectangle. The length of the ν -th edge of the rectangle is denoted by d_ν . The coverage area of collector j on the ν -th face of the object i , denoted by $q_{i,j}^\nu$, can be represented by a segment of the edge ν [54]. Figure 3.5(a) illustrates a top view of object i in the coordinate system with the LiDAR optical center of collector j as the origin. The segment $A_{i,1}^4 B_{i,1}^4$ denote the coverage area of collector j with endpoints $A_{i,1}^4(x_a, y_a)$ and $B_{i,1}^4(x_b, y_b)$. Let q_i^ν denote the coverage area of the fused point cloud on the ν -th face of the object i . It is defined as the union of all collectors' coverage areas, expressed as $q_i^\nu = \cup_{j \in \mathcal{N}_i} q_{i,j}^\nu$.

Point Cloud Resolution

Let $L_{i,j}^\nu$ denote the set of vertical laser beam arrays emitted by collector j that hit the coverage area $q_{i,j}^\nu$. It is given by $L_{i,j}^\nu = \{l | l \in \mathbb{Z}, \lceil \tan^{-1}(\frac{y_a}{x_a}) \rceil \leq la \leq \lfloor \tan^{-1}(\frac{y_b}{x_b}) \rfloor\}$, where l

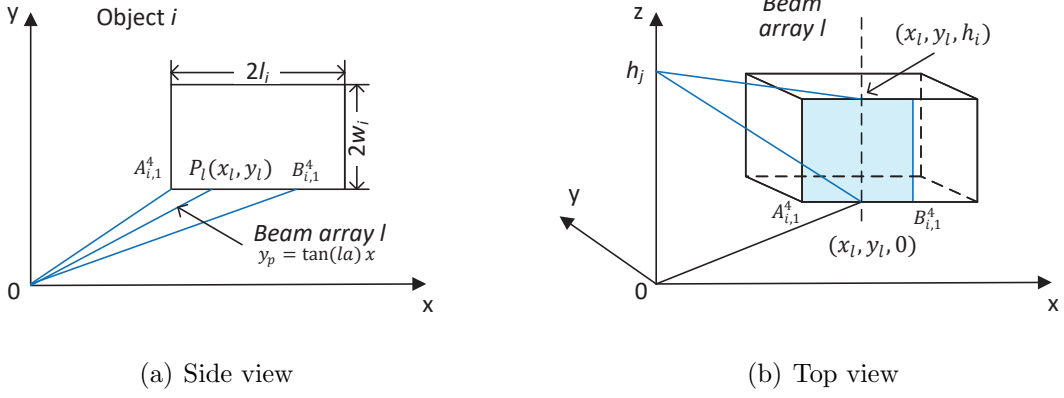


Figure 3.5: An illustration of collector j 's coverage area.

denote the beam array index [54]. For each vertical laser beam array $l \in L_{i,j}^\nu$, its intersection with the face ν in the top view is denoted by P_l with coordinates (x_l, y_l) . Let y_p denote the line connecting the origin of the coordinate system to P_l , which is given by $y_p = \tan(la)x$. Let y_{AB} denote the line connecting the two points, $A_{i,1}^4(x_a, y_a)$ and $B_{i,1}^4(x_b, y_b)$, which is given by $y_{AB} = \frac{y_b - y_a}{x_b - x_a}x + (y_a - \frac{y_b - y_a}{x_b - x_a}x_a)$. Then the coordinates of P_l can be expressed as

$$\begin{cases} x_l = \frac{m_l x_a - y_a}{m_l - \tan(la)}, \\ y_l = \frac{\tan(la)(m_l x_a - y_a)}{m_l - \tan(la)}, \end{cases} \quad (3.5)$$

where $m_l = \frac{y_b y_a}{x_b - x_a}$. Let $P_{i,j}^{\nu,l}$ denote the set of points emitted by the vertical laser beam array l that hit the object i . It is given by $P_{i,j}^{\nu,l} = \{p | p \in \mathbb{Z}, [\tan^{-1}(\frac{\sqrt{x_l^2 + y_l^2}}{h_j - h_i})] \leq pb \leq [\tan^{-1}(\frac{\sqrt{x_l^2 + y_l^2}}{h_j})]\}$, where p is the point index and h_j is the height of the optical center of collector j 's LiDAR [54]. Let $N_{i,j}^{\nu,l}$ be the number of points in the set $P_{i,j}^{\nu,l}$. The number of points collected by collector j in the ν -th face of object i , denoted by $N_{i,j}^\nu$, can be expressed as $N_{i,j}^\nu = \sum_{l \in L_{i,j}^\nu} N_{i,j}^{\nu,l}$. Let $d_{i,j}^\nu$ denote the length of line segment $A_{i,1}^4 B_{i,1}^4$, which is given by

$d_{i,j}^\nu = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}$ [51, 54]. Let $\rho_{i,j}^\nu$ denote the resolution of the point cloud collected by collector j on the ν -th face of object i . It is defined by the density of the point cloud collected on the ν -th face [54], which is given as

$$\rho_{i,j}^\nu = \frac{N_{i,j}^\nu}{d_{i,j}^\nu h_i}. \quad (3.6)$$

Quality Indicator

For object $i \in \mathcal{I}$, the point cloud resolution of its ν -th face depends on the resolutions of point clouds provided by the collectors in the set \mathcal{N}_i . In areas covered by only one collector, the point cloud resolution is that of the only collector in the area. For example, as shown in Figure 3.4(a), the point cloud resolution of the red area on the right-hand side of the object i 's 4-th face is determined by the point cloud collected by smart vehicle 2. In the case where the coverage areas of multiple collectors are overlapped, the resolution of the fused point cloud is determined by the highest-resolution data provided by the collectors involved. This is because the high-resolution point cloud contains information from those with lower resolutions.

To quantify this, we consider the top view of the object. Define $\mathbf{1}_j(\delta)$ as an indicator function where $\mathbf{1}_j(\delta) = 1$ if the area δ of the ν -th face of object is covered by the collector j , and $\mathbf{1}_j(\delta) = 0$ otherwise. The point cloud quality for the ν -th face, denoted by $q_i^\nu(\mathcal{N}_i)$, is defined as the average point cloud resolution over the face, given by

$$q_i^\nu(\mathcal{N}_i) = \frac{1}{d_\nu} \int_0^{d_\nu} \max_{j \in \mathcal{N}_i} \{\rho_{i,j}^\nu\} \mathbf{1}_j(\delta) d\delta. \quad (3.7)$$

The overall quality of the fused point cloud for object i , denoted by $q_i(\mathcal{N}_i)$ is the sum of the qualities across all faces, which is given by

$$q_i(\mathcal{N}_i) = \sum_{\nu=1}^4 q_i^\nu(\mathcal{N}_i). \quad (3.8)$$

It also reflects the average density of points collected from the object i .

For a classification neural network, existing studies have shown that, once the density of the point cloud exceeds a certain threshold ρ_0 , further increasing the density will no longer significantly improve the classification accuracy. To effectively compare the quality of the point cloud data provided by different collector sets, we use the threshold ρ_0 to normalize the point cloud quality. The normalized point cloud quality, denoted by $\tilde{q}_i(\mathcal{N}_i)$, is given by

$$\tilde{q}_i(\mathcal{N}_i) = \frac{\min\{q_i(\mathcal{N}_i), \rho_0\}}{\rho_0}, \quad (3.9)$$

which ensures that the quality indicator ranges from 0 to 1, where 1 indicates that the point cloud density has reached or exceeded the threshold, ρ_0 .

3.4 Communication Model

In this study, there are two types of communication links, namely vehicle-to-network (V2N) links and vehicle-to-vehicle (V2V) links. The smart vehicle exchanges object location, collector selection decision and analyzer selection decision information with the BS via the V2N link and sends the point cloud data to the analyzer via the V2V link. Considering that the size of point cloud data is much larger than that of the object location and vehicle

selection decision information, we focus on point cloud data transmission via V2V links.

The V2V links between smart vehicles are established via orthogonal frequency-division multiplexing access (OFDMA) [55]. Since smart vehicles are equipped with advanced communication modules, each smart vehicle can establish V2V links with multiple smart vehicles simultaneously for both data transmission and data receiving. Let B denote the overall available radio spectrum bandwidth in the service region and $\beta_{j,i}$ denote the fraction of the bandwidth allocated to collector j for task i . The bandwidth allocated to collector j for task i is given by $\beta_{j,i}B$. Considering that the bandwidth allocated to each channel should not exceed the available bandwidth, we have

$$\sum_{j \in \mathcal{N}} \sum_{i \in \mathcal{I}} \beta_{j,i} \leq 1. \quad (3.10)$$

Let $E_{j,k}$ denote the spectrum efficiency of the link established between collector j and analyzer k , which is given by

$$E_{j,k} = \log_2 \left(1 + \frac{p_j g_{j,k}}{\sigma^2} \right), \quad (3.11)$$

where p_j denotes the transmission power of collector j , $g_{j,k}$ denotes the channel gain between collector j and analyzer k and σ^2 denotes the receiving noise power at analyzer k . The channel gain $g_{j,k}$ is primarily influenced by path loss and small-scale fading. Assuming that smart vehicles' communication modules can effectively mitigate the small-scale fading, the channel gain can be represented by a simplified path loss model introduced in [56], given by

$$g_{j,k} = K \left(\frac{d_0}{d_{j,k}} \right)^\gamma, \quad (3.12)$$

where K denotes the channel gain per unit distance, d_0 denotes the reference distance, $d_{j,k}$ denotes the distance between collector j and analyzer k , and γ denotes the path loss exponent, with values ranging from 2.7 to 3.5 in urban microcell scenarios [57]. Therefore, the transmission rate of the link established between collector j and analyzer k for task i , denoted by $r_{j,k}^i$, is given by

$$r_{j,k}^i = \beta_{j,i} B E_{j,k}. \quad (3.13)$$

For each V2V link, the time consumed by the data transmission is the ratio of the data size to the transmission rate. Therefore, the time it takes for collector j to send the point cloud data of task i to analyzer k , denoted by $t_{i,j}^{T,k}$, is given by $t_{i,j}^T = \frac{s_{i,j}}{r_{j,k}^i}$. If the collector and the analyzer are the same vehicle, the data transmission time $t_{i,j}^{T,k}$ is equal to 0 as $j = k$. Therefore, the transmission time of the point cloud data from collector j to analyzer k for task i is

$$t_{i,j}^{T,k} = \begin{cases} \frac{s_{i,j}}{r_{j,k}^i}, & \text{if } j \neq k \\ 0, & \text{otherwise.} \end{cases} \quad (3.14)$$

Given that the collectors can transmit point cloud data to the analyzer simultaneously, the completion time for the point cloud data transmission for task i is determined by the latest completion time among all transmissions. It is mathematically represented as

$$t_i^T = \max\{t_{i,j}^{T,k} : \forall j \in \mathcal{N}_i\}. \quad (3.15)$$

3.5 Computing Model

Equipped with an advanced computing platform, smart vehicles can process multiple types of computing tasks. By allocating computing resources (also known as CPU frequency) to different computing tasks, smart vehicles can process multiple tasks in parallel [58]. Specifically, a smart vehicle allocates its computing resources to first meet the demands of its own tasks, such as navigation and music playback. Any resources left over are then used to process the object classification task. Given the short service period of the vehicular see-through service, we assume that a smart vehicle's available computing resources remain constant during a service period.

