Journal of Advanced Transportation

Computer Vision Techniques in Intelligent Transportation Systems

Lead Guest Editor: Ryan Wen Liu
Guest Editors: Zhiguang Cao, Yisheng Lv, and Naixue Xiong

Computer Vision Techniques in Intelligent Transportation Systems

Journal of Advanced Transportation

Computer Vision Techniques in Intelligent Transportation Systems

Lead Guest Editor: Ryan Wen Liu
Guest Editors: Zhiguang Cao, Yisheng Lv, and Naixue Xiong

Contents

Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 5038832, 8 pages
https://doi.org/10.1155/2021/5038832

Research Article

Online Discrete Anchor Graph Hashing for Mobile Person Re-Identification

Liang Xie and Xi Fang

School of Science, Wuhan University of Technology, Wuhan 430070, China

Correspondence should be addressed to Xi Fang; fangxi@whut.edu.cn

Received 30 April 2021; Accepted 5 November 2021; Published 21 December 2021

Academic Editor: Alessandro Severino

With the advance of mobile technologies, mobile devices such as unmanned aerial vehicle (UAV) become more important in video surveillance. By applying mobile person re-identification (re-id), mobile devices can monitor pedestrians in the transportation system from complex environments. Since the computing and storage resources of mobile devices are limited, traditional person re-id methods are not appropriate for mobile condition. Besides, mobile person re-id task also requires real-time processing. In this paper, we propose a novel hashing method: online discrete anchor graph hashing (ODAGH) for mobile person re-id. ODAGH integrates the advantages of online learning and hashing technology. In ODAGH, we propose an online discrete optimization algorithm to improve the efficiency of anchor graph learning in the online scenario. Experimental results demonstrate the superiority of ODAGH in terms of both effect and efficiency.

1. Introduction

With person re-identification (re-id) technology, it will be able to find the same identity from different and non-overlapping cameras. Person re-id can be widely used for video surveillance; moreover, person re-identification is the key technology in pedestrian traffic monitoring [1]. Detecting and tracking a person across camera is important in traffic monitoring system [2]. Also, person re-id technology can intelligently and efficiently identify and track pedestrians in streets, airports, or other transportation systems.

The task of person re-id is an image retrieval problem. Given a probe image (query), the purpose of person re-id is to search the information about established personnel characteristics in a traffic database for images that contain the same person [3]. Traditional person re-id technology is used in the scene where cameras are unable to move, such as fixed camera networks in different public areas, including urban transport systems.

In recent years, with the advance of mobile technologies, mobile systems such as unmanned aerial vehicle (UAV) are widely used for video surveillance and traffic monitoring [2]. For example, UAV with cameras can play more important roles in tracking people in transportation system from complex environments such as rural area, mountain, and sea, where fixed cameras are lacking or cannot reach, and they can accomplish dangerous and boring visual information gathering tasks with great excellence. Therefore, mobile person re-id is required. Unlike traditional person re-id methods, mobile systems have limited computing and storages resources, and in real-word applications, the communication bandwidth is also limited. However, image processing methods usually have high time and space complexity, and traditional person re-id methods are not suitable for mobile systems.

The hashing technique is an efficient image retrieval method for fast person re-id. It converts high-dimensional data into short binary code while keeping the data similar. With the operations including XOR and bit-counting, it will make it easier to do a fast search. Also, the hashing method based on machine learning is proved to be superior to the hashing method based on random projection [4–6]. However, most existing hashing methods use batch learning

strategy, and they learn hash functions offline by using specific training data. If new person images are collected and added to the database, they also use the pre-trained model to obtain new hashing codes. If the information of new images is different to training data, the performance of offline hashing model will be affected significantly. Offline hashing methods must accumulate all the databases to retrain new hash functions and recompute all the hashing codes. They are obviously very inefficient, especially when the database is frequently updated by new collected images.

Existing hashing methods are not appropriate for the application of mobile system which requires fast real-time methods to solve their tasks such as person re-id. To cope with the real-time demand of mobile systems, a mobile person re-id method should have two import characteristics: (1) it should support fast retrieval of images and (2) it should learn hashing functions from changing training data in online manner. As an emerging technology, online hashing technique [7, 8] can be applied to cope with the online retrieval of changing database.

In the paper, online discrete anchor graph hashing (ODAGH) is proposed for mobile person re-id. ODAGH utilized the advantages of graph learning to effectively preserve the visual correlation of person images. However, traditional graph learning still requires much storage and computing resources. In ODAGH, we propose online anchor graph learning which first uses anchor graph to reduce the space cost of graph construction and then uses an online learning algorithm to optimize the graph model effectively and efficiently. The main contributions of this paper are summarized as follows:

(i) ODAGH integrates the advantages of online learning and hashing, and as a hashing method, it can be easily applied to mobile systems with limited computing and storage resources. By leveraging online discrete algorithm, ODAGH can efficiently update hashing functions and learn discrete hashing codes when new images are collected. It can fulfil the real-time demand of mobile systems.

(ii) ODAGH is an unsupervised hashing method, so it requires no training labels which are hard to obtain. By improving the graph learning, the visual correlation can be effectively preserved in hashing codes. Moreover, by using a GPI-based online optimization, quantization loss can be largely avoided in the hashing process.

(iii) ODAGH only relies on one parameter, which guarantees its robustness in real applications. Also, experimental results demonstrate the effectiveness and efficiency of ODAGH compared to other person re-id methods.

The rest of this paper is organized as follows. The related work is reviewed in Section 2. We present the details of our proposed method in Section 3. The experimental configuration and results are introduced in detail in Section 4. Finally, the conclusion of the paper is given in Section 5.

2. Related Work

2.1. Image Hashing. In recent years, image hashing has gained much attention from researchers in image retrieval. Generally, there are two kinds of hashing types: one is supervised hashing [9–11] and the other is unsupervised hashing [6, 12]. By learning the class labels or leveraging other supervised information, the binary hash coding can be studied in a supervised hashing method. Representative supervised hashing methods include column sampling-based discrete supervised hashing (COSDISH) [9], supervised discrete hashing (SDH) [10], deep supervised discrete hashing (DSDH) [13], and fast scalable supervised hashing (FSSH) [14]. However, because of the extremely expensive costs to annotated supervised labels, the supervised hashing methods have limited application range on large-scale image retrieval.

Unsupervised hashing methods have one obvious advantage that they can learn hash functions without any label information. Therefore, the original geometric structure, including visual information, can be retained in hash coding. Representative supervised hashing methods include spectral hashing (SH) [4], iterative quantization (ITQ) [6], angular reconstructive embeddings (AREs) [15], unsupervised hashing with binary deep neural network (UH-BDNN) [16], and similarity-adaptive deep hashing (SADH) [17]. Nevertheless, most unsupervised and supervised hashing methods learn hash functions in offline manner, and they cannot cope with the scenario where new data are continuously added to the database.

2.2. Pedestrian Detection and Tracking

2.2.1. Person Re-Identification. Traditional research on person re-id mainly includes visual feature representation [18, 19] and distance metric learning [20, 21]. In [22], feature effectiveness was identified in a query-adaptive manner for feature fusion. The method proposed in [23] learns discriminative and robust representations via dictionary learning. In [24], Fisher vectors were used for person re-id. Part loss network was proposed in [25] for deep representation learning. There are also many studies related to distance metric learning. In [26], person re-identification problem was formulated as a ranking problem, and Ensemble RankSVM was developed to solve the problem. KISSME [27] considers the scalability and learns a distance metric from equivalence constraints. Mahalanobis metric leaning was also used in several methods [28, 29]. Feature presentation learning can be regarded as the pre-processing step of person re-id, and it can be combined with metric learning or our proposed hashing method.

Recently, several hashing-based person re-id methods [30–33] were proposed to improve the search efficiency. As we know, fast indexing is so essential between the raw image data and the binary hashing codes. In order to realize the end-to-end fast indexing, the convolutional neural network (CNN) is always adopted in deep regularized similarity comparison hashing (DRSCH) [34]. The spatial information

is integrated with part-based deep hashing (PDH) by representing horizontal parts to binary codes for feature discrimination [35]. A novel coarse-to-fine (CtF) method [36] complementarily learns short and long codes in a pyramid structure from CNN. Most of existing hashing-based person re-id methods use deep network for feature representation learning, and they ignore the preservation of visual correlation in hashing codes. Moreover, deep learning [37, 38] methods cost many computing and storage of resources, so they are not appropriate for the mobile person re-id task which is always performed on mobile systems with limited resources.

3. Proposed Method

In this section, we introduce online discrete anchor graph hashing (ODAGH) in detail. The overall framework of ODAGH is shown in Figure 1. If a mobile device (e.g., UAV) collects new images of a person, then visual features are extracted for online discrete anchor graph learning. ODAGH can efficiently update hashing functions by only new features and training variables with small sizes. Then hashing codes of new images are added to database. In the query step, users can use a query image of specific person to search images of this person from database.

3.1. Problem Setting. Suppose the database consists of streaming images. When new images come in, we update the hash functions. We define $\mathbf{X} \in \mathbb{R}^{N\times d}$ as image matrix, where N is the number of all training images in database and d is the dimension of image feature. In the online learning process, image matrix $\mathbf{X}$ can be represented as $[\mathbf{X}_{\text{old}}^T, \mathbf{X}_{\text{new}}^T]^T$, where $\mathbf{X}_{\text{old}} \in \mathbb{R}^{N_{\text{old}}\times d}$ denotes old images in the database and $\mathbf{X}_{\text{old}} \in \mathbb{R}^{N_{\text{new}}\times d}$ denotes new images, $N = N_{\text{new}} + N_{\text{old}}$. Our goal is to learn hashing functions and hashing codes $\mathbf{H} \in \mathbb{R}^{N\times k}$ for all images, where k is the code length. In order to guarantee efficiency, we directly use linear projection to reduce time cost. Thus, the formulation of hashing function is defined as

$$\mathbf{H} = \text{sgn}(\mathbf{XW}), \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{d\times k}$ is the weight matrix. The main notations used in this paper are summarized in Table 1.

3.2. Basic Formulation. We use graph learning [39] to preserve the visual information in hashing codes. However, traditional graph learning costs many computing and storage resources, and it is hard to design an online algorithm for graph learning. Therefore, we use anchor graph learning which constructs anchor graph to approximate a graph.

For constructing the anchor graph, N_a anchors $x_j^a|_{j=1}^{N_a}$ firstly can be randomly selected from the training data, and later we can approximate the data neighbourhood structure. Then, the truncated similarity matrix $\mathbf{Z} \in \mathbb{R}^{N\times N_a}$ can be achieved as

$$Z_{ij} = e^{-\text{dist}\left(x_i - x_j^a\right)/\sigma}, \tag{2}$$

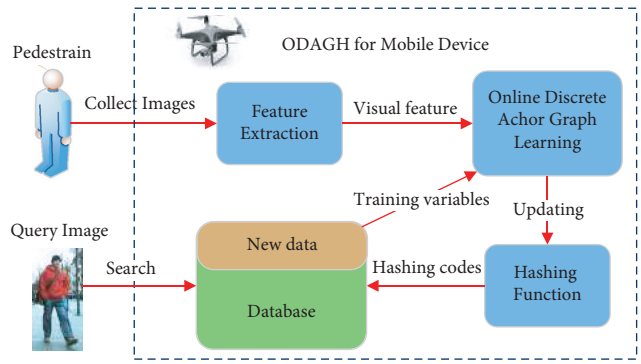


Figure 1: The overall framework of OAGH for mobile person re-id.

Table 1: Summary of main notations.

Notations	Explanations
N	Size of whole database
d	Feature dimension
k	Hashing code length
$\mathbf{X}$	Image feature matrix
$\mathbf{H}$	Hashing code matrix
$\mathbf{Z}$	Truncated similarity matrix
$\mathbf{W}$	Weight matrix for hashing
$\mathbf{X}_{\text{new}}$	Feature matrix of new images
$\mathbf{X}_{\text{old}}$	Feature matrix of old images

where $\text{dist}(x_i - x_j^a)$ is the distance between image x_i and anchor x_j^a (in this paper, we use L2 distance for similarity matrix) and σ is the mean of all distances. In addition, we can approximate the graph matrix of each modality by $\mathbf{A} = \mathbf{ZZ}^T$.

Based on the construction of anchor graph, we can formulate anchor graph learning for hashing as

$$\begin{cases} \min & Tr\left(\mathbf{H}^T\mathbf{LH}\right) \\ \text{s.t.} & \mathbf{H} \in \{0,1\}^{n\times l}. \end{cases} \tag{3}$$

The above formulation is similar to traditional graph learning. The biggest difference is that the Laplacian matrix should be computed as $\mathbf{L} = \mathbf{I} - \mathbf{A}$, and $\mathbf{A}$ is approximated graph matrix. $Tr(\cdot)$ denotes trace operator.