Let f_k in cycles per second denotes the smart vehicle k 's available computing resources. Assume that smart vehicle k is the analyzer of task i . Let $\alpha_{k,i}$ denote the fraction of the computing resources allocated to task i . The amount of computing resources allocated to task i is given by $\alpha_{k,i}f_k$. Considering that the computing resources allocated to each object classification task should not exceed the available resources of the analyzer, we have

$$\sum_{i \in \mathcal{I}_k} \alpha_{k,i} \leq 1, \forall k \in \mathcal{N}, \quad (3.16)$$

where \mathcal{I}_k is the set of tasks assigned to smart vehicle k . Given that the processing time of a computing task is the ratio of the computing workload of the task to the computing resources allocated to the task, the time it takes for vehicle k to process the classification neural network of task i can be calculated according to [58], that is

$$t_i^P = \frac{c_i}{\alpha_{k,i}f_k}. \quad (3.17)$$

3.6 Summary

In this chapter, we present a system model for vehicular see-through service provisioning. The model consists of a scenario model describing the region where the service is provided, a task model describing the generation and processing flow of the object classification task and task parameters, a point cloud quality model describing the proposed assessment indicator for the fused point cloud, a communication model describing the point cloud transmission delay, and a computing model describing the object classification task processing delay.

Chapter 4

Problem Formulation and Solution

4.1 Problem Formulation

In this section, the resource consumption minimization problem in the object classification stage of the vehicular see-through service provisioning is formulated.

To formalize the objective function, we first denote several decision variables. Let $\mathbf{U} = \{u_{i,j} : \forall i \in \mathcal{I}, \forall j \in \mathcal{N}\}$ denote the collector selection decision matrix, with $u_{i,j} = 1$ if smart vehicle j is selected as a collector for task i , and $u_{i,j} = 0$ otherwise. Let $\mathbf{V} = \{v_{i,k} : \forall i \in \mathcal{I}, \forall k \in \mathcal{N}\}$ denote the analyzer selection decision matrix, with $v_{i,k} = 1$ if smart vehicle k is selected as the analyzer for task i , and $v_{i,k} = 0$ otherwise. Let $\boldsymbol{\alpha}$ denote the computing resource allocation decision matrix, which is given by $\boldsymbol{\alpha} = \{\alpha_{k,i} : \forall i \in \mathcal{I}, \forall k \in \mathcal{N}\}$. Let $\boldsymbol{\beta}$ denote the communication resource allocation decision matrix, which is given by $\boldsymbol{\beta} = \{\beta_{j,i} : \forall i \in \mathcal{I}, \forall j \in \mathcal{N}\}$.

For each task $i \in \mathcal{I}$, the communication resource consumption is the sum of the band-

width occupied by all its collectors in transmitting the sensor data of object i , which is given by $\sum_{j \in \mathcal{N}} u_{i,j} \beta_{j,i} B$. In addition, the computing resources consumed by task i are measured by the CPU frequency allocated to the task, which is given by $\sum_{k \in \mathcal{N}} v_{i,k} \alpha_{k,i} f_k$. To concisely represent the resource consumption of the task, we formulate the overall resource consumption as a weighted sum of the communication resource consumption and computing resource consumption as described in [59]. Let O denote overall resource consumption for service provisioning, which is given by

$$O = w \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{N}} u_{i,j} \beta_{j,i} B + (1 - w) \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{N}} v_{i,k} \alpha_{k,i} f_k, \quad (4.1)$$

with weighting factor $w \in [0, 1]$. The selection of the weighting factor w depends on the relative importance of communication resource consumption and computing resource consumption in the system. For example, a value of w close to 0 means that the system places more importance on computing resource consumption. Conversely, setting w to 1 indicates that the system cares more about reducing communication resource consumption.

In the considered two-stage cooperative scheme, each task $i \in \mathcal{I}$ can select multiple smart vehicles as its collectors. The set of selected smart vehicles for task i is denoted by $\mathcal{N}_i = \{j : \forall u_{i,j} = 1\}$. To avoid unnecessary consumption of communication resources, the smart vehicles that are not selected as the collectors for task i should not establish a V2V link for task i . So, we have

$$\beta_{j,i} = 0, \text{ if } u_{i,j} = 0, \forall i \in \mathcal{I}, \forall j \in \mathcal{N}. \quad (4.2)$$

For each task $i \in \mathcal{I}$, the controller selects at most one analyzer for data processing, i.e.,

$$\sum_{k \in \mathcal{N}} v_{i,k} \leq 1, \forall i \in \mathcal{I}. \quad (4.3)$$

For the smart vehicle that is not selected as an analyzer by task i , the controller should not require the smart vehicle to allocate computation resources for that task, i.e.,

$$\alpha_{k,i} = 0, \text{ if } v_{i,k} = 0, \forall i \in \mathcal{I}, \forall k \in \mathcal{N}. \quad (4.4)$$

In this study, two service requirements, namely service delay and classification accuracy are considered. Let τ be the delay requirement of vehicular see-through services, which is less than or equal to the duration of a service period. To satisfy the delay requirement of the service, we have

$$t_i \leq \tau, \forall i \in \mathcal{I}. \quad (4.5)$$

Let μ_0 be the classification accuracy requirement of the service. To satisfy this requirement, the selected collector set \mathcal{N}_i of any task $i \in \mathcal{I}$ should satisfy the following constraint:

$$\mu_i \geq \mu_0, \forall i \in \mathcal{I}. \quad (4.6)$$

To minimize the overall resource consumption O for the vehicular see-through service provision, we formulate a joint collector set selection, analyzer selection and resource allo-

cation problem as optimization problem P1, which is shown as follows

$$(P1) \min_{\alpha, \beta, \mathbf{U}, \mathbf{V}} O \quad (4.7a)$$

$$\text{s.t.} \quad (4.2), (3.10), (4.3), (4.4), (3.16), (4.5), (4.6), \quad (4.7b)$$

$$u_{i,j} = \{0, 1\}, \forall i \in \mathcal{I}, \forall j \in \mathcal{N}, \quad (4.7c)$$

$$v_{i,k} = \{0, 1\}, \forall i \in \mathcal{I}, \forall k \in \mathcal{N}, \quad (4.7d)$$

$$0 \leq \alpha_{k,i} \leq 1, \forall k \in \mathcal{N}, \forall i \in \mathcal{I}, \quad (4.7e)$$

$$0 \leq \beta_{j,i} \leq 1, \forall j \in \mathcal{N}, \forall i \in \mathcal{I}. \quad (4.7f)$$

Constrain (4.2) requires that V2V links should only be established when there is a transmission demand between smart vehicles. Constrain (3.10) requires that the bandwidth allocated to each V2V link should not exceed the available bandwidth. Constrain (4.3) requires that at most one analyzer can be selected for each task. Constrain (4.4) requires that smart vehicles should not allocate computing resources to tasks that are not assigned to them. Constrain (3.16) requires that the computing resources allocated to each task should not exceed the available resources of the analyzer. Constrain (4.5) requires that the service delay requirement for each task should be satisfied. Constrain (4.6) requires that the classification accuracy requirement for each task should be satisfied. Constrains (4.7c-4.7f) are range requirements for decision variables.

4.2 Problem Solution

In this section, we develop a joint collector set and analyzer selection scheme with resource allocation to minimize the overall resource consumption for the vehicular see-through service provision. To meet the classification accuracy requirement of the service, we first characterize the relationship between collector selection and object classification accuracy. Then, we design a backtracking algorithm to find all collector sets that satisfy the accuracy requirement with minimal data redundancy. To minimize the overall resource consumption while satisfying the classification accuracy and service delay requirements, we consider each collector set-analyzer pair of a task as a single unit and assign selection priorities based on their resource consumption. Subsequently, we propose a solution based on the ACO algorithm to select the collector set-analyzer pair for each task and allocate resources accordingly.

4.2.1 Classification Accuracy Fulfillment

In Section 3.3, we define a point cloud quality indicator, $\tilde{q}_i(\mathcal{N}_i)$, for assessing the selected collector set. To investigate the relationship between sensing quality and object classification accuracy, we collect point cloud data from multiple smart vehicles traveling in different driving environments. For each driving environment, we calculate the point cloud quality of the fused point cloud data and obtain the object classification accuracy by processing a pre-trained classification neural network.

Since real-world datasets cannot provide point cloud data from multiple viewpoints for the same object, existing studies often use driving simulators to generate sensor data.

In this study, we generate point cloud data by using the MATLAB Automated Driving Toolbox [60]. Specifically, we consider a 300-meter road segment with five lanes, each of which has a width of 3 meters. In the road segment, we consider four types of objects: sedans, trucks, bicycles, and walkers. For each type of object, we place it in the center of the road segment and randomly place 5 objects on the road segment to simulate obstructed views. For each smart vehicle, we model its shape as a sedan and deploy a 32-line LiDAR in the center of its roof with azimuth resolution $a = 0.1^\circ$ and elevation resolution $b = 0.4^\circ$ [28]. We also train a Voxnet for object classification. It is recommended in [2] that the Voxnet has a point cloud density threshold of 10 point/m³. Here we set the point cloud density threshold $\rho_0 = 10$ point/m³.

To generate the point cloud in different driving environments, we consider four types of sensing, namely independent sensing, two-vehicle cooperation, three-vehicle cooperation and four-vehicle cooperation. In the independent sensing scenario, a smart vehicle is sequentially placed on the centerline of each lane. In each lane, the smart vehicle collects a point cloud for every meter it travels. We repeat this process on all five lanes and eventually obtain a total number of 1,400 point clouds. In the two-vehicle cooperation scenario, we consider the cases where two smart vehicles are driving on the same lane and on different lanes and generate 2,520 driving environments. By fusing the point clouds collected from these two vehicles, we obtained the fused point cloud from each driving environment. In the three-vehicle cooperative scenario, we consider the case of two vehicles traveling in the same lane and the case of three vehicles in three different lanes and obtain 3,360 fused point clouds. In the four-vehicle cooperative scenario, we consider the case of two vehicles traveling in the same lane and the case of four vehicles in different lanes and obtain 6,720

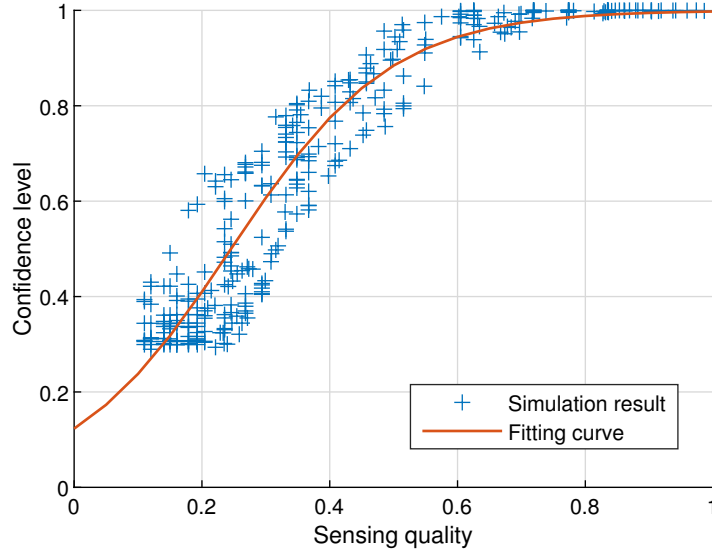


Figure 4.1: Relationship between point cloud quality and classification accuracy.

fused point clouds.