It is unavailable to directly solve the discrete constraint of hashing matrix H in equation (3). Therefore, we relax the hard discrete constraint by introducing a continuous matrix $\mathbf{F} = \mathbf{XW}$ to replace $\mathbf{H}$ in equation (3). We also add the term $\|\mathbf{F} - \mathbf{H}\|_F^2$ to make $\mathbf{H}$ close to $\mathbf{F}$. Then, we can obtain the overall basic formulation as

$$\begin{cases} \min & Tr\left(\mathbf{W}^T\mathbf{X}^T\mathbf{LXW}\right) + \alpha\|\mathbf{XW} - \mathbf{H}\|_F^2 \\ \text{s.t.} & \mathbf{W}^T\mathbf{W} = \mathbf{I}, \mathbf{H} \in \{0,1\}^{n\times l}, \end{cases} \tag{4}$$

where α is the parameter for the second term. The constraint $\mathbf{W}^T\mathbf{W} = \mathbf{I}$ is used to avoid the trivial solution of $\mathbf{W}$.

The objective function (4) can be solved by an alternate iteration method. At each step, we optimize one variable and fix other variables. When we fix $\mathbf{H}$ and optimize $\mathbf{W}$, the objective function can become

$$\begin{cases} \max & Tr\left(\mathbf{W}^T\mathbf{B}\mathbf{W}\right) + 2\alpha Tr\left(\mathbf{W}^T\mathbf{X}^T\mathbf{H}\right) \\ \text{s.t.} & \mathbf{W}^T\mathbf{W} = \mathbf{I}, \end{cases} \tag{5}$$

where

$$\mathbf{B} = \mathbf{X}^T\mathbf{Z}\mathbf{Z}^T\mathbf{X} - \mathbf{X}^T\mathbf{X}. \tag{6}$$

It is obvious that equation (5) is a quadratic problem on the Stiefel manifold [40], and the Lagrangian function of this objective function is

$$\begin{aligned} L(\mathbf{W}, \mathbf{\Lambda}) = {} & Tr\left(\mathbf{W}^T\mathbf{B}\mathbf{W}\right) + 2\alpha Tr\left(\mathbf{W}^T\mathbf{X}^T\mathbf{H}\right) \\ & - Tr\left(\left(\mathbf{W}^T\mathbf{W} - n\mathbf{I}\right)\mathbf{\Lambda}\right). \end{aligned} \tag{7}$$

By setting the derivative of equation (5) w.r.t $\mathbf{W}$ to 0, we have

$$\frac{\partial L(\mathbf{W}, \mathbf{\Lambda})}{\partial \mathbf{W}} = 2\mathbf{B}\mathbf{W} + 2\alpha\mathbf{X}^T\mathbf{H} - 2\mathbf{W}\mathbf{\Lambda}. \tag{8}$$

We can use generalized power iteration (GPI) [40] to solve $\mathbf{W}$, and the detailed algorithm of GPI is shown in Algorithm 1.

Then, we fix $\mathbf{W}$ and optimize $\mathbf{H}$, and (4) becomes

$$\begin{cases} \min & \|\mathbf{X}\mathbf{W} - \mathbf{H}\|_F^2 \\ \text{s.t.} & \mathbf{H} \in \{0, 1\}^{n\times l}. \end{cases} \tag{9}$$

The solution of (4) can be easily obtained by

$$\mathbf{H} = \text{sgn}(\mathbf{X}\mathbf{W}). \tag{10}$$

3.3. Online Algorithm. In this section, we consider the online optimization of (4). In the online learning process, image matrix $\mathbf{X}$ can be represented as $[\mathbf{X}_{\text{old}}^T, \mathbf{X}_{\text{new}}^T]^T$. Similarly, hashing matrix $\mathbf{H}$ can be represented as $[\mathbf{H}_{\text{old}}^T, \mathbf{H}_{\text{new}}^T]^T$, where $\mathbf{H}_{\text{old}}$ denotes the hashing codes of old images and $\mathbf{H}_{\text{new}}$ denotes the hashing codes of new images.

Then, we consider the online improvement of GPI, and we can obtain that

$$\mathbf{C}_{\text{new}} = \mathbf{X}^T\mathbf{X} = \mathbf{C}_{\text{old}} + \mathbf{X}_{\text{new}}^T\mathbf{X}_{\text{new}}, \tag{11}$$

where $\mathbf{C}_{\text{old}} = \mathbf{X}_{\text{old}}^T\mathbf{X}_{\text{old}}$, and we can find that $\mathbf{C}_{\text{old}}$ is only related to old images; it can be preserved in previous learning. So, in the online process, we only need to compute $\mathbf{X}_{\text{new}}^T\mathbf{X}_{\text{new}}$, whose time complexity is $O(N_{\text{new}})$. The time of computing $\mathbf{C}_{\text{new}}$ is linear to the size of new images, and it is irrelevant to size of the database.

The truncated similarity matrix $\mathbf{Z}$ also can be represented as $[\mathbf{Z}_{\text{old}}^T, \mathbf{Z}_{\text{new}}^T]^T$.Therefore, we can obtain that

$$\mathbf{D}_{\text{new}} = \mathbf{Z}^T\mathbf{X} = \mathbf{D}_{\text{old}} + \mathbf{Z}_{\text{new}}^T\mathbf{X}_{\text{new}}, \tag{12}$$

where $\mathbf{D}_{\text{old}} = \mathbf{Z}_{\text{old}}^T\mathbf{X}_{\text{old}}$. Similar to (8), we can find that the time complexity of computing $\mathbf{D}$ is also linear to the size of new images, and it is irrelevant to the size of the whole database.

Based on (11) and (12), we can obtain online updation of $\mathbf{B}_{\text{new}}$ with linear time complexity which is irrelevant to the size of the database:

$$\mathbf{B}_{\text{new}} = \mathbf{D}_{\text{new}}^T\mathbf{D}_{\text{new}} - \mathbf{C}_{\text{new}}. \tag{13}$$

Similarly, when we compute $\mathbf{M}$, its term $\mathbf{X}^T\mathbf{H}$ can be computed in an online manner:

$$\mathbf{E}_{\text{new}} = \mathbf{X}^T\mathbf{H} = \mathbf{E}_{\text{old}} + \mathbf{X}_{\text{new}}^T\mathbf{H}_{\text{new}}, \tag{14}$$

where $\mathbf{E}_{\text{old}} = \mathbf{X}_{\text{old}}^T\mathbf{H}_{\text{old}}$, and the time complexity of computing $\mathbf{E}_{\text{new}}$ is linear to the size of new data.

According to the above division for online updating, we propose an online optimization algorithm based on GPI to solve $\mathbf{W}$ and optimize $\mathbf{H}$. The detailed procedure is summarized in Algorithm 2.

In Algorithm 2, according to above analysis, we can find that the time complexities of steps 1–5 and steps 7–8 are all $O(N_{\text{new}})$, and N_{new} is the size of new images. The time complexity of steps 9 and 11 is $O(d^2k)$, d is the dimension of image features, and k is the code length. Since the size of $\mathbf{M}$ is $d \times k$, performing SVD on $\mathbf{M}$ costs $O(d^2)$. d and k are irrelevant to the new data size N_{new}, and thus they can be ignored in the computation of whole time complexity. As a result, the overall time complexity of Algorithm 2 is $O(N_{\text{new}})$. The time complexities of the steps in Algorithm 1 are all irrelevant to the size of the database. Thus, even when the database is very large, our algorithm is constantly efficient.

3.4. Overall Process. Suppose the initial database has N_I images $\mathbf{X}_I$, and it becomes larger when new images come in. The overall process of our OAGH is described in Algorithm 3.

Suppose that the database size is N currently. We can easily find that the time complexity of Algorithm 1 is $O(N_I)$, and the time complexity of Algorithm 2 is $O(N_{\text{new}})$. Therefore, the time cost of our overall iterative online process is linear to the size of all images. Also, it is equivalent to the time cost of one-round learning of offline hashing.

4. Experiments

4.1. Datasets. We use Market-1501 dataset [41] to evaluate the performance of our method. There are 32,668 bounding boxes of 1,501 identities in Market-1501 dataset. It has the largest person re-id dataset with 14.8 cross-camera ground truths for each query on average. Also, it is closer towards realistic situations than previous ones. Market-1501 dataset can better show the effectiveness of online methods. Since we mainly focus on the efficiency of person image retrieval and do not focus on feature learning, we directly use IDE_ResNet_50 features.

4.2. Experimental Settings. We compared our methods with two non-hashing methods: Euclidean [41] and KISSME [27], and two supervised offline hashing methods: COSDISH [9] and SDH [10]. We set $N_I = 1000$, which means that the

(i) **Input:** $\mathbf{B}, \mathbf{X}, \mathbf{H}$
(ii) **Output:** $\mathbf{W}$
(1) Initialize an orthogonal matrix $\mathbf{W}$;
(2) **for** iter < max_iter **do**
(3) Update $\mathbf{H}$ according to equation (10);
(4) Compute $\mathbf{M} = 2\mathbf{BW} + 2\alpha^{\mathbf{X}}T\mathbf{H}$;
(4) Perform the compact SVD $\mathbf{M} = \mathbf{US}^{\mathbf{V}}T$;
(6) **Compute** $\mathbf{W} = \sqrt{n}\mathbf{U}^{\mathbf{V}}T$;
(7) **end for**

Algorithm 1: Generalized power iteration for solving $\mathbf{W}$ and $\mathbf{H}$.

(i) **Input:** $\mathbf{C}_{\text{old}}, \mathbf{D}_{\text{old}}, \mathbf{E}_{\text{old}}, \mathbf{H}_{\text{old}}, \mathbf{X}_{\text{old}}, \mathbf{X}_{\text{old}}$
(ii) **Output:** $\mathbf{W}, \mathbf{C}_{\text{old}}, \mathbf{D}_{\text{old}}, \mathbf{E}_{\text{old}}, \mathbf{H}$
(1) Compute truncated similarity matrix $\mathbf{Z}_{\text{new}}$;
(2) **Compute** $\mathbf{C}_{\text{new}}$ according to (11);
(3) Compute $\mathbf{D}_{\text{new}}$ according to (12);
(4) **Compute** $\mathbf{B}_{\text{new}}$ according to (13);
(5) **Initial** an orthogonal matrix $\mathbf{W}$;
(6) **for** iter < max_iter **do**
(7) **Compute** $\mathbf{H}_{\text{new}} = \text{sgn}(\mathbf{X}_{\text{new}}\mathbf{W})$;
(8) **Update** $\mathbf{E}_{\text{new}}$ according to (14);
(9) **Compute** $\mathbf{M} = 2\mathbf{B}_{\text{new}}\mathbf{W} + 2\alpha\mathbf{E}_{\text{new}}$;
(10) **Perform** the compact SVD $\mathbf{M} = \mathbf{USV}^T$;
(11) Compute $\mathbf{W} = \sqrt{n}\mathbf{UV}^T$;
(12) end for
(13) **Update** $\mathbf{C}_{\text{old}} = \mathbf{C}_{\text{new}}, \mathbf{D}_{\text{old}} = \mathbf{D}_{\text{new}}, \mathbf{E}_{\text{old}} = \mathbf{E}_{\text{new}}$;
(14) Add **new** hashing codes by $\mathbf{H} = [\mathbf{H}_{\text{old}}^T, \mathbf{H}_{\text{new}}^T]^T$

Algorithm 2: Online optimization algorithm based on GPI.

(i) **Input:** $\mathbf{X}_I, \mathbf{X}_{\text{new}}$
(ii) **Output:** $\mathbf{W}, \mathbf{H}$
(1) Using $\mathbf{X}_I$ as training data, compute $\mathbf{W}_{ol\,d}, \mathbf{H}_{ol\,d}$ according to Algorithm 1;
(2) While new images $\mathbf{X}_{\text{new}}$ come in
(3) Update $\mathbf{W}, \mathbf{C}_{\text{old}}, \mathbf{D}_{\text{old}}, \mathbf{E}_{\text{old}}, \mathbf{H}$ by Algorithm 2;
(5) End While

Algorithm 3: Overall process of ODAGH.

database contains 1000 images at the beginning. Then, we set $N_{\text{new}} = 1000$, which means that at each round, 1000 new images are added to the database. Finally, the database contains 32668 images; at the last round, 668 images are added. Since COSDISH and SDH are offline methods and it is obviously time-consuming to train in every round, to be consistent with real applications, we only train hash functions at first round for COSDISH and SDH. Deep learning-based hashing methods [34–36] usually require many computing and storage resources, which are not appropriate for mobile embedded systems. Therefore, we do not use them for comparison.

Our method only relies on one parameter α. We select a proper value of α from a candidate set $\{0.1, 1, 10, 10^2, 10^3, 10^4, 5 \times 10^4, 10^5\}$, and we finally choose the best $\alpha = 5 \times 10^4$. The hashing code length for all hashing methods is set to 512.