For each point cloud, we calculate the point cloud quality and obtain the classification accuracy by processing a pre-trained classification neural network. Then we set up a rectangular coordinate system with the point cloud quality as the x-axis and the classification accuracy as the y-axis and plot each point cloud in this coordinate system. Due to the overfitting issue of the classification neural network, some fused point clouds have abnormal classification accuracies. Here, we remove the fused point clouds whose classification accuracy is much lower than that of the independent sensing. The point cloud quality and classification accuracy of each point cloud data are shown in Figure 4.1.

As can be seen from this figure, the classification accuracy first increases rapidly with the improvement of the point cloud quality, and then gradually tends to saturate. However, for the point clouds with same point cloud quality, their classification accuracies may have

multiple values. This is because classification accuracy is not solely determined by point cloud quality and may also be influenced by other unobserved variables. In addition, the classification neural network has the characteristic of randomness. Even if the same point cloud data is fed into the same neural network, different classification accuracies may be obtained.

To obtain the relationship between point cloud quality and classification accuracy, we can use the cumulative distribution function (CDF) to show the probability that a given combination of collectors meets the accuracy requirement at a specific point cloud quality. However, this method cannot help us identify whether a combination of collectors can meet the accuracy requirement. By observing the Figure 4.1, we find that the trend between point cloud quality and classification accuracy is consistent with the characteristics of the S-curve. To capture this relationship, we use a trendline analysis method to fit the data with an S-curve. The fitting results are as follows:

$$\mu_i = \frac{a_s}{1 + e^{-b_s(\bar{q}_i(\mathcal{N}_i) - c_s)}}, \quad (4.8)$$

where $a_s = 1$, $b_s = 7.9966$, and $c_s = 0.2456$. This curve simplifies the relationship between point cloud quality and classification accuracy into a one-to-one mapping. Although the classification accuracy of a point cloud is a random variable at a given point cloud quality, this curve provides a predictive value for classification accuracy. Therefore, we use this curve to identify collector combinations that meet the accuracy requirement.

To meet the classification accuracy requirement μ_0 , the point cloud quality of the

selected collector set should be no less than the threshold value \tilde{q}_0 , which is given by

$$\tilde{q}_0 = -\frac{1}{b_s} \ln \left(\frac{a_s}{\mu_0} - 1 \right) + c_s. \quad (4.9)$$

Therefore, the accuracy constraint (4.6) in problem P1 is reformulated as

$$\tilde{q}_i(\mathcal{N}_i) \geq \tilde{q}_0. \quad (4.10)$$

To ensure that the selected collector sets meet the required classification accuracy, we pre-select potential collector sets for each object based on the point cloud quality model. There are two benefits of the pre-selection. On one hand, by assessing the classification accuracy of potential collector sets, we can filter out the collector sets whose accuracy cannot meet the service requirements, thus the selection space can be reduced. On the other hand, when the classification neural network fuses some point clouds with complementary coverage and high resolution, it is able to accurately classify the object. Continuing to fuse new point clouds will lead to an accuracy that exceeds the service requirement. By pre-selecting the collector sets, we can identify the collector sets with minimal data redundancy.

We design a backtracking algorithm as shown in Algorithm 1 to identify the collector sets with minimal data redundancy. Let \mathcal{N}_i^c denote the set of all collector sets for object i that satisfy the accuracy requirement and have the minimal data redundancy where each element is referred to as a valid collector set for object i . For each object, the algorithm starts with an empty collector set $\mathcal{N}_i^t = \Phi$ and recursively tries to add each unexplored smart vehicle n to the current collector set \mathcal{N}_i^t in descending order of point cloud quality. In each recursion, if point cloud quality \tilde{q}_i^t of the current collector set \mathcal{N}_i^t reaches the

Algorithm 1 Find all potential collector sets with minimal data redundancy

Input: List of tasks \mathcal{I} , List of smart vehicles \mathcal{N} , Threshold value \tilde{q}_0

Output: List of valid collector sets \mathcal{N}_i^c for each task $i \in \mathcal{I}$

```

1: for each task  $i \in \mathcal{I}$  do
2:   Sort smart vehicles by point cloud quality in descending order
3:   Initialize an empty list  $\mathcal{N}_i^c$  to store valid collector sets
4:   Call BACKTRACK( $\mathcal{N}$ , empty set  $\Phi$ , 0)
5: end for
6: function BACKTRACK(remaining smart vehicle set  $\mathcal{N}^r$ , current set  $\mathcal{N}_i^t$ , current quality  $\tilde{q}_i^t$ )
7:   if  $\tilde{q}_i^t \geq \tilde{q}_0$  then
8:      $\mathcal{N}_i^c = \mathcal{N}_i^c \cup \mathcal{N}_i^t$ 
9:     Return
10:  else
11:    Calculate the point cloud quality of  $\mathcal{N}_i^t$  for each remaining smart vehicle after
    it has been added to  $\mathcal{N}_i^t$ 
12:    Sort the remaining smart vehicles by point cloud quality in descending order
13:    for each remaining smart vehicle  $n \in \mathcal{N}^r$  do
14:      Add smart vehicle  $n$  to the current set  $\mathcal{N}_i^t$ 
15:      Calculate the point cloud quality  $\tilde{q}_i^t$  of the current set  $\mathcal{N}_i^t$ 
16:      if  $\tilde{q}_i^t \geq \tilde{q}_0$  then
17:         $\mathcal{N}_i^c = \mathcal{N}_i^c \cup \mathcal{N}_i^t$ 
18:      else
19:        Remove all remaining smart vehicles that have been checked from  $\mathcal{N}^r$ 
20:        Call BACKTRACK( $\mathcal{N}^r$ ,  $\mathcal{N}_i^t$ ,  $\tilde{q}_i^t$ )
21:      end if
22:    end for
23:  end if
24:  Return result  $\mathcal{N}_i^c$ 
25: end function

```

minimum requirement q_0 , this set is recorded in \mathcal{N}_i^c and the algorithm stops adding new smart vehicles to the set \mathcal{N}_i^t . The algorithm then reverts to the state before the last smart vehicle n was added and tries to add the next unexplored smart vehicle to set \mathcal{N}_i^t , thus exploring all possible combinations that might satisfy the requirements. However, if the

point cloud quality does not meet the minimum requirement q_0 , the smart vehicle n is added to the current collector set \mathcal{N}_i^t . The remaining unexplored smart vehicles are sorted in descending order of point cloud quality and the algorithm continues to explore whether the next smart vehicle can be added to the current set. This process is repeated until all smart vehicles have been considered. Since the algorithm stops adding new smart vehicles to the set when the accuracy requirement is satisfied, it can avoid unnecessary exploration. Therefore, it is more efficient than exhaustive search methods.

4.2.2 Joint Vehicle Selection and Resource Allocation

In the previous subsection, we present a pre-selection algorithm for collector set selection. The algorithm identifies all collector sets with minimal data redundancy for each task that meet the classification accuracy requirement. In this subsection, we propose a joint vehicle selection and resource allocation scheme to deal with the tradeoff between service requirements and resource consumption. The main idea of our scheme is to pair all potential collector sets with analyzers for each task and treat each pair as a single unit. We refer to each pair in our scheme as a worker pair. Then, we assign selection priorities to each worker pair based on their resource consumption for completing the transmission and computing of the corresponding task within a service period. The original problem is transformed into a worker pair selection problem with constraints related to communication resources, computing resources, and service delay. To solve this problem, we iteratively adjust the selection priority of each worker pair by using an ACO algorithm. This approach allows us to ultimately derive a solution that meets the service requirements and minimizes overall resource consumption.

Pairing of Collector Sets with Analyzers

Let N_i^c be the total number of valid collector sets of task i , given by $N_i^c = |\mathcal{N}_i^c|$. For each classification task $i \in \mathcal{I}$, its potential analyzer includes all smart vehicles located in the service region. Therefore, each valid collector set of task i can pair with all N smart vehicles and form $N_i^c \times N$ worker pairs. Let l and k denote the collector set index and the analyzer index of a task, respectively. A worker pair of a task can be expressed as $s = (l, k)$.

Here, we treat each worker pair as a single unit. Let \mathcal{N}_i^l denote the set of smart vehicles in task i 's l -th valid collector set. Let k_i denote the analyzer of task i . The communication and computing resources consumed by pair (l_i, k_i) are given by $\sum_{j \in \mathcal{N}_i^l} \beta_{j,i} B$ and $\alpha_{k_i,i} f_{k_i}$, respectively. In problem P1, we formulate the service provisioning problem as a joint collector set selection, analyzer selection and resource allocation problem. Since all valid collector sets of each task have been paired with all potential analyzers, problem P1 can be transformed into a worker pair selection and resource allocation problem. Let $\mathbf{S} = \{s_i : \forall i \in \mathcal{I}\}$ be the worker pair selection decision vector with $s_i = (l_i, k_i)$ be the worker pair selected for task i . The problem is then represented as optimization problem

P2, given by

$$(P2) \min_{\alpha, \beta, \mathbf{S}} w \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{N}_i^l} u_{i,j} \beta_{j,i} B + (1-w) \sum_{i \in \mathcal{I}} \alpha_{k_i,i} f_{k_i} \quad (4.11a)$$

$$\text{s.t.} \quad \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{N}_i^l} \beta_{j,i} \leq 1, \quad (4.11b)$$

$$\sum_{i \in \mathcal{I}} \alpha_{k_i,i} \leq 1, \quad (4.11c)$$

$$t_i \leq \tau, \forall i \in \mathcal{I}, \quad (4.11d)$$

$$l_i \in \mathcal{N}_i^c, \forall i \in \mathcal{I}, \quad (4.11e)$$

$$k_i \in \mathcal{N}, \forall i \in \mathcal{I}, \quad (4.11f)$$

$$0 \leq \alpha_{k_i,i} \leq 1, \forall i \in \mathcal{I}, \forall k_i \in \mathcal{N}, \quad (4.11g)$$

$$0 \leq \beta_{j,i} \leq 1, \forall i \in \mathcal{I}, \forall j \in \mathcal{N}_i^l. \quad (4.11h)$$

To minimize the overall resource consumption, we propose a priority-based worker pair selection scheme. For a worker pair $s = (l, k)$ of task i , the bandwidth occupied by the collector j to transmit its point cloud to analyzer k within a service period, denoted by $\bar{B}_{i,s}^j$, is given by

$$\bar{B}_{i,s}^j = \begin{cases} \frac{s_{i,j}}{\tau E_{j,k}}, & \text{if } j \neq k, \\ 0, & \text{otherwise.} \end{cases} \quad (4.12)$$

The bandwidth occupied by all of the task i 's collectors to transmit point cloud to analyzer k within a service period and the CPU frequency occupied by the analyzer k to process task i 's point cloud data within a service period, denoted by $\bar{B}_{i,s}$ and $\bar{f}_{i,s}$ respectively, are

given by

$$\bar{B}_{i,s} = \sum_{j \in \mathcal{N}_i^t} \bar{B}_{i,s}^j, \quad (4.13)$$

$$\bar{f}_{i,s} = \frac{c \sum_{j \in \mathcal{N}_i^t} n_{i,j} + c_0}{f_k}. \quad (4.14)$$

The overall resource consumption of the worker pair s and the total available resources in the service region, denoted by $\bar{o}_{i,s}$ and o_0 , are given by

$$\bar{o}_{i,s} = w \bar{B}_{i,s} + (1 - w) \bar{f}_{i,s}, \quad (4.15)$$

$$o_0 = wB + (1 - w) \sum_{k \in \mathcal{N}} f_k. \quad (4.16)$$

Let $\eta_{i,s}$ denote the selection priority of a worker pair s , which is defined as

$$\eta_{i,s} = \frac{o_0 - \bar{o}_{i,s}}{o_0}, \quad (4.17)$$

which ranges from $[0, 1]$. When priority $\eta_{i,s}$ is close to 0, the selected worker pair s will consume more resources to complete transmission and computation within a unit of time. On the contrary, when the priority is close to 1, the worker pair has very low resource consumption.