Non-interpolated mean average precision (MAP) score is utilized [42] to evaluate the performance of all compared methods. Given a query, the average precision (AP) is defined as

$$\text{AP} = \frac{1}{p}\sum_{i=1}^{N}\text{pre}(i)\text{rel}(i), \tag{15}$$

where p is the number of relevant images, $\text{pre}(i)$ is the precision of top i retrieved images, and $\text{rel}(i) = 1$ if the image is relevant to $i-th$ query; otherwise, $\text{rel}(i) = 0$. The MAP score is the mean of AP scores from all queries. Besides

Table 2: Comparison of MAP results on Market-1501 dataset.

Method	MAP
Euclidean	0.5422
KISSME	0.5677
COSDISH	0.3829
SDH	0.2989
OAGH	**0.5723**

The best result is highlighted in bold.

MAP, precision-recall (PR) curves are also used to measure the performance of all methods.

4.3. Experimental Results. Table 2 shows the MAP score of all compared methods on Market-1501 dataset. The results of Euclidean and KISSME are reported in [41], and we carefully tune the parameters of COSDISH and SDH to report best results. From Table 2, we can find that OAGH performs best. Although OAGH is an unsupervised method, it can effectively preserve visual correlation of images in hashing codes. The performance of two offline hashing methods is much worse than other methods, even though they are supervised methods. The main reason is that they cannot support online learning of new images, and they only use 1000 initial images for training. The MAP scores obtained by the non-hashing method KISSME are very close to OAGH. The main reason is that hashing will introduce quantization loss. It is reasonable that hashing methods perform even worse than non-hashing methods.

Figure 2 shows the PR curves of all compared methods, and we can further observe the retrieval performance of them. We can obtain similar results as MAP scores. OAGH and two non-hashing methods significantly outperform other two hashing methods. Also, the yellow curve of KISSME is very close to blue curve of OAGH.

Both MAP scores and PR curves demonstrate the superiority of our ODAGH. It can easily outperform offline hashing methods. Also, it can even outperform classical non-hashing methods. The above results prove that the proposed online discrete anchor graph learning process has good applicability and makes the process effective.

At last, we also evaluate the total training time of ODAGH. Table 3 shows the comparison of training time, and the experiment described in Table 3 is conducted on a PC with Core i5 2.11 GHz CPU and 16 GB memory. For offline hashing, we use SDH as representative. When new images come in, offline hashing trains the whole database, while ODAGH only needs to process new data. As a result, in our experiment, ODAGH totally processes 32668 images, and SDH processes 560668 images in fact. Therefore, the training time of ODAGH is significantly less than SDH. If new data continue to come in and the database continues to be enlarged, the superiority of our method in total training time will be much more significant.

4.4. Influence of Code Length. Traditional graph-based hashing methods cannot directly solve the discrete constraint of hashing codes, which will introduce much quantization loss. When code length increases, the quantization loss will also increase, and thus their performance cannot be improved and even deteriorates when the code length increases [43].

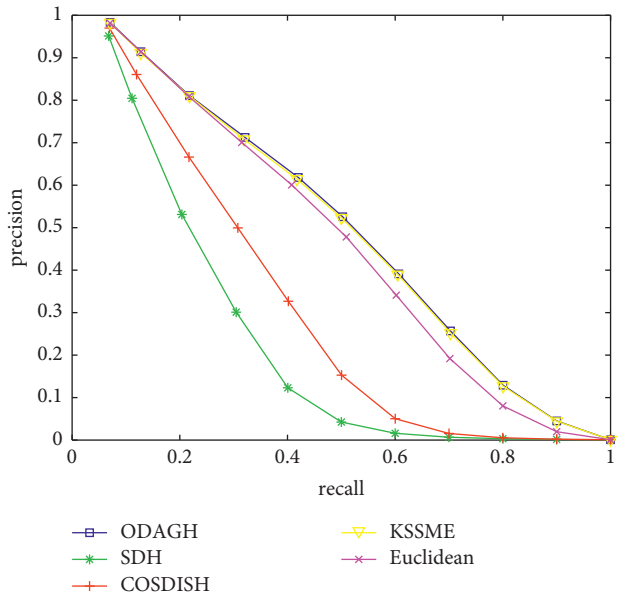


Figure 2: PR curves of all compared methods on Market-1501 dataset.

Table 3: Comparison of total training time.

Method	Time (seconds)
Offline hashing (SDH)	1964.8
ODAGH	**141.4**

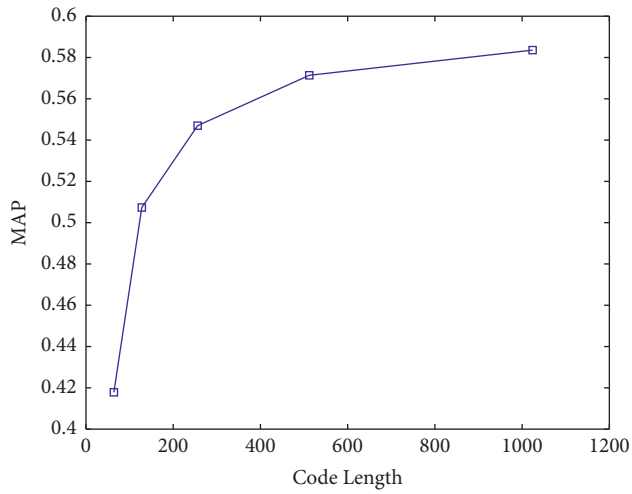


Figure 3: MAP variations with code length of 64, 128, 256, 512, and 1024.

Figure 3 shows the performance of ODAGH with increasing code length, and we can observe that the performance increases constantly with the increase of code length. The main reason is that ODAGH uses the GPI-based online discrete optimization procedure which can effectively avoid the quantization loss. We can also find that when that code

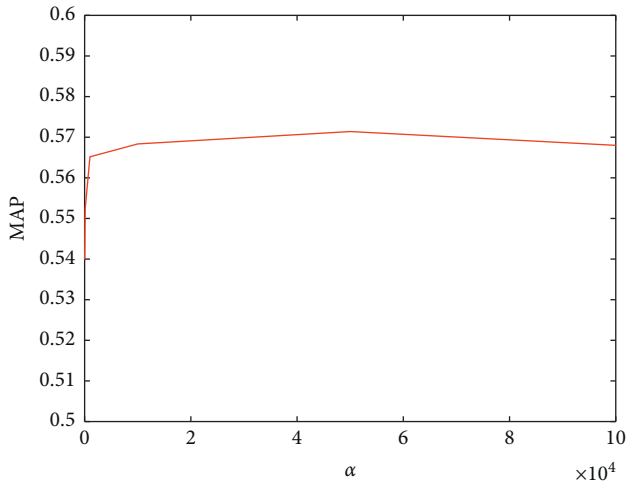


Figure 4: MAP variations with different values of α.

length is larger than 512, the increase of MAP score is not significant, which indicates that 512 bits are sufficient for this person re-id task.

4.5. Parametric Analysis. In this section, we analyse the influence of parameter on our method. Figure 4 shows the MAP scores with different values of α, where α varies from 1 to 10^5. We can observe that the performance of ODAGH is stable with the variation of α, and thus our method is not sensitive to the parameter and is robust in person re-id. Moreover, we can find that MAP score achieves relatively high value when $\alpha > 10^4$, which has the same magnitude with image number. This phenomenon indicates that we can easily set the parameter α according to database size.

5. Conclusions

In this paper, we propose online discrete anchor graph hashing (ODAGH) for mobile person re-id. We first introduce the basic formulation of discrete anchor graph learning which can effectively preserve visual correlation of images and avoid quantization loss. Then, we derive a novel online optimization algorithm to update hashing functions and compute discrete hashing codes in online manner. The time complexity of online optimization algorithm is linear to the size of new images and irrelevant to the database size. Experimental results on real-world dataset Market-1501 demonstrate both effectiveness and efficiency of ODAGH compared to several state-of-the-art non-hashing and offline hashing person re-id methods.

In future work, we will continue to improve ODAGH to make it more suitable for the real-world application of mobile person re-id. For example, several UAVs usually work together with each other, and distributed hashing [44] is required for the person re-id of UAV swam.

Data Availability

The Market-1501 data used in this study are from previously reported study which has been cited, and they are publicly available on https://github.com/zhunzhong07/IDE-baseline-Market-1501. The experimental result data used to support the findings of this study are available from the first author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This study was supported by the National Natural Science Foundation of China (no. 61702388), Equipment Pre-Research Fund (JZX7Y20190253036101), Equipment Pre-Research Ministry of Education Joint Fund (6141A02033703), and Hubei Natural Science Foundation (2019CFC897).

References

[1] J. Liu, W. Jing, and M. Liu, "UAV monitoring and forecasting model in intelligent traffic oriented applications," *Computer Communications*, vol. 153, pp. 499–506, 2020.

[2] G. Kataev, V. Varkentin, and K. Nikolskaia, "Method to estimate pedestrian traffic using convolutional neural network," *Transportation Research Procedia*, vol. 50, pp. 234–241, 2020.

[3] X. Liu, Y. Mu, D. Zhang, B. Lang, and Y. Li, "Large-scale unsupervised hashing with shared structure learning," *IEEE Transactions on Cybernetics*, vol. 45, no. 9, pp. 1811–1822, 2014.

[4] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," *Advances in Neural Information Processing Systems*, vol. 282, no. 3, pp. 1753–1760, 2009.

[5] W. Liu, J. Wang, R. Ji, Y. Jiang, and S. Chang, "Supervised hashing with kernels," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2074–2081, Providence, RI, USA, June 2012.

[6] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2916–2929, 2013.

[7] L.-K. Huang, Q. Yang, and W.-S. Zheng, "Online hashing," *IEEE Trans Neural Network Learning System*, vol. 29, no. 6, pp. 2309–2322, 2017.

[8] C. Leng, J. Wu, J. Cheng, X. Bai, and H. Lu, "Online sketching hashing," in *Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition IEEE Computer Society*, pp. 2503–2511, Boston, MA, USA, June 2015.

[9] W. Kang, W. Li, and Z. Zhou, "Column sampling based discrete supervised hashing," in *Proceedings of the AAAI Conference Artificial Intelligence (AAAI)*, pp. 1230–1236, Phoenix, AZ, USA, February 2016.

[10] X. Shi, F. Xing, J. Cai, Z. Zhang, Y. Xie, and L. Yang, "Kernel-based supervised discrete hashing for image retrieval," in *Proceedings of the European Conference on Computer Vision Springer International Publishing*, Amsterdam, The Netherlands, October 2016.

[11] F. Shen, C. Shen, W. Liu, and H. Tao Shen, "Supervised discrete hashing," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) IEEE*, Boston, MA, USA, June 2015.

[12] Q. Jiang and W. Li, "Scalable graph hashing with feature transformation," in *Proceedings of the Joint Conference*

Artificial Intelligence (IJCAI), pp. 2248–2254, Buenos Aires, Argentina, July 2015.

[13] J. Gui, "Deep supervised discrete hashing," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 27, p. 1, 2017.

[14] X. Luo, L. Nie, X. He, Y. Wu, Z. Chen, and X. Xu, "Fast scalable supervised hashing," in *Proceedings of the 41st International ACM SIGIR Conference Research Development Information Retrieval (SIGIR)*, pp. 735–744, Ann Arbor, MI, U.S.A., July 2018.

[15] M. Hu, Y. Yang, F. Shen, N. Xie, and H. T. Shen, "Hashing with angular reconstructive embeddings," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 545–555, 2018.

[16] T.-T. Do, A.-D. Doan, and N.-M. Cheung, "Learning to hash with binary deep neural network," in *Proceedings of the Computer Vision - ECCV 2016*, pp. 219–234, Amsterdam, The Netherlands, October 2016.

[17] F. Shen, Y. Xu, L. Liu, Y. Yang, Z. Huang, and H. T. Shen, "Unsupervised deep hashing with similarity-adaptive and discrete optimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 3034–3044, Dec. 2018.

[18] H. P. Zhu, "Massive-scale image retrieval based on deep visual feature representation," *Journal of Visual Communication and Image Representation*, vol. 70, Article ID 102738, 2020.

[19] W. He, X. Zhu, D. Cheng, R. Hu, and S. Zhang, "Unsupervised feature selection for visual classification via feature-representation property," *Neurocomputing*, vol. 236, pp. 5–13, 2017.

[20] X. Deng, K. Y. Liao, Y. L. Zheng, and H. Lei, "A deep multi-feature distance metric learning method for pedestrian re-identification," *Multimedia Tools and Applications*, vol. 80, pp. 1–19, 2021.

[21] Y. Ruan, Y. Xiao, Z. Hao, and B. Liu, "A nearest-neighbor search model for distance metric learning," *Information Sciences*, vol. 552, pp. 261–277, 2021.

[22] L. Zheng, S. Wang, L. Tian, F. He, Z. Liu, and Q. Tian, "Query-adaptive late fusion for image search and person re-identification," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1741–1750, Boston, MA, USA, June 2015.

[23] S. Li, M. Shao, and Y. Fu, "Cross-view projective dictionary learning for person re-identification," in *Proceedings of the 24th International Conference on Artificial IntelligenceIJCAI'15*, pp. 2155–2161, Buenos Aires, Argentina, July 2015.

[24] B. Ma, Y. Su, and F. Jurie, "Local descriptors encoded by fisher vectors for person re-identification," in *Proceedings of the 12th International Conference on Computer Vision - Volume Part I ECCV'12*, pp. 413–422, Florence, Italy, October 2012.

[25] H. Yao, S. Zhang, R. Hong, Y. Zhang, C. Xu, and Q. Tian, "Deep representation learning with Part Loss for person Re-identification," *IEEE Transactions on Image Processing*, vol. 28, no. 6, pp. 2860–2871, 2019.

[26] B. Prosser, W.-S. Zheng, S. Gong, and T. Xiang, "Person Re-identification by support vector ranking," in *Proceedings of the British Machine Vision Conference*, pp. 1–11, Aberystwyth, UK, August 2010.

[27] M. Kostinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof, "Large scale metric learning from equivalence constraints," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2288–2295, Providence, RI, USA, June 2012.

[28] M. Hirzer, P. M. Roth, and H. Bischof, "Person Re-identification by efficient impostor-based metric learning," in *Proceedings of the 2012 IEEE 9th International Conference on Advanced Video and Signal-Based Surveillance*, pp. 203–208, Beijing, China, September 2012.

[29] P. M. Roth, M. Hirzer, M. Köstinger, C. Beleznai, and H. Bischof, "Mahalanobis distance learning for person Re-identification," *Person Re-Identification*, Springer, Berlin, Germany, pp. 247–267, 2014.

[30] J. Chen, Y. Wang, J. Qin, L. Liu, and L. Shao, "Fast person Re-identification via cross-camera semantic binary transformation," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5330–5339, Honolulu, HI, USA, July 2017.

[31] Z. Liu, J. Qin, A. Li, and L. Van Gool, "Adversarial binary coding for efficient person Re-identification," in *Proceedings of the 2019 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 700–705, Shanghai, China, July 2019.

[32] L. Wu, Y. Wang, Z. Ge, Q. Hu, and X. Li, "Structured deep hashing with convolutional neural networks for fast person Re-identification," *Computer Vision and Image Understanding*, vol. 167, pp. 63–73, 2017.

[33] F. Zheng and L. Shao, "Learning cross-view binary identities for fast person Re-identification," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence IJCAI'16*, pp. 2399–2406, New York, Ny, USA, July 2016.

[34] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, "Bit-Scalable deep hashing with regularized similarity learning for image retrieval and person Re-identification," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 4766–4779, 2015.

[35] F. Zhu, X. Kong, L. Zheng, H. Fu, and Q. Tian, "Part-based deep hashing for large-scale person Re-identification," *IEEE Transactions on Image Processing*, vol. 26, no. 10, pp. 4806–4817, 2017.

[36] G. a. Wang, S. Gong, J. Cheng, and Z. Hou, "Faster person Re-identification," in *Proceedings of the Computer Vision - ECCV 2020*, pp. 275–292, Glasgow, UK, August 2020.

[37] A. Brunetti, D. Buongiorno, G. Francesco Trotta, G. F. Trotta, and V. Bevilacqua, "Computer vision and deep learning techniques for pedestrian detection and tracking: a survey," *Neurocomputing*, vol. 300, pp. 17–33, 2018.

[38] M. Zhong, C. Li, L. Liu, J. Wen, J. Ma, and X. Yu, "Fuzzy n learning for deep 3-D segmentation of point cloud," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 12, pp. 3181–3192, 2020.

[39] L. Liu, F. Nie, A. Wiliem, Z. Li, T. Zhang, and B. C. Lovell, "Multi-modal Joint clustering with application for unsupervised attribute discovery," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4345–4356, 2018.

[40] F. Nie, R. Zhang, and X. Li, "A generalized power iteration method for solving quadratic problem on the Stiefel manifold," *Science China*, vol. 60, no. 11, pp. 146–155, 2017.

[41] L. Zheng, L. Shen, L. Tian, S. Wang, T. Wang, and Q. Tian, "Scalable person Re-identification: a benchmark," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV) IEEE*, Santiago, Chile, December 2015.

[42] L. Xie, L. Zhu, P. Pan, and Y. Lu, "Cross-Modal Self-Taught Hashing for large-scale image retrieval," *Signal Processing*, vol. 124, pp. 81–92, 2016.

[43] W. Liu, C. Mu, and S. Kumar, "Discrete graph hashing," *Advances in Neural Information Processing Systems*, vol. 4, pp. 3419–3427, 2014.

[44] S. Wang, C. Li, and H.-L. Shen, "Distributed graph hashing," *IEEE Transactions on Cybernetics*, vol. 49, no. 5, pp. 1896–1908, 2019.

Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 4438861, 15 pages
https://doi.org/10.1155/2021/4438861

Research Article

Development of AI-Based Vehicle Detection and Tracking System for C-ITS Application

Sehyun Tak,[1] **Jong-Deok Lee**,[1] **Jeongheon Song**,[2] **and Sunghoon Kim**[1]

[1]*The Korea Transport Institute, 370 Sicheong-daero, Sejong-si 30147, Republic of Korea*
[2]*CAL Lab., HyperSensing Inc., 169-84 Gwahak-ro, Yuseong-gu, Daejeon 34133, Republic of Korea*

Correspondence should be addressed to Sunghoon Kim; sunghoon@koti.re.kr

Received 28 April 2021; Revised 7 July 2021; Accepted 11 August 2021; Published 19 August 2021

Academic Editor: Wen LIU

There are various means of monitoring traffic situations on roads. Due to the rise of artificial intelligence (AI) based image processing technology, there is a growing interest in developing traffic monitoring systems using camera vision data. This study provides a method for deriving traffic information using a camera installed at an intersection to improve the monitoring system for roads. The method uses a deep-learning-based approach (YOLOv4) for image processing for vehicle detection and vehicle type classification. Lane-by-lane vehicle trajectories are estimated by matching the detected vehicle locations with the high-definition map (HD map). Based on the estimated vehicle trajectories, the traffic volumes of each lane-by-lane traveling direction and queue lengths of each lane are estimated. The performance of the proposed method was tested with thousands of samples according to five different evaluation criteria: vehicle detection rate, vehicle type classification, trajectory prediction, traffic volume estimation, and queue length estimation. The results show a 99% vehicle detection performance with less than 20% errors in classifying vehicle types and estimating the lane-by-lane travel volume, which is reasonable. Hence, the method proposed in this study shows the feasibility of collecting detailed traffic information using a camera installed at an intersection. The approach of combining AI and HD map techniques is the main contribution of this study, which shows a high chance of improving current traffic monitoring systems.

1. Introduction

Urban road traffic is a complex phenomenon caused by interactions among various moving entities, such as vehicles and pedestrians. The growth in urban population during the past decades has raised the severity of urban traffic congestion, leading to socioeconomic and environmental problems in modern cities. To mitigate this issue, brisk trials have been conducted to apply intelligent transportation systems (ITS) in urban roads. In this regard, traffic monitoring is one of the most valuable functions of traffic management systems (TMSs). Particularly in advanced TMSs (ATMSs), real-time collection of precise information through traffic monitoring plays a crucial role for traffic managers when they develop various control strategies [1–3]. Furthermore, the detailed numerical status of real-time traffic such as lane-by-lane travel volume and queue length can be used as supplementary information for cooperative intelligent transportation system (C-ITS) operations based on autonomous vehicles [4, 5].

Traffic monitoring systems have been developed in various ways, and traffic information is collected indirectly or directly depending on the characteristics of a specific monitoring system. Indirect methods estimate traffic status such as travel volume and travel time within a road section based on the data samples collected via roadside units (RSU) or global positioning systems (GPS), which are instances of automatic vehicle identification (AVI) technologies [6–8]. However, the estimation performance of these methods is highly dependent on the market penetration rate (MPR) of equipped vehicles for vehicle-to-infrastructure (V2I) communication. On the contrary, direct methods measure the traffic conditions using point sensors such as loop detectors [9–11], radars [12–14], and video cameras [15, 16]. Loop

detectors have been widely used for traffic monitoring due to relatively higher reliability in collecting travel volume, occupancy, and spot speed, but their installation and maintenance complexity is higher because they are normally installed on road surfaces [17]. Radar-based monitoring systems are relatively easier to install, but the cost of the hardware itself is more expensive [18]. Moreover, the common limitation of both loop detectors and radar is difficulty in classifying vehicle types. However, cameras are relatively cheaper than radars, and camera-based monitoring systems are able to classify vehicle types [19]. They can also distinctively obtain traffic information in each lane of a road spot [17]. They have a high potential for extracting more detailed traffic information at a specific location, but it requires advanced image processing techniques to obtain reliable information, which is problematic. Automatic traffic data collection via camera-based monitoring systems can be operated at lower costs only when proper image processing techniques support the system.

Various methods have been proposed in several studies related to automatic image processing techniques. Some studies from the early 2000s had focused on improving the poor performance with respect to vehicle detection owing to several technical issues, such as segmentation of objects in the background and shadows [20], difficulties in detecting dark-colored vehicles [21], differences in day- and night-vision data [22], and influences of weather conditions [23]. An attempt to develop a technique to detect accidents automatically was also reported [24].

Recently, studies began to focus on using machine-learning or deep-learning techniques, and one of the most popular examples is the application of You Only Look Once (YOLO) to process traffic vision data [25]. YOLO has high applicability to real-time traffic monitoring based on its capability to process multiple images faster than conventional region-based convolutional neural networks (R-CNNs). With the aid of deep-learning techniques such as YOLO or faster R-CNN, the performance of detecting vehicles using real-time traffic vision data has been tried to improve in several studies. Their common purpose was to accurately count vehicles for estimating traffic conditions in specific road spots [26]. Some of them specifically focused on detecting vehicles in captured scenes with several objects (vehicles) with high density [27], while others focused on detecting small objects (vehicles) in complex scenes [28, 29]. Some studies have also attempted to distinctively detect road vehicles and pedestrians [30, 31].

Such object detection techniques have evolved into real-time visual object tracking approaches. Several studies have proposed methods for tracking multiple objects in time series based on convolutional neural networks (CNNs) [32–34]. There are also some examples of using kernelized correlation filter (KCF) for high-speed tracking of objects on roads and even in waterway traffic [35, 36]. Within the context of object tracking on roads, there were a few studies related to tracking moving vehicles particularly for the purpose of collecting more detailed traffic behaviors [37]. They have proposed methods for extracting and analyzing trajectories of multiple vehicles within a specified road spot for capturing lane-change events [38] or measuring the speeds of individual vehicles [39]. However, till now, only rough estimations have been conducted on trajectories without accurately measuring vehicle positions over time. For example, with the current machine-learning-based image processing techniques, a possibility of detecting multiple vehicles as a single object arises when they travel through similar paths and speeds, even though on different road lanes. Hence, it is still difficult to obtain an accurate trajectory of a vehicle by tracking the exact position of the road lane where the vehicle is located. Obtaining accurate trajectories of multiple vehicles would be advantageous to traffic managers intending to improve the accuracy of collecting travel volume or queue length values in each traveling direction at an intersection. Furthermore, it would enable us to obtain information on different road lanes, which can be useful for deeper analysis of traffic flow behavior and supporting autonomous vehicle operations.

Therefore, we present a method for deriving traffic information using a camera installed at an intersection for improving monitoring performance. The method uses a deep-learning-based approach for image processing for vehicle detection and vehicle type classification. Then, the method estimates lane-by-lane vehicle trajectories by matching the detected vehicle locations with the high-definition map (HD map). While estimating the vehicle trajectories, we attempt to reduce the error of estimating the center points of the bounding boxes in the images of vehicles to ensure proper performance of the HD map-matching process. Based on the estimated vehicle trajectories, the traffic volumes of each lane-by-lane traveling direction and queue lengths of each lane were estimated as well. In fact, this is not the first attempt to increase the accuracy of trajectory estimation to the lane level. The work in [40] had a similar purpose and approach but differs from the present study in that recent deep-learning techniques and HD map technology are combined for estimating vehicle positions accurately.

The remainder of this paper is organized as follows. Section 2 provides a description of the method of vehicle detection and classification, along with the method of matching the detected vehicle positions with the HD map for lane-by-lane trajectory estimation. Section 3 describes the settings for testing the performance of the proposed method, and Section 4 presents the test results. Section 5 concludes this paper and offers suggestions for further work.

2. AI-Based Vehicle Detection System at Intersection

2.1. Data Flow Framework. In fact, the image processing technology these days can easily identify a vehicle in a captured image, as long as the image resolution is sufficient. However, the focus of this study is on how to precisely extract traffic information upon multiple vehicles on roads rather than a single vehicle and how to deal with the extracted data from the traffic monitoring perspective. Hence, it is necessary to consider the data flow framework of the camera-based vehicle detection system.