Since worker pairs with higher selection priorities consume fewer resources per unit of time, we can select high-priority worker pairs for each task to reduce overall resource consumption. However, the resources available to each worker pair may vary with the selection of worker pairs. On one hand, worker pairs with the same analyzer will compete

for computing resources. On the other hand, worker pairs also compete for communication resources. As a result, the actual resources allocated to each worker pair may be less than expected, resulting in a longer service delay for the task. To address the issue, in the following section, we present a worker pair selection and resource allocation algorithm to minimize resource consumption while satisfying delay requirements.

Worker Pair Selection and Resource Allocation Algorithm

Considering that the selection of worker pairs has an impact on the available resources, we propose an iterative algorithm based on the ACO algorithm to optimize the selection of worker pairs and resource allocation. In each iteration, we first adjust the selection priority of each worker pair according to the information provided by the ACO algorithm, and then select worker pairs with lower resource consumption. Then, we allocate appropriate resources to the selected worker pairs by solving a resource allocation problem.

The main idea of the ACO algorithm is that in each iteration, a higher weight (also known as a pheromone) is accumulated for each worker pair in solutions that have lower costs and satisfy the delay requirements [61, 62]. In the next iteration, the worker pairs with high weights have a higher probability of being selected. As time goes on, the weights for each worker pair in the lower-cost solutions gradually increase. Finally, the algorithm selects the worker pairs that consume fewer resources and satisfy the delay requirement.

Worker Pair Selection Let T denote the total number of iterations. Let G denote the total number of ants. In the t -th iteration, the pheromone value of each worker pair $s = (l, k)$ of task $i \in \mathcal{I}$ is denoted by $\xi_{i,s}(t)$. At the beginning, the pheromone value of each

worker pair, denoted by $\xi_{i,s}(0)$, is equal to a constant value ξ_0 . In each iteration t , each ant g sequentially selects a worker pair for each task in a probabilistic manner until a complete solution is obtained. Let $\mathbf{S}_g(t) = \{s_i^g(t) : \forall i \in \mathcal{I}\}$ denote the worker pair selection decisions made by ant g in the t -th iteration. Let $\zeta_{i,s}(t)$ denote the selection priority of worker pair s for task i , given by

$$\zeta_{i,s}(t) = \begin{cases} \xi_{i,s}(t)(\eta_{i,s})^\lambda, & \text{if } l \in \mathcal{N}_i^c, \\ 0, & \text{otherwise,} \end{cases} \quad (4.18)$$

where constant λ represents the relative importance of the original selection priority with respect to the pheromone value. The probability that worker pair s is selected for task i , denoted by $p_{i,j}(t)$, is calculated according to [62], which is given by

$$p_{i,j}(t) = \begin{cases} \frac{\xi_{i,s}(t)(\eta_{i,s})^\lambda}{\sum_{r \in \mathcal{N}_i^c} \xi_{i,r}(t)(\eta_{i,r})^\lambda}, & \text{if } l \in \mathcal{N}_i^c, \\ 0, & \text{otherwise,} \end{cases} \quad (4.19)$$

Resource Allocation After each iteration, we need to allocate appropriate resources to the selected worker pairs and calculate the total resource consumption of the selection decisions of all G ants. Then, the ACO updates the pheromone values for each worker pair based on the resource consumption of the selection decisions and the number of times each worker pair has been selected. The resource allocation subproblem for each ant g is given

by

$$(P3) \min_{\hat{\alpha}, \hat{\beta}} w \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{N}} \beta_{j,i} B + (1-w) \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{N}} \alpha_{k,i} f_k \quad (4.20a)$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{N}} \sum_{i \in \mathcal{I}} \beta_{j,i} \leq 1, \quad (4.20b)$$

$$\sum_{i \in \mathcal{I}_k} \alpha_{k,i} \leq 1, \forall k \in \mathcal{N}, \quad (4.20c)$$

$$\frac{s_{i,j}}{\beta_{j,i} B E_{j,k}} + \frac{c_i}{\alpha_{k,i} f_k} \leq \tau, \forall i \in \mathcal{I}, \forall j \in \mathcal{N}, j \neq k \quad (4.20d)$$

$$0 \leq \alpha_{k,i} \leq 1, \forall k \in \mathcal{N}, \forall i \in \mathcal{I}, \quad (4.20e)$$

$$0 \leq \beta_{j,i} \leq 1, \forall j \in \mathcal{N}, \forall i \in \mathcal{I}. \quad (4.20f)$$

where $\hat{\alpha}$ and $\hat{\beta}$ are the communication and computing resource allocation decision matrices for ant g , respectively. Since the (4.20d) is a nonlinear constraint, the subproblem P3 is non-convex. To solve this problem, we transform the subproblem P3 into a second-order cone programming (SOCP) problem P4 by introducing two auxiliary continuous decision matrices, $\hat{\gamma} = \{\gamma_{i,j} : \forall i \in \mathcal{I}, \forall j \in \mathcal{N}\}$ and $\hat{\delta} = \{\delta_{k,i} : \forall i \in \mathcal{I}, \forall k \in \mathcal{N}\}$ [38, 63]. The

problem P4 is shown as follows

$$(P4) \min_{\hat{\alpha}, \hat{\beta}, \hat{\gamma}, \hat{\delta}} w \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{N}} \beta_{j,i} B + (1-w) \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{N}} \alpha_{k,i} f_k \quad (4.21a)$$

$$\text{s.t.} \quad (4.20b), (4.20c), (4.20e), (4.20f) \quad (4.21b)$$

$$\frac{s_{i,j}}{B E_{j,k}} \gamma_{i,j} + \frac{c_i}{f_k} \delta_{k,i} \leq \tau, \forall i \in \mathcal{I}, \forall j \in \mathcal{N}, j \neq k, \quad (4.21c)$$

$$\gamma_{i,j} \beta_{j,i} \geq 1, \forall i \in \mathcal{I}, \forall j \in \mathcal{N} \quad (4.21d)$$

$$\delta_{k,i} \alpha_{k,i} \geq 1, \forall i \in \mathcal{I}, \forall k \in \mathcal{N}. \quad (4.21e)$$

It can be solved by using commercial optimization solvers such as Gurobi and CVX.

Parameter updates Let $H_g(t)$ denote an indicator where $H_g(t) = 1$ if the subproblem P4 has a feasible solution, and $H_g(t) = 0$ otherwise. Let $O_g(t)$ denote the overall resource consumed by the worker pairs selected by ant g in t -th iteration. The pheromone value released by ant g on worker pair j for task i , denoted by $\Delta \xi_{i,s}^g(t)$, is given by

$$\Delta \xi_{i,s}^g(t) = \begin{cases} \frac{1}{O_g(t)}, & \text{if } s = s_i^g(t) \text{ and } H_g(t) = 1, \\ 0, & \text{otherwise,} \end{cases} \quad (4.22)$$

where $s = s_i^g(t)$ means that worker pair s is selected by ant g for task i and $H_g(t) = 1$ means that worker pair s can meet the delay requirement. Let $\Delta \xi_{i,s}(t)$ denote the pheromone value released by all ants on worker pair j for task i , which is given by $\Delta \xi_{i,s}(t) = \sum_{g=1}^G \Delta \xi_{i,s}^g(t)$ [62]. Let constant ρ denote the pheromone persistence coefficient, which indicates the rate of pheromone volatilization. The pheromone value of each worker pair is updated as

described in [62], given by

$$\xi_{i,s}(t+1) = \rho\xi_{i,s}(t) + \Delta\xi_{i,s}(t). \quad (4.23)$$

After T iterations, the selection priority for each worker pair gradually stabilizes. We take the selection decision and resource allocation result with the lowest resource consumption from these T iterations as the solution to the problem.

The joint vehicle selection and resource allocation algorithm can be summarized as follows.

Algorithm 2 Joint vehicle selection and resource allocation algorithm

Input: List of tasks \mathcal{I} , List of smart vehicles \mathcal{N}

Output: Vehicle selection decisions \mathbf{U} and \mathbf{V} , resource allocation decisions $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$

- 1: **for** each task $i \in \mathcal{I}$ **do**
 - 2: Pair each each valid collector set with all smart vehicles
 - 3: Calculate selection priority for each worker pair
 - 4: **end for**
 - 5: **for** $t \leq T$ **do**
 - 6: Calculate selection priority and probability for each worker pair
 - 7: Sequentially select a worker pair for each task in a probabilistic manner
 - 8: Allocate resources to each worker pair by solving subproblem P4
 - 9: Update pheromone value for each worker pair
 - 10: **end for**
 - 11: Return the result with lowest resource consumption
-

4.3 Summary

In this chapter, we formulate a joint collector set selection, analyzer selection and resource allocation problem for the provisioning of vehicular see-through services. To meet the

classification accuracy requirement of the problem, we characterize the relationship between collector selection and object classification accuracy and design a backtracking algorithm to find all collector sets that satisfy the accuracy requirement with minimal data redundancy. To optimize the overall resource consumption while satisfying the service delay requirement, we pair all potential collector sets with analyzers for each task and iteratively adjust their selection priorities. Finally, we propose a solution based on the ACO algorithm to effectively select vehicles and allocate resources to each task.

Chapter 5

Performance Evaluation

5.1 Simulation Setup

The simulation is conducted on a one-dimensional road segment with four lanes. The length of the road segment is 150 meters. The width of each lane is 3 meters. On the left sidewalk of this segment, a utility pole is placed in the median position. The bottom of the pole is one meter from the edge of the nearest drive lane. A BS equipped with a controller and an omni-directional antenna is deployed at a height of 8 meters from the bottom of the pole. The simulation environment is shown in Figure 5.1.