Figure 1 shows the data flow framework of the artificial intelligence (AI) based vehicle detection system for C-ITS. As shown, the system consists of four components: roadside sensor, traffic monitoring center, RSU for communication, and an on-board unit (OBU) in vehicles. In this study, traffic cameras installed at intersections were considered as the main roadside sensors. First, the vision data of the traffic status at an intersection were collected in real-time via a roadside sensor and sent to a data collecting server in the traffic monitoring center. Then, using the vision data, the center conducted the vehicle detection task using the AI-based image processing technique. The information gathered from the vehicle detection task was then used to extract and predict the trajectories of vehicles. Then, the trajectory data information underwent the HD map-matching task to improve the prediction accuracy. The information message of the detected vehicles and their predicted trajectories were sent to the OBU in a subject vehicle via RSU using infrastructure-to-vehicle (I2V) communication. When a message was received, the collision risk of the subject vehicle could be calculated based on the predicted trajectories and also be displayed to the vehicle monitoring system. The status of the subject vehicle could be sent back to the traffic monitoring center via the RSU using V2I communication.

The framework described above provides two major advantages in terms of C-ITS operations. The first is that vehicle-to-vehicle collisions can be prevented by providing vehicles with their detection information traveling through intersections. Implementing a service that provides detailed information, such as vehicle location, speed, and abnormal status, is possible. In addition, it provides predictive information in seconds using the previously detected information. Second, a more detailed road status can be provided by extracting lane-by-lane traffic conditions near intersections. It is possible to provide a service that provides information on the traffic volume and vehicle queue of each lane. Furthermore, a service that detects illegally parked vehicles on streets can also be implemented. In this study, we aim to improve the advantages of the framework. The focus of this study is to develop methodologies for AI-based vehicle detection and HD map matching, which are the tasks of the traffic monitoring center described above.

2.2. AI-Based Vehicle Detection and Trajectory Prediction. In this study, a deep-learning algorithm is adopted using roadside sensors to extract object information such as vehicle location, movement trajectory, and vehicle speed at intersections and surrounding areas, and useful traffic information, such as traffic volume and queue length, is estimated. The proposed algorithm is based on vision data transmitted from the roadside sensors to a vision data collecting server located in the traffic monitoring center, and the predicted data are stored in a real-time database for real-time data communication. As shown in Figure 2, the proposed algorithm consists of (1) vehicle detection and classification with deep learning, (2) trajectory extraction, (3) trajectory correction, and (4) trajectory prediction, and the details are outlined as follows.

2.2.1. Vehicle Detection and Classification with Deep Learning. We used a deep-learning-based algorithm for vehicle detection as it has higher applicability to real-time traffic monitoring compared to other image processing techniques such as traditional labeling due to its capability of processing multiple images faster than others. The proposed system performs real-time detection of vehicle location and speed from the vision data sent from the vision data collecting server based on the YOLOv4 deep-learning algorithm and performs vehicle type classification. The YOLOv4 algorithm uses the state-of-the-art deep-learning method and is optimized, showing 10% improved performance for the detection accuracy index (MAP: mean average prediction) and a 12% improved detection speed index (FPS: frame per second) compared to YOLOv3, the previous version of the algorithm. In particular, YOLOv4 can process vision data with efficiency, enhancing its applicability in the traffic safety sector where detection, preprocessing, and warning message generation must be performed within 0.1 seconds.

In the process of vehicle detection and classification with deep learning, the algorithm processes vision data in frames and primarily generates vehicle type information such as cars, trucks, and buses and vehicle location information based on pixels. As for vehicle type information, data derived from YOLOv4 can be directly used, and additional separate training was performed based on the target site data to improve the accuracy of vehicle type information. Vehicle location information was generated based on the information of each vertex and the center point of the bounding box. This information was then converted into longitude and latitude coordinates based on the center point of the vehicle's bottom through correction.

2.2.2. Trajectory Extraction. The vision data collected from the roadside are distorted when converting 3D real-world images into 2D images. Because of this distortion, a significant error occurs between the actual physical coordinates and the image coordinates depending on the degree of vision data distortion when the location information detected in pixel units is converted directly into longitude and latitude coordinates. In this study, to remove this error, the corrected vision data were generated from the distorted vision data by inverse application of the camera intrinsic parameters extracted through its calibration. Note that the focal length, principal point, and distortion are the intrinsic parameters of the camera. The values of the intrinsic parameters were determined by projecting a 2D image into 3D world space. Also, note that an existing method is used for the distortion correcting process in this study. For a better understanding of the details of the distortion correcting method, refer to the work by Seong et al. [40]. The equation for correcting the vision data distortion is as follows:

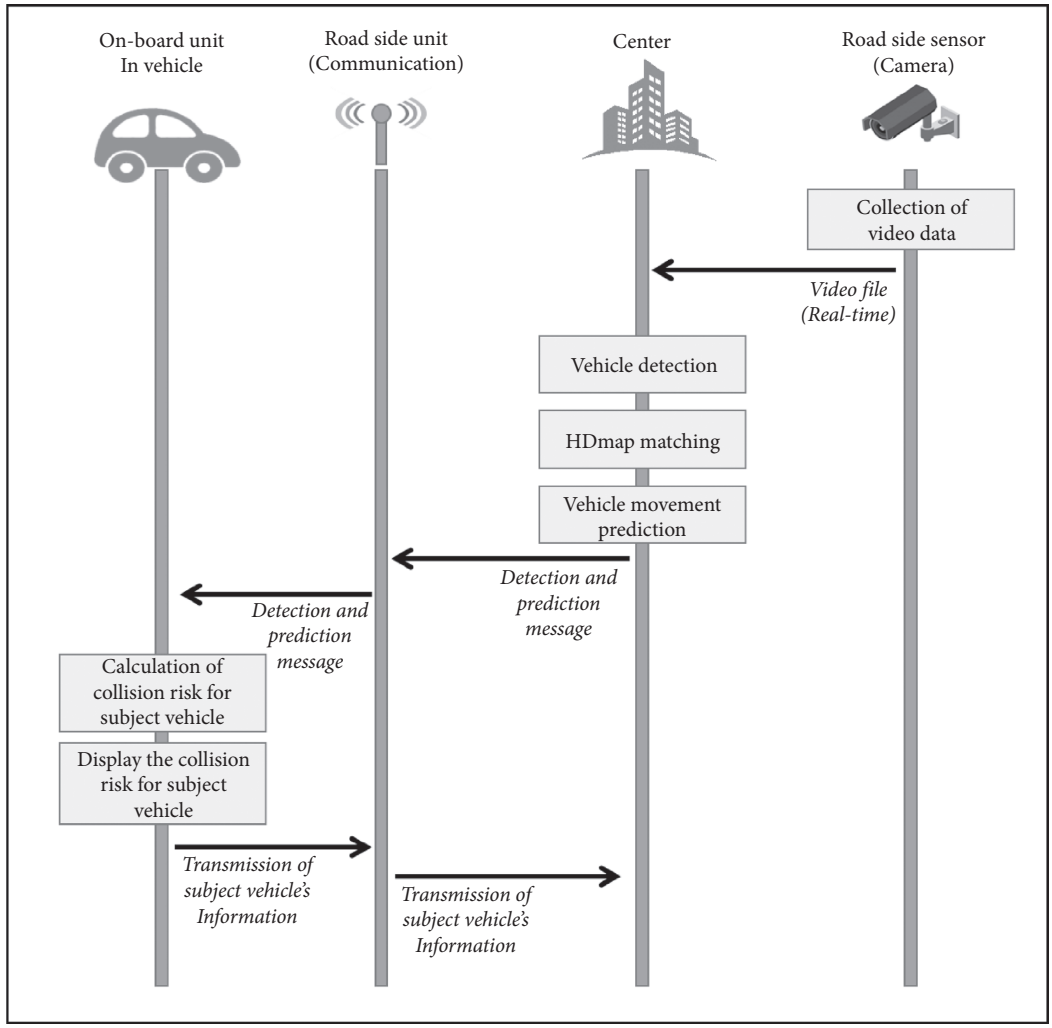


Figure 1: Data flow of AI-based Vehicle Detection system for C-ITS.

$$
\begin{bmatrix} x_v \\ y_v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \text{skew} & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x_{\text{pu}} \\ y_{\text{pu}} \\ 1 \end{bmatrix},
$$

$$
r_u^2 = x_v^2 + y_v^2,
$$

$$
\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \left(1 + k_1 r_u^2 + k_2 r_u^4 + k_3 r_u^6\right) \begin{bmatrix} x_v \\ y_v \end{bmatrix} + \begin{bmatrix} 2p_1 x_v y_v + p_2\left(r_u^2 + 2x_v^2\right) \\ p_1\left(r_u^2 + 2y_v^2\right) + 2p_2 x_v y_v \end{bmatrix}, \tag{1}
$$

$$
\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \text{skew} & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix},
$$

where x_p, y_p are the pixel coordinates of an image, x_{pu}, y_{pu} are the pixel coordinates of the image corrected for distortion, x_n, y_n are the normalized planar coordinates with distortion, and x_v, y_v are the normalized planar coordinates with corrected distortion. Focal length: $f_x = 664.821$; $f_y = 668.333$. Principal point: $c_x = 350.377$; $c_y = 350.377$. Distortion: $k_1 = 0.278027$, $k_2 = 0.058863$, $p_1 = 0.000278$, and $p_2 = -0.001996$.

The vehicle location information detected from each image frame was expanded to a continuous frame for extracting the vehicle trajectory information and data for use in vehicle location correction. In the video images captured

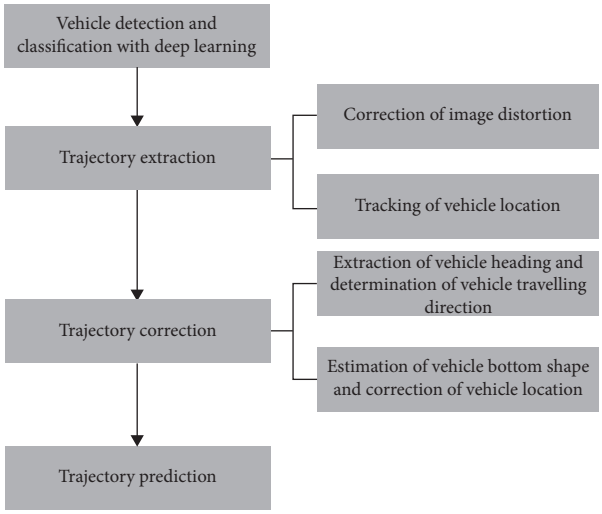


Figure 2: Algorithm for vehicle detection and trajectory prediction with deep learning.

by a camera, the similarity between the feature information of the object in the image frame is used to track the location change of the objects. To track the vehicle's location, its location and size in the previous frame were compared with those of the vehicle object detected in the next frame. As a result, the vehicle with the largest intersection of union (IOU) was classified as an identical vehicle to the vehicle existing in the previous frame; based on this classification, the continuous movement of the vehicle was tracked. In addition, if there was no intersection of union where the location and size of the detected object for a set frame (0.2 s) overlapped with that in the previous frame, the object was recognized as a new object, and a new vehicle tracking ID was assigned.

The pixel coordinates extracted from an image are calculated based on a matrix transformed using Transverse Mercator coordinates of four designated points in the HD map. The transformed matrix is derived by using homography that generalizes transformation relationships after obtaining coordinates corresponding to sample image coordinates. If there are four points (x_1, y_1), (x_2, y_2), (x_3, y_3), and (x_4, y_4) in a plane and these points are projected to another plane as (x_1', y_1'), (x_2', y_2'), (x_3', y_3'), and (x_4', y_4'), there exists a 3 by 3 homography matrix H satisfying relationship among these corresponding points. The camera image coordinates are converted to real-world coordinates using such a mechanism.

2.2.3. Trajectory Correction. In general, deep-learning-based vehicle detection extracts information in the form of a bounding box, and the central point of the bounding box represents the overall vehicle location information. However, as shown in the example in Figure 3, when vehicle location information is extracted with reference to the center point of the bounding box, the result differs from the location with reference to the center point of the vehicle bottom, which is the actual required information for traffic monitoring. In addition, when the center point is estimated

Figure 3: Comparison before (red-colored dot) and after (orange-colored dot) the trajectory correction.

based on the bounding box, an error occurs in the estimated position according to the heading shown (by captured angle) in the vehicle image. This type of error can lead to another error in trajectory prediction. This subsequent error can lower the performance of the HD map-matching process, which deals with extracting lane-by-lane traffic information later. Furthermore, if we assume that the trajectory prediction with such an error is utilized in a vehicle's collision warning or avoidance system, it can also lead to insufficient performance of the safety system. Hence, it is necessary to give an effort in reducing the errors while estimating the center point of the bounding box.

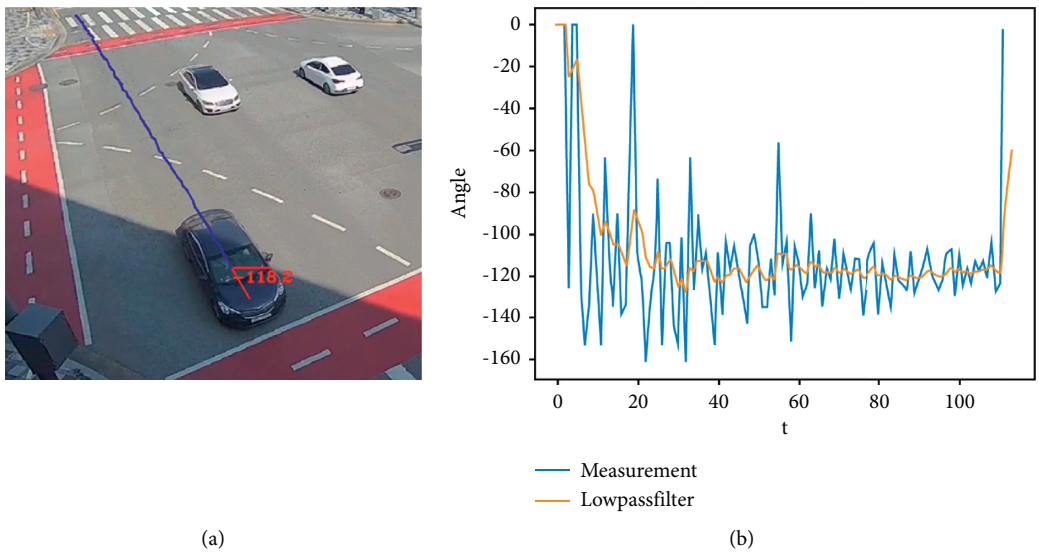
In this study, to reduce the error in center point estimation, real-time correction of vehicle location was performed through the following two steps: (1) extracting the heading and determining the traveling direction of the vehicle and (2) estimating the shape of the vehicle bottom and correcting the location.

For the first task, the vehicle heading was obtained using the pixel coordinates detected in the vision data collected from the road (the bounding box center point value) and the pixel coordinates of the previous frame, as shown in Figure 4. The heading of a vehicle is extracted through the following steps: (1) The vehicle position of the previous image frame and the position of the current image frame are converted into coordinates using a transformation matrix. (2) The angle formed by the two positions is calculated using the Pythagorean equation, and the distance between the two positions is calculated using the coordinate values. The extracted heading for each frame was corrected based on the low-pass filter as follows:

$$\overline{z_n} = \sigma \cdot \overline{z_{n-1}} + (1 - \sigma) \cdot z_n, \tag{2}$$

where $\overline{z_n}$ is the corrected heading, $\overline{z_{n-1}}$ is the heading at previous time, z_n is the heading at current time, and σ is the weight.

The vehicle traveling direction and the vertical direction are derived using the heading obtained from the real-time estimation and the detected pixel coordinates. Figure 4(a) shows the corrected results of the low-pass filter. In Figure 4(b), the orange and blue colored lines represent the



(a) (b)

Figure 4: Example of vehicle heading estimation. (a) Example image. (b) Correction result (example).

filtered and raw data, respectively. Noisy data points and variation in heading information are smoothed using a low-pass filter.

Based on the previously derived information of vehicle traveling direction and vehicle type, the shape of the vehicle bottom was estimated, as shown in Figure 3. Because the vehicle height varies depending on the type, the shape of the bottom surface within the bounding box is estimated by applying the average vehicle height per vehicle type. The bottom surface information is estimated based on the following steps: (1) The center points of the bounding boxes in the previous image frame and in the current image frame are converted into Transverse Mercator coordinates. (2) Since the vector formed by the two center points is the moving direction of the vehicle, a hypothetical vector perpendicular to the moving direction is drawn to create a rectangular vehicle bottoms shape (assuming that vehicles have a rectangular shape from the top view). (3) Let h_{camera} be the height between camera and ground surface, h_{vehicle} be the height of a vehicle, d_1 be the distance on the surface between camera and vehicle, and d_2 be the distance on the surface between the camera and point where the line connecting between the camera and the top of the vehicle meets the surface. Here, h_{camera}, h_{vehicle}, and d_2 are directly obtained from image data, and d_1 then can be calculated by the triangle proportional theorem. Note that the height of the vehicle is assumed to be half of the actual height because the center point of the bounding box detected in the image is half the actual height in usual. Based on this method, the four corner points (in 3D coordinates) of the vehicle bottom are estimated. (4) The 3D coordinates of the vehicle bottom (a', b', c', d') are then converted into the image coordinates (a'', b'', c'', d'') using an inverse transformation matrix, and this finalizes estimating the vehicle bottom. The center point information of the vehicle's bottom surface is extracted based on the estimated pixel information of the bottom surface, and the final pixel-based location information of the vehicle is derived based on this information.

2.2.4. Trajectory Prediction. Using the previously derived real-time trajectory data of the vehicle, the upcoming vehicle trajectory information from 1 to 3 seconds was estimated. Location information for each time slot was used to estimate the future trajectory of the vehicle. In addition, a polynomial curve fitting algorithm was used, as shown in the following equation, by applying a linear equation if the past data is a vehicle traveling forward or a quadratic equation for a turning vehicle, to extract the future location of the vehicle.

$$y(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_n + 1,$$

$$\begin{pmatrix} x_1^n & x_1^{n-1} & \cdots & 1 \\ x_1^n & x_1^{n-1} & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ x_m^n & x_m^{n-1} & \cdots & 1 \end{pmatrix} \begin{pmatrix} p_1 \\ p_1 \\ \vdots \\ p_{n+1} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}. \quad (3)$$

When estimating the future location of the vehicle based on the detected vehicle location information alone, the result showed that the prediction performance is decreased at the intersection approach where a fewer number of points exist in the trajectory data. To address this limitation, the HD map previously built at the intersection was used, as shown by the solid black lines in Figure 5. Using the location information per link in the HD map, the future vehicle location was estimated assuming that the vehicle trajectory will follow the shape of the HD map link, and the estimated result is shown in Figure 5. The blue solid line represents the ground truth, the green- and blue-dotted lines represent the link of the HD map where the detected vehicle is assigned, and the red-dotted line represents the estimated future location of the vehicle.

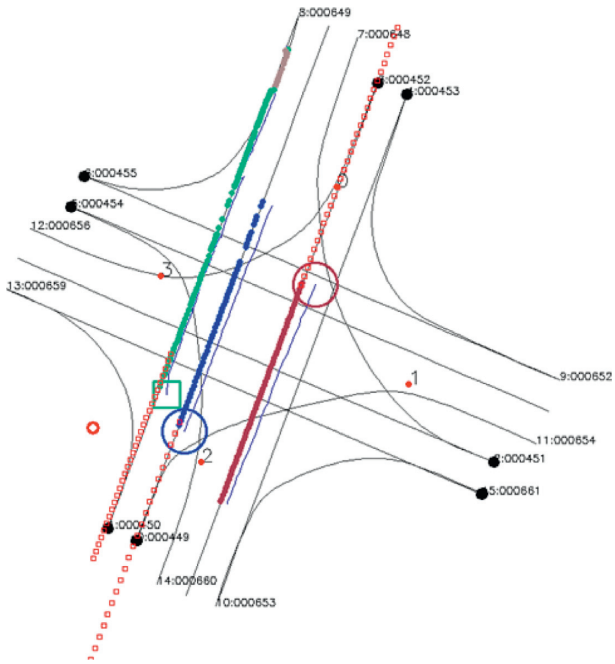


Figure 5: Example of vehicle location prediction.

2.3. Provision of V2X Communication-Based Detection Information

2.3.1. Generation of HD Map-Based Information. The current C-ITS provides information such as unforeseen incidents or accidents via messages that include longitude and latitude data. For such type of information, C-ITS has an advantage in terms of general use of information but is disadvantageous when the number of messages increases sharply with the increase in the number of related pieces of information. Furthermore, in the case of existing C-ITS based on location information, the computational load increases rapidly as the number of messages increases when matching the predicted vehicle trajectory information and point of event occurrence for each event. What is worse in the case of predicting trajectory based on the past trajectory is that the accuracy decreases at curved sections and intersections, leading to reduced accuracy when matching events. Therefore, in this study, to overcome the limitation in sending location information based on longitude and latitude, the AI-based detection and prediction information provided in the previous subsection was combined with the HD map link information, as shown in Figure 6(a).

Figure 6(b) shows the HD map link allocation algorithm. First, in the process of extracting HD map link information, information such as the length, linearity, and type of link and the longitude and latitude of the start and end points are extracted from the link attribute information of the HD map. This information of the HD map is compared with the detected location coordinates of the vehicle, and matching is performed with the nearest link, extracting the lane on which the vehicle is currently traveling. Figure 7 shows an example of the HD map link allocation based on the trajectories of the forward-traveling vehicle and turning vehicle. As shown in the figure, information on whether the vehicle travels forward or turns is extracted based on the vehicle trajectory for the past 1 s. Based on this information, if the vehicle is determined to be traveling forward, links with forward-type traveling are extracted from the HD map links, and the extracted candidate links and vehicle trajectory for 1 s are matched based on the start and end points, thereby extracting the HD map link with the closest matching. Finally, the HD map link extracted based on the distance is compared with the heading of the vehicle traveling direction, and when the latter shows consistency within a set threshold, the HD map link is allocated.

To enhance the applicability of the extracted information based on AI, the information extracted from the vision sensor is allocated in HD map link units. Then, the number of vehicles present in the link representing density, the most necessary information in traffic management, and queue length information are generated by the link. The density is calculated as the difference $(n_{in} - n_{out})$ between the number of vehicles entering the starting point (n_{in}) and that leaving the end point of the link (n_{out}). As for the queue length of a vehicle, when the average speed over the last 1 s is smaller than the set speed for each HD map link, the corresponding vehicle is classified as the vehicle in the queue. To improve the applicability of the information, the queue length is expressed based on the offset of the HD map. For example, if the length of the HD map link is 50 m, the start point of the link is set to 0, and the end point of the link is set to 50 based on the vehicle traveling direction. Based on these values, when the vehicle queue length is 20 m from the end point of the link, the start point of the queue is offset by 30, and the end point by 50.

2.3.2. Data Design for V2X Communication-Based Information Provision. Data converted based on the link format of the HD map are stored in the server in the format shown in Tables 1 and 2 to be utilized in messages in C-ITS in the future. Table 1 shows the storage format of vehicle information, which is used for storing and sharing object information (vehicle type, longitude, and latitude coordinates) extracted from AI. However, to improve the applicability of the information and accuracy of matching with the vehicle trajectory, the information allocated to the HD map link is combined. In addition, providing predicted information of vehicle objects based on the HD map link ID facilitates the calculation of the probability of collision in the future traveling direction of an autonomous vehicle.

Table 2 shows the storage format of data, primarily processed to facilitate the application of the information extracted from AI-based detection information to the traffic management field. As described above, information of the number of vehicles present in the link (density), queue length information, and average speed information is generated with reference to the HD map link. Similar to the storage format of vehicle information (Table 1), the predicted information is provided to facilitate the calculation of the collision probability in the future traveling direction of

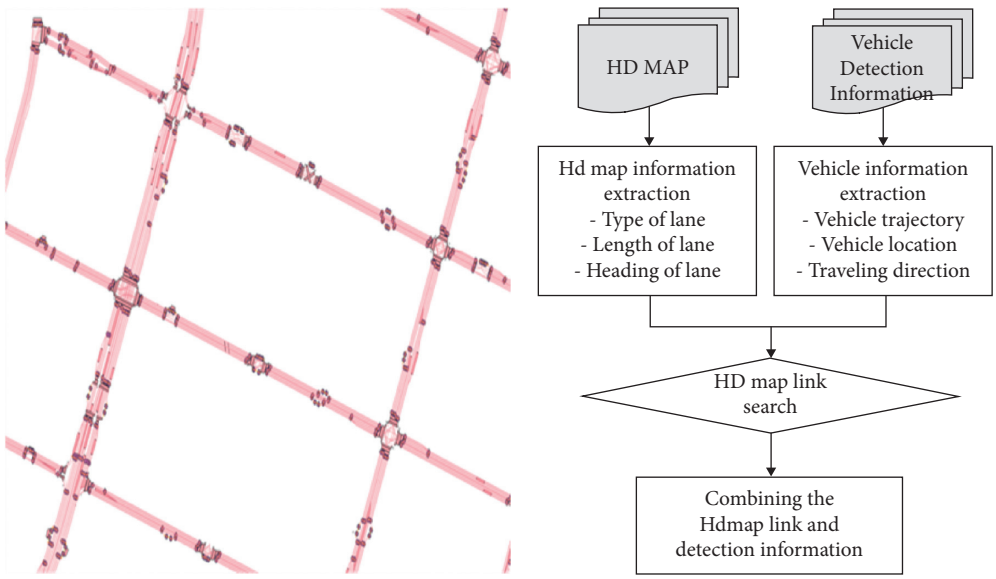


Figure 6: HD map example for generation of HD map-based information of the target site (a) and HD map link allocation algorithm (b).