The road segment contains five types of traffic participants, namely, smart vehicles, sedans, trucks, bicycles, and walkers. The smart vehicle, which has the shape of a sedan, has dimensions of 4.7 meters in length, 1.8 meters in width, and 1.4 meters in height. The truck has dimensions of 8.2 meters in length, 2.5 meters in width, and 3.5 meters in height. The bicycle has dimensions of 1.7 meters in length, 0.45 meters in width, and 1.7 meters

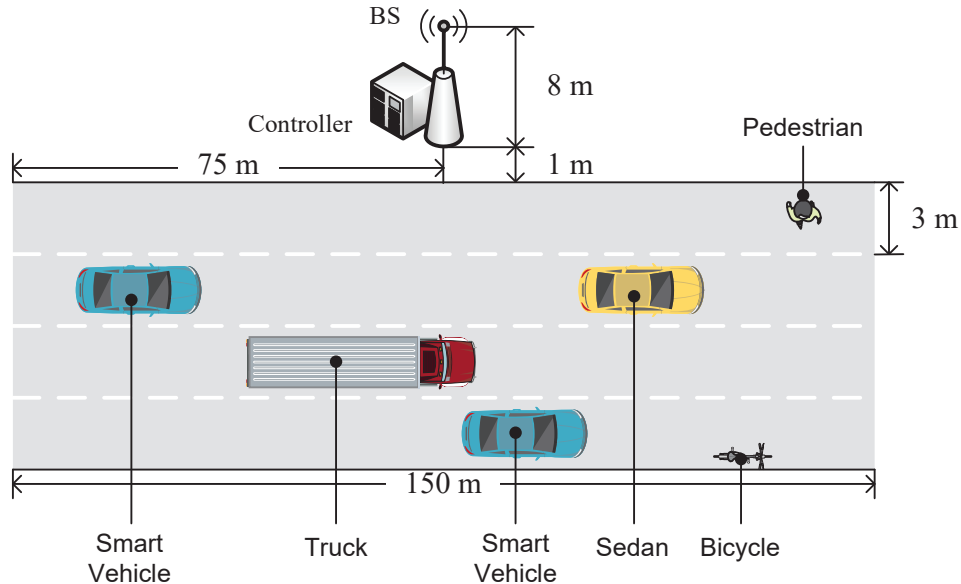


Figure 5.1: Simulation environment.

in height. The walker has dimensions of 0.24 meters in length, 0.45 meters in width, and 1.7 meters in height. Assume that all smart vehicles are traveling at a speed of 60km/h. The average reaction time of the driver is $t_c = 3$ ms. The length of the ROI is $\ell_n = 50$ meters.

A 32-line LiDAR is deployed at the center of the roof of the smart vehicle with an elevation resolution of $b = 0.4^\circ$ and an azimuth resolution of $a = 0.1^\circ$ [28]. The vertical sensing range of the LiDAR is $[-25^\circ, 15^\circ]$. The radius d_s of its sensing dead zone is 5 meters, and its sensing period is 50 ms. The sensing data generated by the LiDAR sensor is stored in a point cloud, where each point is represented by seven 64bit double-precision numbers. Therefore, each point has a data volume of $\epsilon = 448$ bit [53].

In this study, we assume that each smart vehicle utilizes the same classification neural

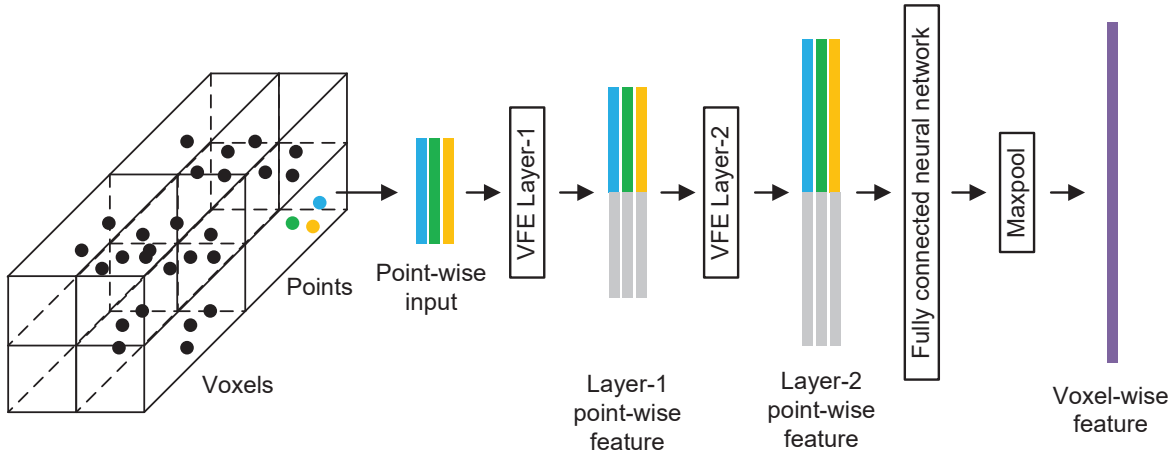


Figure 5.2: The structure of the feature learning network [1].

network to process the point cloud data. This network consists of two main steps: the point cloud fusion step and the object classification step.

In the point cloud fusion step, the point clouds collected by selected collectors are first fused and then processed by the feature learning network (FLN) to extract the features. The structure of the FLN is illustrated in Figure 5.2 [1]. The FLN begins by evenly dividing the bounding box into $128 \times 128 \times 128$ small cubes, known as voxels, with each voxel representing a specific area within the bounding box. The FLN then sequentially extracts the features of each point within a voxel by using two voxel feature encoding (VFE) layers. These features are subsequently combined by using a fully connected layer followed by a max pooling layer, where the output of the max pooling layer represents the overall feature of the voxel.

The two VFE layers share a similar structure, which is illustrated in Figure 5.3. Initially, the features of each point are extracted by using a fully connected neural network, which consists of a linear layer, a batch normalization (BN) layer, and a rectified linear unit

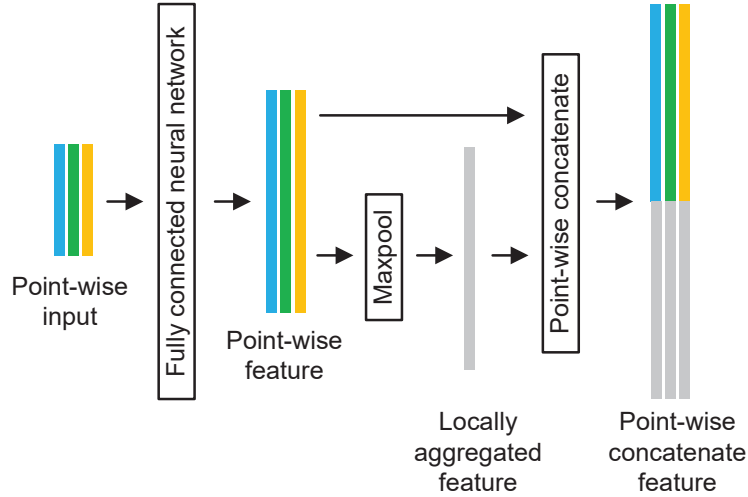


Figure 5.3: The structure of the voxel feature encoding layer [1].

(ReLU) layer. Following this, a max pooling layer is applied to aggregate the features of all individual points within the voxel and form a local voxel feature. Finally, the features of each point are concatenated with the local voxel feature to form the final point-wise feature representation. In the FLN, the input of the first VFE layer includes the location and reflectivity information of each point, with dimensions of 7. The output of the first VFE layer, which serves as the input to the second VFE layer, has dimensions of 32. The second VFE layer then outputs a feature vector for each point with dimensions of 128. Consequently, the FLN produces a 128-dimensional feature representation for each of the $128 \times 128 \times 128$ voxels within the object.

Considering that the computing resources required for point cloud concatenation and voxel feature combination are negligible, the floating point operations (FLOPs) for these processes can be ignored. Therefore, the computing cost of the FLN for processing a single point is primarily determined by the VFE layers, which is approximately 8,640 FLOPs per

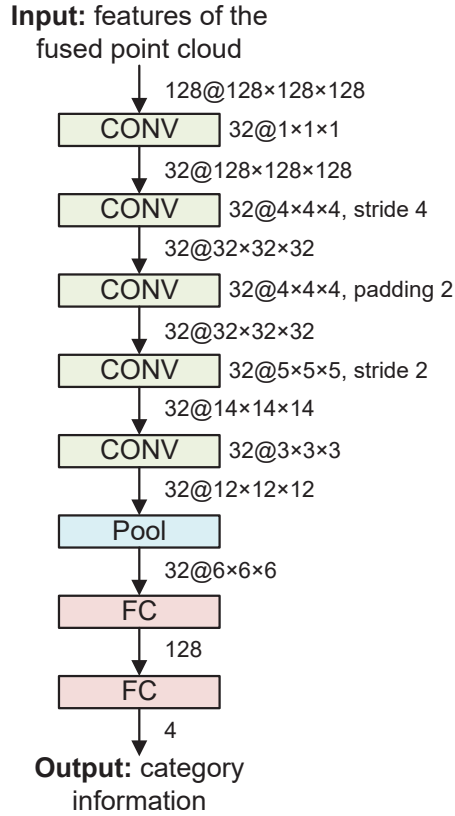


Figure 5.4: The structure of the Voxnet [2].

point. The overall computing cost for processing the fused point cloud of an object is the sum of the computing cost for all points within the point cloud.

In the object classification step, the Voxnet proposed in [2] is used to determine the category of the object. The structure of the Voxnet is illustrated in Figure 5.4. The input of the Voxnet is the 128-dimensional feature map generated by the FLN, which represents the features of $128 \times 128 \times 128$ voxels of the object. The Voxnet first processes this feature map through a series of convolutional layers to extract and compress the features, then applies pooling and fully connected layers to estimate the probabilities of the object belonging to

each category. In each convolutional layer, we use the leaky ReLU as the activation function with a slope of 0.1. Based on the number of additions and multiplications required in each layer, the overall computing cost for the Voxnet to process the feature map of an object is estimated to be approximately 295,150,848 FLOPs.

The training of the classification neural network’s parameters is performed by using stochastic gradient descent (SGD) with momentum. The objective is set as described in [2], which is the sum of multinomial negative log-likelihood and 0.001 times the L_2 weight norm for regularization. The learning rate of the SGD is 0.01. The momentum is 0.9 and the batch size is 32. The training data is generated by using the MATLAB Automated Driving Toolbox [60]. We consider four types of road objects: sedans, trucks, bicycles, and pedestrians. For each type of object, we place it at the center of a 5-lane, 200-meter road segment and generate 480 point clouds with different viewpoints at different locations. By fusing the point clouds from different locations, we produce 800 point clouds for each object type. The classification neural network is trained by using both single-sensor point clouds and fused point clouds. The overall number of point clouds in the training data set is 5120.

Assume that each smart vehicle is equipped with a Jetson AGX Orin computing unit with an available computing resources of $f_k = 2 \times 10^9$ cycles/sec, which is equivalent to a CPU frequency of 2 GHz. The CPU of this unit is based on the Cortex-A78 architecture, which can process 8 FLOPs per CPU cycle [17]. Therefore, the computing resource consumed by the classification neural network to process a point in the fusion step is $c = 1080$ cycles and the computing resource consumed to process a point cloud in the classification step is $c_0 = 36,893,856$ cycles.