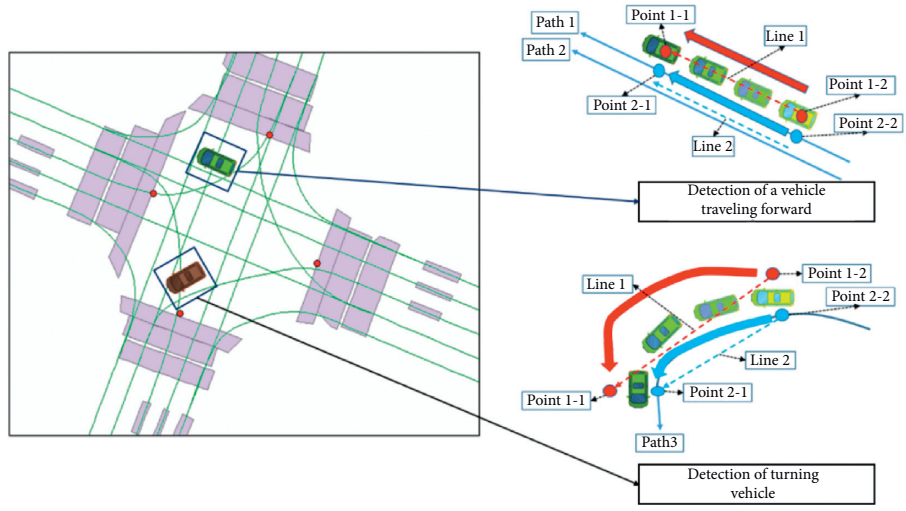


Figure 7: Example of road lane allocation for a vehicle traveling forward and a turning vehicle.

an autonomous vehicle. As shown in Tables 1 and 2, not only does the proposed system enhance the applicability of information by actively utilizing HD map links, but also the object extraction and traffic-related information are provided in combination with a similar format considering the need for other information attributes depending on the situation.

3. Target Site for Application of the Proposed Method and Evaluation

3.1. Target Site. Figure 8 shows the target site for applying the AI-based vehicle detection and prediction technique proposed in this study. The proposed system was evaluated using data collected for three days, and data for accuracy verification were generated in two steps as follows. First, the ground truth data for calculating the accuracy of vehicle location information were generated using a drone, by capturing the same area as the image data collected from the roadside vision sensor and collecting vertical images. Second, information such as the vehicle type, number of vehicles in a link, and queue length was generated based on a field survey, and the vehicle type and the number of vehicles were manually counted from visual observation of image data. To prevent human errors in counting, a cross-check and final check were performed using labeled image data.

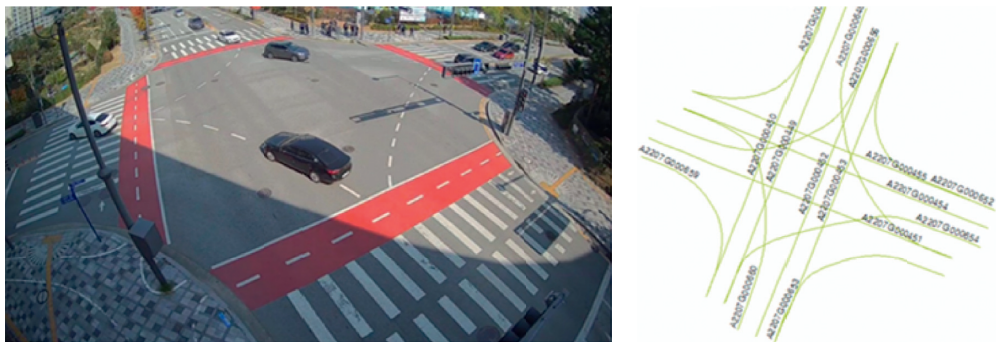
3.2. Accuracy Evaluation. Descriptions of how we evaluate the performance of the proposed methodologies are provided in this subsection, which is based on five different

Table 1: Vehicle information storage type.

Field name		Description
LinkID		Serial number of link ID in road central line of HD map in the detection area
Timestamp		Vehicle detection time: Unix timestamp (UTC) of accuracy in milliseconds
ObjectID		Object ID
Vehicletype		Vehicle type
Vehicletypeprob		Vehicle type probability
Objectstatus		Normal/abnormal traveling status (abnormal when not in motion for a certain period of time)
Offset		Offset of event point for link ID
Posdistance		Distance between the present HD map link and the extracted coordinates in longitude/latitude
Poslong		Longitude
Poslat		Latitude
Speed		Speed of detected vehicle (km/h)
Heading		Heading (°)
Object prediction	Index	Index (0–29 for 3 s prediction in units of 0.1 s)
	Timestamp	Vehicle detection time (prediction)
	Poslong	Longitude
	Poslat	Latitude
	Speed	Speed (km/h)
	Heading	Heading (°)
	LinkID	Link ID of predicted location at the timestamp of the detected object
	Offset	Offset of link ID of predicted location at the timestamp of the detected object

Table 2: Traffic information storage format including the number of vehicles on the link and queue length.

Field name		Description
LinkID		Serial number of link ID in road central line of HD map in the detection area
Timestamp		Vehicle detection time (present)
Avgspeed		Average speed of vehicles with link ID
Linktraveltime		Difference between the entry and exit time of vehicle
Numvehicle		Number of vehicles for the applicable link (present)
Object status	ObjectID	ID of an object with abnormal driving
	Offset	Offset of event point for link ID
Queue	QueueID	Queue event ID
	Offsetstart	Start point of offset of event point for link ID
	Offsetend	End point of offset of event point for link ID
Road prediction	Index	Index (0–29 for 3 s prediction in units of 0.1 s)
	Timestamp	Vehicle detection time (prediction)
	Numvehicle	Number of vehicles for the applicable link (prediction)
	Avgspeed	Average vehicle speed (prediction)



(a) (b)

Figure 8: (a) Example of roadside sensor screenshot. (b) Example of HD map.

evaluation criteria: vehicle detection rate, vehicle type classification, trajectory prediction, traffic volume estimation, and queue length estimation.

3.2.1. Accuracy of Vehicle Detection and Classification. To evaluate the vehicle detection performance, the detection rate was calculated to determine whether all vehicles were

successfully detected regardless of vehicle type. As in equation (4), it is defined as the ratio of the total number of detected objects to that of ground truths:

$$\text{detection rate} = \frac{\text{total number of detected objects}}{\text{total number of ground truths}}. \quad (4)$$

Vehicle classification performance is evaluated through MAP, which is a performance evaluation index widely used in the field of computer vision. MAP is the mean of the average precision (AP) values of each vehicle type. The AP represents the performance of the classification algorithm as a single value, and it is calculated as the area below the graph line in the precision-recall graph. As in equation (5), the precision is calculated as the ratio of the number of correct answers for vehicle type classification (true positives) to that of all detected vehicles (sum of true and false positives). The recall is calculated as the ratio of the number of correct answers (true positives) to that of all ground truths (sum of true positives and false negatives), as shown in equation (5). Precision and recall are inversely related to each other. Hence, the changes in such a relationship are analyzed to properly evaluate the overall performance of the proposed method.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\text{all detection}}, \quad (5)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{all ground truths}}. \quad (6)$$

3.2.2. Accuracy of Vehicle Trajectory Estimation. The performance of the vehicle trajectory prediction was evaluated by comparing the predicted and actual trajectories. The predicted trajectory is the set of coordinates within an intersection derived by the AI-based detection technique, while the actual trajectory is that directly generated from the image data. The average Euclidean distance is used for calculating the prediction accuracy, as shown in the following equation:

$$\text{average Euclidean distance} = \frac{\sqrt{\sum_{t=1}^{N}\left(x_{a,t} - x_{p,t}\right)^2 + \left(y_{a,t} - y_{p,t}\right)^2}}{N}, \quad (7)$$

where N is the number of sets of t for the comparison, $(x_{a,t}, y_{a,t})$ are the actual coordinates of the vehicle location at t, and $(x_{p,t}, y_{p,t})$ are the predicted coordinates at t.

3.2.3. Accuracy of Traffic Volume Estimation. Traffic volume was estimated by comparing the number of vehicles counted by the image processing (estimated value) technique and that counted manually (actual value). The evaluation was performed by calculating the root mean square error (RMSE) and mean absolute percentage error (MAPE). The former is used to check the degree of difference between the estimated and actual values, which can be calculated using the following equation:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}, \quad (8)$$

where n is the number of data points for the comparison, y_i is the i-th predicted value, and $\hat{y}_i$ is the i-th actual value of the traffic volume.

However, RMSE is highly influenced by the size of the estimation subject (scale-dependent errors), and it may emphasize only greater errors than the small ones. Hence, we calculate MAPE as well, which is independent of the scale of the estimation subject and can be calculated using equation (9):

$$\text{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{A_i - F_i}{A_i}\right|, \quad (9)$$

where n is the number of data points for the comparison, A_i is the i-th predicted value, and F_i is the i-th actual value of the traffic volume.

3.2.4. Accuracy of Queue Length Estimation. The evaluation of the performance of the queue length estimation is similar to that of the traffic volume estimation. This is done by comparing the queue length in meters derived by the image processing (estimated value) technique and that collected from a drone image (actual value). Here, we calculate the RMSE of the queue length estimation using equation (8), where $\hat{y}_i$ is the i-th actual value of queue length. We also calculate the MAPE of the queue length estimation using equation (9), where F_i is the i-th actual value of the queue length.

4. Result of Applying AI-Based Vehicle Detection and Trajectory Prediction

4.1. Accuracy of Vehicle Detection and Classification. Figure 9 shows an example of vehicle detection using the proposed training model based on YOLOv4. The system detects vehicles within the detection range and saves the results of the vehicle classification and coordinates of the bounding box as an image file (∗ .jpg) and data files (∗ .txt), using the same filename. By using the data collected by the drone (considered as actual data) and that extracted by the classification model, the performance evaluation is performed with the detection rate and MAP described in the previous section. The number of tested samples was 6,804. As a result, the detection rate was 99%, indicating that it

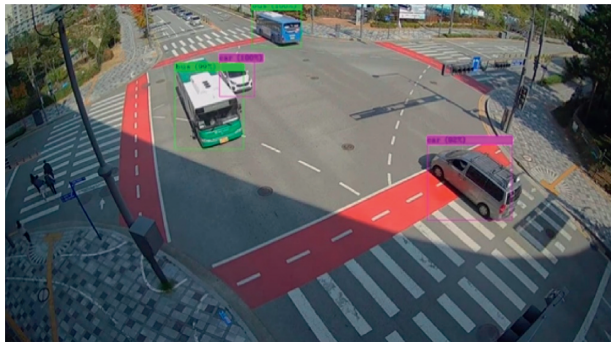


Figure 9: Vehicle detection by the proposed method based on YOLOv4.

could judge the detected objects as vehicles very well. In terms of vehicle type classification, the MAP value was 95% for cars, 87% for trucks, and 81% for buses.

4.2. Accuracy of Vehicle Trajectory Extraction. In vehicle trajectory extraction, during preprocessing, the locations of a vehicle are extracted at each frame by the proposed method. Then, the locations of the vehicles are projected onto the drone image. By matching the vehicle locations of the drone image and those from the proposed method to the same coordinates, it is determined that the vehicle area overlaps the most with the same vehicle. As shown in the figure, the vehicle locations extracted by the proposed method (red box area) are projected onto the drone image, where the actual vehicle locations are displayed in the drone image (blue box area) to determine them as the same vehicle. The performance of vehicle trajectory prediction is evaluated using these two different vehicle trajectories, as described in the previous section. The number of tested samples was 60,531. As a result, the average Euclidean distance was 1.138 m.

4.3. Accuracy of Traffic Volume Estimation. Figure 10 shows an example of comparing drone and camera images for the performance evaluation of traffic volume estimation. The test area is the blue box area within an intersection. As shown in the figure, the identification names are assigned for each in/out lane, and the pairs of the lane-by-lane travel directions of the vehicles can be seen in Tables 3 and 4. Then, the number of vehicles in each lane-by-lane traveling direction is counted from the drone images manually to obtain the actual data. On the contrary, the proposed method extracts the number of vehicles in each lane traveling direction based on the camera images to obtain the estimated data. Table 3 shows the traffic volume in each traveling direction counted from the drone images, and Table 4 shows that extracted from the camera data. In these tables, the notations in the second column represent the identification numbers of departure lanes (from I_1 to I_12) in approaching roads (from Road_1 to Road_4). However, those in the second row are the identification numbers of arrival lanes (from O_1 to O_12) in the roads in each direction. For example, if some vehicles pass through the intersection from the right-most lane of Road_1 (I_1) to the left-most lane of Raod_3 (O_5) and they are counted as 5, we record the counted number as shown in the tables. Hence, the entire table represents the lane-by-lane vehicle count values (travel volumes) of all departure and arrival pairs. The unknown in the latter table is the case when the camera-based system fails to detect a vehicle. When comparing the results of the two, the RMSE is 4.20 vehicles, and the MAPE is 16.41%.