Table 5.1: System parameters in simulation

Parameters	Value
Road section length	150 m
ROI length (ℓ_n)	50 m
Bandwidth (B)	20 MHz
Available computing resources of smart vehicle (f_k)	2×10^9 cycles/sec
Classification accuracy requirement (μ_0)	0.8
Service delay requirement (τ)	50 ms
Resource weighting factor w	0.3

Assume that each smart vehicle has the same transmit power, $p_j = 23$ dBm. The noise power of the receiver is $\sigma^2 = -94$ dBm. The reference distance is $d_0 = 1$ m. The channel gain per unit distance is $K = \frac{0.33}{4\pi}$. The path loss exponent is $\gamma = 3$. The key parameters of the system are given in the Table 5.1.

5.2 Simulation Results

5.2.1 Performance under Moderate Traffic Conditions

In this section, we consider a moderate traffic condition as defined in [32], where 9 traffic participants are evenly distributed across 4 lanes. We assume that the traffic participants in each lane are located on the centerline of the road. Let the penetration rate of the smart vehicle be 50%, which means that there are 5 smart vehicles and 4 objects on the road. The available computing resources of the controller is 1×10^{10} cycles/sec, which are equal to the sum of the available computing resources of all the smart vehicles. To evaluate the performance of the proposed method, we randomly generated 5,000 driving

scenarios. In each scenario, service provisioning was completed by using the proposed method, Autocast [27], and EMP [31], respectively. In Autocast, the controller requires all smart vehicles to provide point cloud data for the object classification task. Conversely, in EMP, the controller selects the smart vehicle closest to the object to provide the point cloud data. After receiving the point cloud data, the controller checks whether the classification accuracy of each object meets the service requirement. For tasks that satisfy the accuracy requirement, resources are allocated according to the specific resource allocation method proposed in each scheme. The performance of the proposed method is then compared with that of Autocast and EMP in terms of task completion rate, overall resource consumption, communication resource consumption, and computing resource consumption.

We define a completed task as a task that satisfies both the classification accuracy requirement and the service delay requirement. We define the task completion rate as the ratio of the number of completed tasks to the number of all tasks in the service region. Figure 5.5 shows the average task completion rate for different number of tasks, where the horizontal axis represents the number of tasks in the service region, and the vertical axis represents the average task completion rate of different methods. It can be seen that both the proposed method and Autocast have a high task completion rate, while EMP has a lower completion rate. This is because in the proposed method and Autocast, each task can select multiple collectors to provide point cloud data. The point cloud data collected from different viewing angles and different distances can compensate each other and form a comprehensive description of the object, which greatly increases the classification accuracy of the object. In contrast, the EMP requires each task to select the smart vehicle closest to the object to provide point cloud data. In most cases, the distance between objects

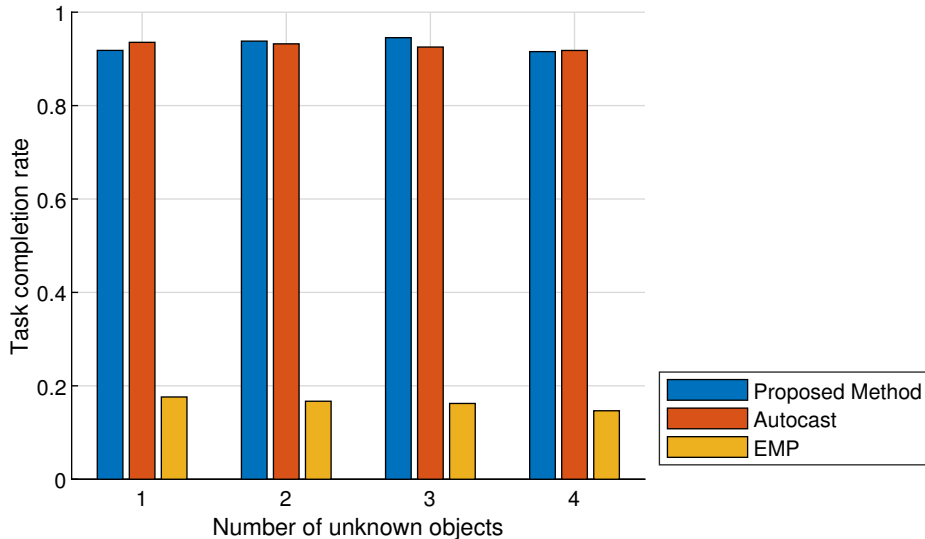


Figure 5.5: Average task completion rate for moderate traffic conditions.

and vehicles is very large and the view of the LiDAR may be obstructed by surrounding objects, leading to the low resolution and low coverage of the point cloud. Therefore, the point cloud data provided by a single vehicle can hardly meet the classification accuracy requirement of the service.

Figure 5.6 shows the overall resource consumption of three methods under different numbers of tasks, where the overall resource consumption is defined as the weighted sum of consumed communication and computing resources as specified in (4.1). Considering that the magnitudes of communication resource consumption, measured in MHz, and computational resource consumption, measured in Giga cycles per second, are comparable, we adopt these measurements as their respective units. We set the weighting factor w to 0.3, indicating a greater concern for computing resource consumption.

As can be seen from Figure 5.6, the resources consumed by all three methods gradu-

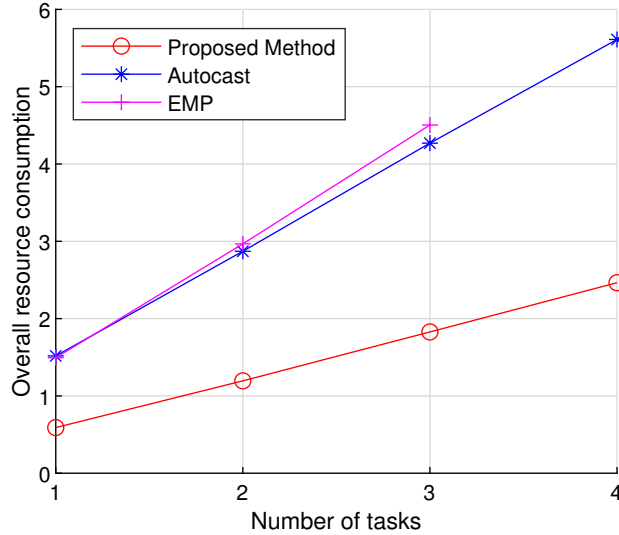


Figure 5.6: Overall resource consumption versus number of tasks.

ally increase as the number of tasks increases. The resource consumption of the method proposed in our thesis is much lower than that of Autocast and EMP. This is because the proposed method selects only a subset of all smart vehicles as collectors. In contrast, Autocast requires all smart vehicles to provide point clouds for object classification, resulting in higher resource consumption. Although EMP selects only one smart vehicle for each task, its resource consumption is close to that of Autocast. This is because the collectors selected by EMP are smart vehicles that are very close to the objects. Due to the effect of geometric perspective, the amount of point cloud data collected by each collector is very large, resulting in large resource consumption. Although Autocast uses the point clouds of all smart vehicles, in most cases, the smart vehicles are far away from the objects, resulting in the total amount of data in the point clouds being less than the amount of data in EMP. Therefore, the resource consumption of EMP is higher than that of Autocast.

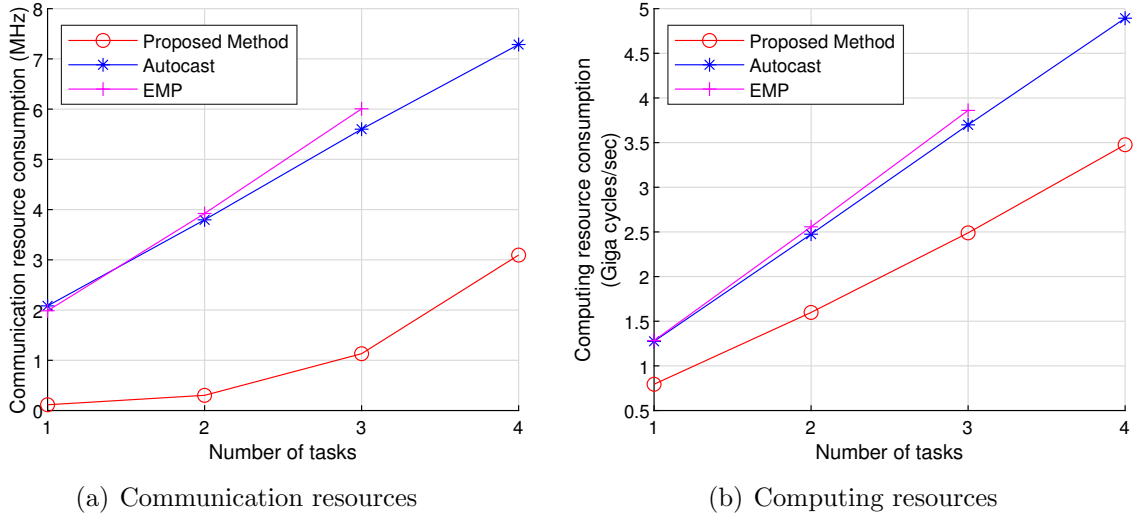


Figure 5.7: Communication and computing resource consumption versus number of tasks.

In addition, if there are four unknown objects in the service region, EMP cannot fulfill all the tasks. By observing the experiment data, we found that at most three unknown objects can have collectors that meet the accuracy requirement. This is because, in most cases, the distances between smart vehicles and objects are very large. The point clouds collected by smart vehicles have low resolution and limited coverage area, which makes it difficult for the selected collector to meet the classification accuracy requirement.

The figures 5.7(a) and 5.7(b) show the communication resource consumption and computing resource consumption of the three methods for different numbers of tasks, respectively. As the number of tasks increases, the resources consumption of all three methods gradually increases. The communication and computing resource consumption of our proposed method is much lower than that of Autocast and EMP. Since the computing and communication resource consumption of all three methods is less than the available resources of the system, the communication and computing resources are not the perfor-

mance bottlenecks under the current traffic condition. The main factor affecting the task completion rate is the sensing capability of each smart vehicle.

5.2.2 Performance under Different Traffic Conditions

In this section, we simulate the proposed method under different traffic conditions. We consider three types of traffic conditions: low levels of congestion condition, moderate traffic condition, and heavy congestion condition, which belong to level of service B, C, and D as defined in [32]. In the low levels of congestion condition, we consider that there are 6 traffic participants. In the moderate traffic condition, there are 9 traffic participants. In the heavy congestion condition, we consider the scenarios of 12 traffic participants and 15 traffic participants, respectively. We assume that the penetration rate of smart vehicles is 50%, i.e., half of the traffic participants are smart vehicles. In each scenario, the service provisioning was completed by using the proposed method, Autocast, and EMP, respectively. Considering that the Autocast and EMP process the point cloud at the controller, for a fair comparison between these three methods, we dynamically adjust the available computing resources of the controller to the sum of the available computing resources of all smart vehicles in the system under different traffic conditions.

We first define the completion rate of a request as the ratio of the number of completed tasks belonging to the front blind spots of the requester to the number of all tasks generated from the front blind spots. Figure 5.8 shows the average request completion rate for different numbers of traffic participants. It can be seen that the proposed method and Autocast have a higher request completion rate. The request completion rate of EMP is much lower and it decreases as the number of traffic participants increases. By observing

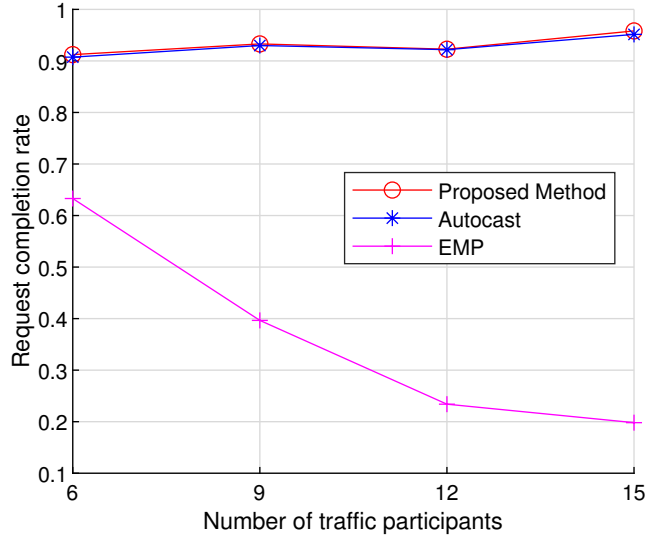


Figure 5.8: Average request completion rate versus number of traffic participants.

the experiment data, we found that the increase of the number of traffic participants will intensify the view obstruction of smart vehicles. Since EMP selects a single smart vehicle that is closest to the object to provide the point cloud, it is very difficult for EMP to meet the accuracy requirements of all tasks. In addition, the request completion rate of our proposed method is slightly higher than that of Autocast. This is because the performance of Autocast is limited by the available computing resources. In some driving environments, the computing resources required by the method exceed the system available resources. As a result, the tasks cannot be completed within the delay requirement. However, the method proposed in our thesis consumes less computing resources than Autocast. Its request completion rate is higher than that of Autocast.

Figures 5.9(a) and 5.9(b) show the average communication resource consumption and computing resource consumption for each task under different numbers of traffic partici-

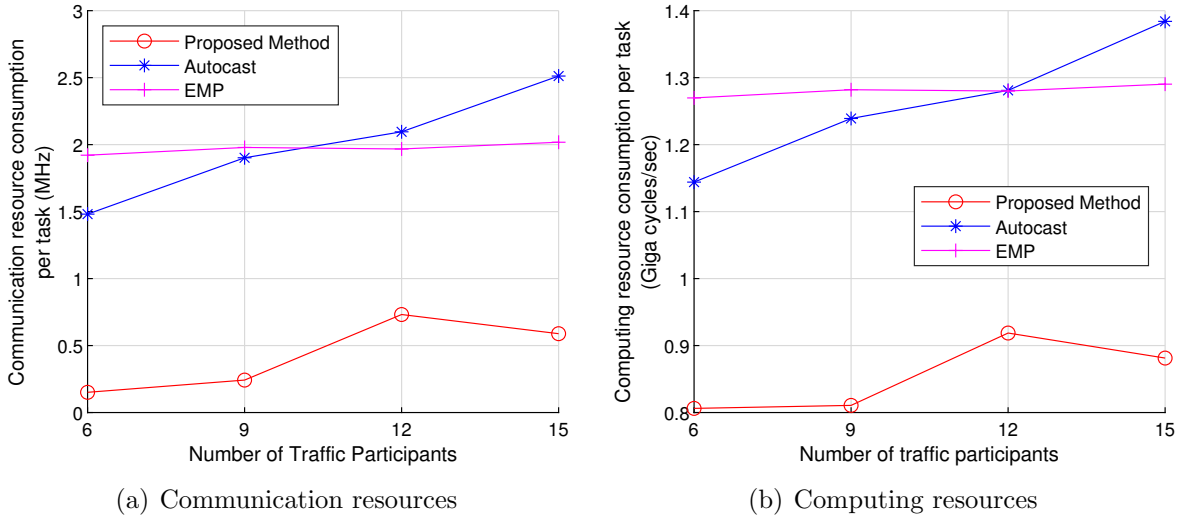


Figure 5.9: Resource consumption per task versus number of traffic participants.

pants, respectively. It can be seen that the resource consumption of the proposed method first increases and then gradually decreases. However, it is consistently smaller than that of Autocast and EMP. This is because the view obstruction of the smart vehicle increases with the number of traffic participants. To achieve the classification accuracy requirement, each task should select more collectors to provide the point cloud data. When the number of traffic participants is larger than 12, the distance between the smart vehicle and the object becomes smaller, which increases the resolution of the collected point cloud. At this point, we can select fewer collectors for fulfilling the accuracy requirement and the overall resource consumption can be reduced. On the contrary, EMP's resource consumption is smoother. This is because EMP selects the smart vehicle closest to the object as the collector. To meet the accuracy requirement, the collectors selected by EMP are always within a certain range from the object. Therefore, the variation in data size of the point cloud collected by each collector is small, resulting in more stable resource consumption. Since

Autocast requires all smart vehicles to provide point cloud data, its resource consumption increases as the number of smart vehicles increases.

5.3 Summary

In this chapter, the performance of the proposed cooperative sensor data collection and processing scheme is numerically evaluated and compared with benchmark schemes. We first evaluate the task completion rate and resource consumption under moderate traffic conditions. Then, we evaluate the request completion rate and resource consumption under three types of traffic conditions, including low levels of congestion condition, moderate traffic condition and heavy congestion condition. The simulation results show that the proposed scheme outperforms the benchmark schemes in terms of task completion rate, request completion rate and resource consumption.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this study, we investigate a resource management problem in the provisioning of vehicular see-through services. To meet the accuracy and delay requirements of the service while improving resource efficiency, we propose a cooperative sensor data collection and processing scheme. In this scheme, we categorize smart vehicles into three roles: requesters with service demands, collectors that provides point cloud data, and analyzers that process point cloud data. To ensure that the collector set selected for each object achieves the required classification accuracy, we propose a novel model for characterizing the relationship between collector selection and object classification accuracy. Based on this model, we propose a collector set pre-selection algorithm that identifies all collector sets for each task that satisfy the classification accuracy and have minimal data redundancy. To address the tradeoff between service requirements and the overall resource consumption of the system,

we treat each potential collector set and analyzer pair as a single unit and assign selection priorities based on their resource consumption per unit of time. The original complex problem is thus transformed into a simpler worker pair selection problem with constraints on communication resources, computing resources, and service delay. To solve this problem, we iteratively adjust the selection priority of each worker pair by using an ACO algorithm. This scheme allows us to ultimately derive a solution that satisfies the service requirements while minimizing the overall resource consumption. Compared to the benchmark service provisioning methods, the proposed scheme demonstrates higher resource efficiency and better request completion rate under various traffic conditions.

6.2 Future Work

Although the proposed cooperative sensor data collection and processing scheme has improved the resource efficiency and service completion rate of vehicular see-through service, there are still many open issues to extend the research in the following aspects:

- In the proposed service provisioning scheme, the selection of collectors and analyzers as well as the allocation of resources heavily relies on a controller deployed at the BS with global information of the system. However, the BSs equipped with such controllers are not widely deployed, particularly in remote areas where full BS coverage has not yet been achieved. Therefore, it is necessary to propose a distributed scheme in which smart vehicles can autonomously select collectors and analyzers and allocate resources for point cloud transmission and processing. However, the design of a distributed decision-making scheme is a challenging issue. This is because the

distributed decision-making system requires extensive state information exchange among smart vehicles, which will consume a large number of communication resources. Recently, multi-agent reinforcement learning has attracted much attention for its ability to solve complex problems while minimizing the data exchange among agents. In future research, we plan to utilize this method to help smart vehicles make vehicle selection and resource allocation decisions.

- In this study, we consider service provision within a service region covered by a BS. When a smart vehicle is about to leave the current service region, its ROI will extend to the next service region. To obtain the complete environment information of the ROI, large information exchange between service regions is required. To reduce the communication costs, a cross-regional service provisioning scheme needs to be proposed. Based on the vehicles' ROI and sensing ranges, smart vehicles can spontaneously divide into several groups. Within each group, the smart vehicles cooperate with each other to obtain environment information about the front blind spots. However, it is challenging to effectively group these vehicles. On one hand, larger groups mean that more smart vehicles are available to be selected as collectors, which can improve the classification accuracy. On the other hand, larger groups require smart vehicles to exchange more information to facilitate vehicle selection and resource allocation decisions. Therefore, a vehicle grouping method should be proposed to address the tradeoff between classification accuracy and communication resource consumption.

References

- [1] Y. Zhou and O. Tuzel, “Voxelnet: End-to-end learning for point cloud based 3D object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4490–4499, 2018.
- [2] D. Maturana and S. Scherer, “Voxnet: A 3D convolutional neural network for real-time object recognition,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 922–928, 2015.
- [3] G. A. Association, “C-V2X use cases, methodology, examples and service level requirements,” tech. rep., 5G Automotive Association, 2019.
- [4] R. L. Thompson, Z. Hu, J. Cho, J. Stovall, and M. Sartipi, “Enhancing driver awareness using see-through technology,” *SAE Technical Paper Series*, vol. 1, 2018.
- [5] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, “Voxel R-CNN: Towards high performance voxel-based 3D object detection,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 1201–1209, 2021.
- [6] L. Chen, Y. Li, C. Huang, B. Li, Y. Xing, D. Tian, L. Li, Z. Hu, X. Na, Z. Li, S. Teng, C. Lv, J. Wang, D. Cao, N. Zheng, and F.-Y. Wang, “Milestones in autonomous

- driving and intelligent vehicles: Survey of surveys,” *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1046–1056, 2023.
- [7] R. Hussain and S. Zeadally, “Autonomous cars: Research results, issues, and future challenges,” *IEEE Communications Surveys and Tutorials*, vol. 21, no. 2, pp. 1275–1313, 2019.
- [8] J. Wang, J. Liu, and N. Kato, “Networking and communications in autonomous driving: A survey,” *IEEE Communications Surveys and Tutorials*, vol. 21, no. 2, pp. 1243–1274, 2019.
- [9] Z. Xiao, J. Shu, H. Jiang, G. Min, H. Chen, and Z. Han, “Perception task offloading with collaborative computation for autonomous driving,” *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 2, pp. 457–473, 2023.
- [10] D. Wang, W. Fu, Q. Song, and J. Zhou, “Potential risk assessment for safe driving of autonomous vehicles under occluded vision,” *Scientific Reports*, vol. 12, no. 1, p. 4981, 2022.
- [11] 5GAA, “C-V2X use cases and service level requirements volume I,” tech. rep., 5G Automotive Association, 2020.
- [12] Y. Siriwardhana, P. Porambage, M. Liyanage, and M. Ylianttila, “A survey on mobile augmented reality with 5g mobile edge computing: Architectures, applications, and technical aspects,” *IEEE Communications Surveys and Tutorials*, vol. 23, no. 2, pp. 1160–1192, 2021.

- [13] S.-W. Kim, W. Liu, M. H. Ang, E. Frazzoli, and D. Rus, “The impact of cooperative perception on decision making and planning of autonomous vehicles,” *IEEE Intelligent Transportation Systems Magazine*, vol. 7, no. 3, pp. 39–50, 2015.
- [14] Y. Yu, X. Tang, J. Wu, B. Kim, T. Song, and Z. Han, “Multi-leader-follower game for MEC-assisted fusion-based vehicle on-road analysis,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11200–11212, 2019.
- [15] L. Du, X. Ye, X. Tan, E. Johns, B. Chen, E. Ding, X. Xue, and J. Feng, “AGO-Net: Association-guided 3d point cloud object detection network,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 8097–8109, 2022.
- [16] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, “Pointpillars: Fast encoders for object detection from point clouds,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12697–12705, 2019.
- [17] L. S. Karumbunathan, *NVIDIA Jetson AGX Orin Series Technical Brief*. Santa Clara, CA: NVIDIA Corporation, 2022.
- [18] M. K. Abdel-Aziz, C. Perfecto, S. Samarakoon, M. Bennis, and W. Saad, “Vehicular cooperative perception through action branching and federated reinforcement learning,” *IEEE Transactions on Communications*, vol. 70, no. 2, pp. 891–903, 2022.
- [19] Y. Jia, R. Mao, Y. Sun, S. Zhou, and Z. Niu, “Online V2X scheduling for raw-level cooperative perception,” in *ICC 2022 - IEEE International Conference on Communications*, pp. 309–314, 2022.

- [20] T. Shehzadi, K. A. Hashmi, D. Stricker, and M. Z. Afzal, “Sparse semi-DERT: Sparse learnable queries for semi-supervised object detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5840–5850, 2024.
- [21] A. Gambashidze, A. Dadukin, M. Golyadkin, M. Razzhivina, and I. Makarov, “Weak-to-strong 3D object detection with x-ray distillation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15055–15064, 2024.
- [22] Z. Yang, Z. Yu, C. Choy, R. Wang, A. Anandkumar, and J. M. Alvarez, “Improving distant 3D object detection using 2D box supervision,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14853–14863, 2024.
- [23] Q. Chen, S. Tang, Q. Yang, and S. Fu, “Cooper: Cooperative perception for connected autonomous vehicles based on 3D point clouds,” in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 514–524, 2019.
- [24] Q. Chen, X. Ma, S. Tang, J. Guo, Q. Yang, and S. Fu, “F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds,” in *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing, SEC ’19*, (New York, NY, USA), pp. 88–100, Association for Computing Machinery, 2019.
- [25] A. Caillot, S. Ouerghi, P. Vasseur, R. Boutteau, and Y. Dupuis, “Survey on cooperative perception in an automotive context,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14204–14223, 2022.

- [26] R. Xu, H. Xiang, X. Xia, X. Han, J. Li, and J. Ma, “OPV2V: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication,” in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 2583–2589, IEEE Press, 2022.
- [27] H. Qiu, P.-H. Huang, N. Asavisanu, X. Liu, K. Psounis, and R. Govindan, “Autocast: Scalable infrastructure-less cooperative perception for distributed collaborative driving,” in *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services, MobiSys ’22*, (New York, NY, USA), pp. 128–141, Association for Computing Machinery, 2022.
- [28] Velodyne, *ULTRA Puck VLP-32C Datasheet*. San Jose, CA: Velodyne, 2018.
- [29] X. Zhang, Z. He, Y. Sun, S. Yuan, and M. Peng, “Joint sensing, communication, and computation resource allocation for cooperative perception in fog-based vehicular networks,” in *2021 13th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp. 1–6, 2021.
- [30] H. Qiu, F. Ahmad, F. Bai, M. Gruteser, and R. Govindan, “AVR: Augmented vehicular reality,” in *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys ’18*, (New York, NY, USA), pp. 81–95, Association for Computing Machinery, 2018.
- [31] X. Zhang, A. Zhang, J. Sun, X. Zhu, Y. E. Guo, F. Qian, and Z. M. Mao, “EMP: edge-assisted multi-vehicle perception,” in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking, MobiCom ’21*, (New York, NY, USA), pp. 545–558, Association for Computing Machinery, 2021.

- [32] T. R. Board, “Highway capacity manual,” *Washington, DC*, 2000.
- [33] R. Xu, H. Xiang, Z. Tu, X. Xia, M.-H. Yang, and J. Ma, “V2X-VIT: Vehicle-to-everything cooperative perception with vision transformer,” in *European conference on computer vision*, pp. 107–124, Springer, 2022.
- [34] T.-H. Wang, S. Manivasagam, M. Liang, B. Yang, W. Zeng, and R. Urtasun, “V2VNet: Vehicle-to-vehicle communication for joint perception and prediction,” in *Computer Vision - ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II*, (Berlin, Heidelberg), pp. 605–621, Springer-Verlag, 2020.
- [35] Z. Yu, F. Yang, J. Teng, A. C. Champion, and D. Xuan, “Local face-view barrier coverage in camera sensor networks,” in *2015 IEEE Conference on Computer Communications (INFOCOM)*, pp. 684–692, 2015.
- [36] Y. Wu, Y. Wang, W. Hu, and G. Cao, “Smartphoto: A resource-aware crowdsourcing approach for image sensing with smartphones,” *IEEE Transactions on Mobile Computing*, vol. 15, no. 5, pp. 1249–1263, 2016.
- [37] Y. Sun, X. Guo, J. Song, S. Zhou, Z. Jiang, X. Liu, and Z. Niu, “Adaptive learning-based task offloading for vehicular edge computing systems,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3061–3074, 2019.
- [38] X. Ye, K. Qu, W. Zhuang, and X. Shen, “Accuracy-aware cooperative sensing and computing for connected autonomous vehicles,” *IEEE Transactions on Mobile Computing*, vol. 23, no. 8, pp. 8193–8207, 2024.

- [39] E. E. Marvasti, A. Raftari, A. E. Marvasti, Y. P. Fallah, R. Guo, and H. Lu, “Co-operative lidar object detection via feature sharing in deep networks,” in *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*, pp. 1–7, 2020.
- [40] Y. Li, S. Ren, P. Wu, S. Chen, C. Feng, and W. Zhang, “Learning distilled collaboration graph for multi-agent perception,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 29541–29552, 2021.
- [41] J. Guo, D. Carrillo, Q. Chen, Q. Yang, S. Fu, H. Lu, and R. Guo, “Slim-FCP: Lightweight-feature-based cooperative perception for connected automated vehicles,” *IEEE Internet of Things Journal*, vol. 9, no. 17, pp. 15630–15638, 2022.
- [42] E. Arnold, M. Dianati, R. de Temple, and S. Fallah, “Cooperative perception for 3D object detection in driving scenarios using infrastructure sensors,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 3, pp. 1852–1864, 2022.
- [43] H. Xiong, D. Zhang, G. Chen, L. Wang, V. Gauthier, and L. E. Barnes, “iCrowd: Near-optimal task allocation for piggyback crowdsensing,” *IEEE Transactions on Mobile Computing*, vol. 15, no. 8, pp. 2010–2022, 2016.
- [44] J. Wang, Y. Wang, D. Zhang, F. Wang, H. Xiong, C. Chen, Q. Lv, and Z. Qiu, “Multi-task allocation in mobile crowd sensing with individual task quality assurance,” *IEEE Transactions on Mobile Computing*, vol. 17, no. 9, pp. 2101–2113, 2018.
- [45] M. Zhang, P. Yang, C. Tian, S. Tang, X. Gao, B. Wang, and F. Xiao, “Quality-aware sensing coverage in budget-constrained mobile crowdsensing networks,” *IEEE Transactions on Vehicular Technology*, vol. 65, no. 9, pp. 7698–7707, 2016.

- [46] Y. Wang and G. Cao, “Barrier coverage in camera sensor networks,” in *Proceedings of the Twelfth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc ’11, (New York, NY, USA), Association for Computing Machinery, 2011.
- [47] Y. Zhou, D. Wang, X. Xie, Y. Ren, G. Li, Y. Deng, and Z. Wang, “A fast and accurate segmentation method for ordered lidar point cloud of large-scale scenes,” *IEEE Geoscience and Remote Sensing Letters*, vol. 11, no. 11, pp. 1981–1985, 2014.
- [48] X. Zhang, W. Xu, C. Dong, and J. M. Dolan, “Efficient l-shape fitting for vehicle detection using laser scanners,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 54–59, 2017.
- [49] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, “On calibration of modern neural networks,” in *International conference on machine learning*, pp. 1321–1330, PMLR, 2017.
- [50] S. Teerapittayanon, B. McDanel, and H. Kung, “Distributed deep neural networks over the cloud, the edge and end devices,” in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pp. 328–339, 2017.
- [51] A. Wehr and U. Lohr, “Airborne laser scanning-an introduction and overview,” *ISPRS Journal of photogrammetry and remote sensing*, vol. 54, no. 2-3, pp. 68–82, 1999.
- [52] Velodyne, *HDL-64e Data Sheet Rev-B*. Morgan Hill, CA: Velodyne Acoustics, 2014.

- [53] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [54] E. P. Baltsavias, “Airborne laser scanning: basic relations and formulas,” *ISPRS Journal of photogrammetry and remote sensing*, vol. 54, no. 2-3, pp. 199–214, 1999.
- [55] 3GPP, “Technical specification group radio access network; NR; study on NR vehicle-to-everything (V2X) (release 16),” report, 3rd Generation Partnership Project (3GPP), Technical Report TR-38.885 Release 16, 2019.
- [56] V. Erceg, L. Greenstein, S. Tjandra, S. Parkoff, A. Gupta, B. Kulic, A. Julius, and R. Bianchi, “An empirically based path loss model for wireless channels in suburban environments,” *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 7, pp. 1205–1211, 1999.
- [57] A. Goldsmith, *Wireless communications*. Cambridge university press, 2005.
- [58] X. Lin, Y. Wang, Q. Xie, and M. Pedram, “Task scheduling with dynamic voltage and frequency scaling for energy minimization in the mobile cloud computing environment,” *IEEE Transactions on Services Computing*, vol. 8, no. 2, pp. 175–186, 2015.
- [59] Y. Xuehan, “Cooperative sensing and computation for environment perception in autonomous driving with vehicular edge computing,” Master’s thesis, University of Waterloo, 2022.
- [60] MathWorks, “Automated driving toolbox.” <https://www.mathworks.com/products/automated-driving.html>, 2024.

- [61] M. Dorigo, V. Maniezzo, and A. Colorni, “Ant system: optimization by a colony of cooperating agents,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996.
- [62] Z.-g. Ren, Z.-r. Feng, and A.-m. Zhang, “Fusing ant colony optimization with lagrangian relaxation for the multiple-choice multidimensional knapsack problem,” *Information Sciences*, vol. 182, no. 1, pp. 15–29, 2012.
- [63] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, “Applications of second-order cone programming,” *Linear Algebra and its Applications*, vol. 284, no. 1, pp. 193–228, 1998.