4.4. Accuracy of Queue Length Derivation. Figure 11 shows an example of a drone image for queue length derivation, which was also performed manually. A person selects the starting and ending points of the vehicles within the delayed section on the road. Then, data containing the bounding box information of the vehicles at the starting and end points, map coordinates, and queue length within the image are saved. Using the information from these data, the true value of the queue length is calculated by converting the values into the real-world scale, which is considered the actual queue length. However, the proposed method directly derives the queue length through HD map matching to obtain the estimated data, which is compared with the actual queue length from the drone image. The number of tested samples was 62,205. Comparing the two, the RMSE is 2.37 m, and the MAPE value is 13.25%.

4.5. Comprehensive Evaluation. The overall performance of the proposed method is presented in Table 5. As described in the previous subsections, the detection rate is the total number of detected objects over the total number of ground truths, and a successful detection performance of 99% for 6,804 attempts is achieved, which can be judged to be highly consistent. The performance of the vehicle classification is performed in terms of MAP. With 6,804 test samples, the MAP values were 95%, 87%, and 81% for cars, trucks, and buses, respectively. Hence, the proposed method also shows reasonable performance in classifying the vehicle types. In terms of trajectory prediction, the average Euclidean distance was 1.138 m when 60,531 samples were tested. Such a low degree of error indicates the high performance of the proposed method. In terms of both traffic volume and queue length estimations, the absolute differences are only 4.20 vehicles for vehicle counting and 3.08 m in queue length estimation upon the RMSE values for more than 60,000 test samples. The MAPE values are less than 20%, which means that the performance of the proposed method is reasonable, particularly when estimating the lane-by-lane traffic information. Overall, based on the analyses of the five different evaluation criteria, the method proposed in this study shows the feasibility of collecting detailed traffic information with a camera installed at an intersection. In addition, the average time taken from image collection, data processing, and data storage in the server is 0.034 seconds, showing that the performance of the entire process can be completed within 0.1 seconds in general. Considering the results of this study, the proposed method is a highly optimistic technology to be applied to the fields of ITS and C-ITS.

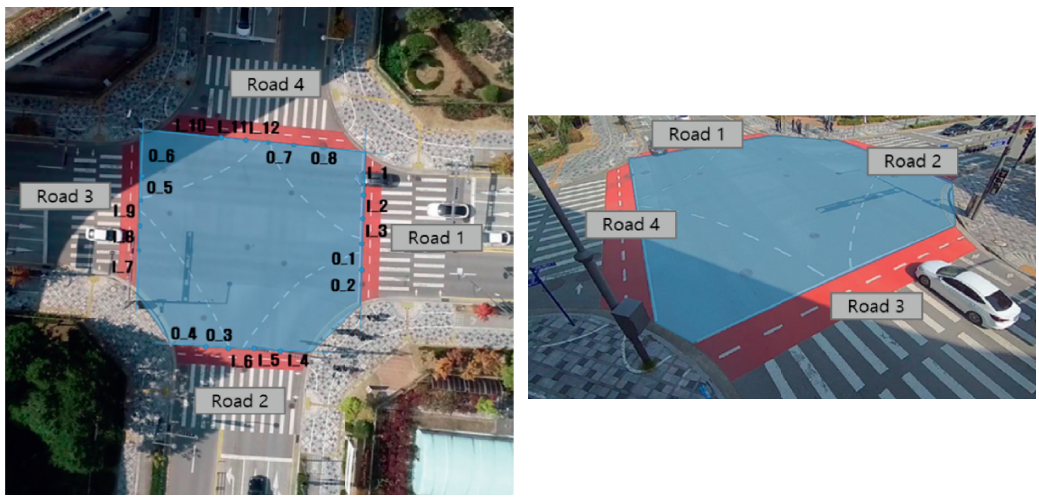


Figure 10: An example of comparing drone image and camera image for traffic volume estimation.

Table 3: Traffic volume in each direction counted from drone images.

Out		Road_1		Road_2		Road_3		Road_4		Total
In		O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	
Road_1	I_1	N/A	N/A	0	0	5	93	0	8	106
	I_2	N/A	N/A	0	0	117	1	0	0	118
	I_3	N/A	N/A	74	0	0	0	0	0	74
Road_2	I_4	0	26	N/A	N/A	0	0	1	9	36
	I_5	0	0	N/A	N/A	0	0	39	0	39
	I_6	0	0	N/A	N/A	47	0	0	0	47
Road_3	I_7	1	59	0	18	N/A	N/A	0	0	78
	I_8	82	1	0	0	N/A	N/A	0	0	83
	I_9	0	0	0	0	N/A	N/A	29	0	29
Road_4	I_10	0	0	0	3	0	35	N/A	N/A	38
	I_11	0	0	32	1	0	0	N/A	N/A	33
	I_12	23	0	0	0	0	0	N/A	N/A	23
Total		106	86	106	22	169	129	69	17	704

Table 4: Traffic volume in each direction extracted through camera data.

Out		Road_1		Road_2		Road_3		Road_4		Unknown	Total
In		O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8		
Road_1	I_1	N/A	N/A	0	0	5	87	0	8	7	107
	I_2	N/A	N/A	0	0	109	1	0	0	3	113
	I_3	N/A	N/A	74	0	0	0	0	0	0	74
Road_2	I_4	0	23	N/A	N/A	0	0	1	8	3	35
	I_5	0	0	N/A	N/A	0	0	37	0	2	39
	I_6	0	0	N/A	N/A	43	0	0	0	3	46
Road_3	I_7	1	49	0	7	N/A	N/A	0	0	12	69
	I_8	79	0	0	0	N/A	N/A	0	0	2	81
	I_9	0	0	0	0	N/A	N/A	25	0	0	25
Road_4	I_10	0	0	0	3	0	32	N/A	N/A	1	36
	I_11	0	0	31	0	0	0	N/A	N/A	2	33
	I_12	22	0	0	0	0	0	N/A	N/A	1	23
Unknown		6	11	0	5	10	6	6	1	0	45
Total		108	83	105	15	167	126	69	17	36	726

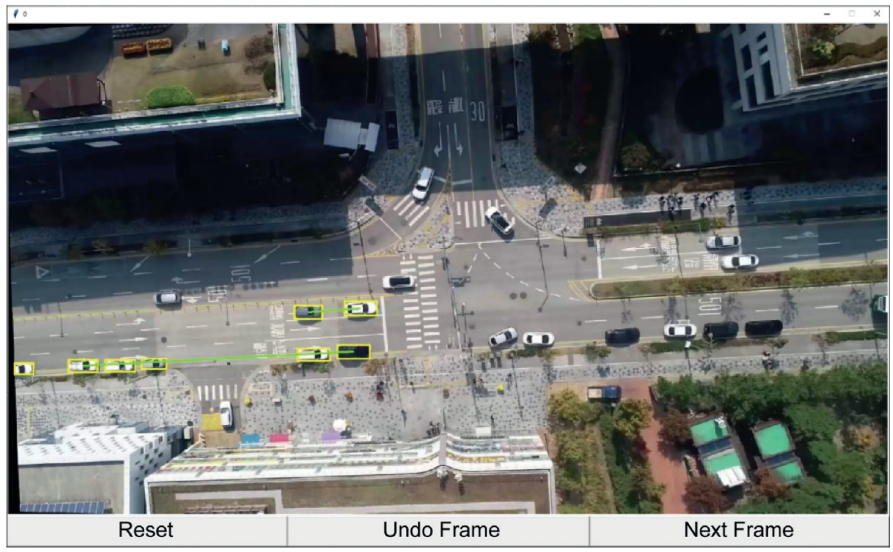


Figure 11: An example of comparing drone image for queue length derivation.

Table 5: Overall performance.

No.	Evaluation item	Number of samples (frame)	Subitems	Value	Unit	Evaluation method
1	Object detection	6804	—	99	%	Detection rate
2	Object classification	6804	Car	95	%	MAP
			Truck	87	%	MAP
			Bus	81	%	MAP
3	Trajectory location	60531	—	1.138	Meter	Mean Euclidean distance
4	Traffic volume	60531	—	4.20	Number of vehicles	RMSE
				16.41	Error rate	MAPE
5	Queue length	62205	—	3.08	Meter	RMSE
				18.28	Error rate	MAPE

5. Conclusion

In this study, we considered a method for deriving traffic information using a camera installed at an intersection to improve the monitoring system for roads. The method uses a deep-learning-based approach for image processing for vehicle detection and vehicle type classification. The method then estimates the lane-by-lane vehicle trajectories using the detected locations of vehicles. Based on the estimated vehicle trajectories, the traffic volumes of each lane-by-lane traveling direction and queue lengths of each lane were estimated. The performance of the proposed method was tested with thousands of samples according to five different evaluation criteria: vehicle detection rate, vehicle type classification, trajectory prediction, traffic volume estimation, and queue length estimation. As a result, the method shows the feasibility of collecting detailed traffic information with a camera installed at an intersection.

The proposed method has two research values. It has shown high accuracy in (1) real-time vehicle detection and classification based on deep-learning-based image processing and (2) estimating lane-by-lane vehicle trajectories by matching the detected vehicle locations with the HD map. While estimating the vehicle trajectories, this study has attempted to reduce the error of estimating the center points of the bounding boxes in the images of vehicles to ensure proper performance of the HD map-matching process. Hence, the approach of combining AI and HD map techniques is the main contribution of this study. This study shows a high chance of improving current traffic monitoring systems.

Although the proposed method has shown reasonable performance, this study is not without limitations. The error rates for both lane-by-lane traffic volume and queue length estimations are greater than 15% even though the vehicle detection showed a 99% performance, which is reasonable but not sufficient in terms of the reliability of traffic information. This is due to intermittent mismatches between the vehicle locations of the camera images and the HD map coordinates. Hence, further studies should consider enhancing the matching performance between camera image-based data and map data. Furthermore, the results of this study confirmed that the error increased with the distance between the camera and vehicle. Thus, investigating the minimum required distance between the camera and the intersection area can be a topic for future studies. In addition, for road lanes, additional research is required to develop a vehicle location correction algorithm. It is also necessary to perform training with trucks and buses to further improve the detection rate. Subsequent studies

should consider these limitations for the further development of image processing-based traffic monitoring systems.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to acknowledge Editage (http://www.editage.co.kr) for English language editing. This research was supported by the Ministry of Land, Infrastructure, and Transport (MOLIT, Korea) under the Connected and Automated Public Transport Innovation National R&D Project (Grant no. 21TLRP-B146733-04).

References

[1] A. H. F. Chow, R. Sha, and S. Li, "Centralised and decentralised signal timing optimisation approaches for network traffic control," *Transportation Research Part C: Emerging Technologies*, vol. 113, pp. 108–123, 2020.

[2] L. Adacher and M. Tiriolo, "Performance analysis of decentralized vs. centralized control for the traffic signal synchronization problem," *Journal of Advanced Transportation*, vol. 2020, Article ID 8873962, 19 pages, 2020.

[3] S. Kim, S. Tak, D. Lee, and H. Yeo, "Distributed model predictive approach for large-scale road network perimeter control," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2673, no. 5, pp. 515–527, 2019.

[4] S. C. Calvert, W. J. Schakel, and J. W. C. van Lint, "Will automated vehicles negatively impact traffic flow?" *Journal of Advanced Transportation*, vol. 2017, Article ID 3082781, 17 pages, 2017.

[5] S. Tak, J. Yoon, S. Woo, and H. Yeo, "Sectional information-based collision warning system using roadside unit aggregated connected-vehicle information for a cooperative intelligent transport system," *Journal of Advanced Transportation*, vol. 2020, Article ID 1528028, 12 pages, 2020.

[6] M. L. Tam and W. H. K. Lam, "Using automatic vehicle identification data for travel time estimation in Hong Kong using automatic vehicle identification data for travel time estimation in Hong Kong," *Transportmetrica*, vol. 4, no. 3, pp. 179–194, 2008.

[7] P. W. Wang, H. B. Yu, L. Xiao, and L. Wang, "Online traffic condition evaluation method for connected vehicles based on multisource data fusion," *Journal of Sensors*, vol. 2017, Article ID 7248189, 11 pages, 2017.

[8] J. M. Salanova Grau, E. Mitsakis, P. Tzenos, I. Stamos, L. Selmi, and G. Aifadopoulou, "Multisource data framework for road traffic state estimation," *Journal of Advanced Transportation*, vol. 2018, Article ID 9078547, 9 pages, 2018.

[9] J. W. C. Van Lint and N. J. Van der Zijpp, "Improving a travel-time estimation algorithm by using dual loop detectors," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1855, no. 1, pp. 41–48, 2003.

[10] Q. Gan, G. Gomes, and A. Bayen, "Estimation of performance metrics at signalized intersections using loop detector data and probe travel times," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 11, pp. 2939–2949, 2017.

[11] J. Tang, Y. Zou, J. Ash, S. Zhang, F. Liu, and Y. Wang, "Travel time estimation using freeway point detector data based on evolving fuzzy neural inference system," *PLoS One*, vol. 11, no. 2, Article ID e0147263, 2016.

[12] A. Roy, N. Gale, and L. Hong, "Automated traffic surveillance using fusion of doppler radar and video information," *Mathematical and Computer Modelling*, vol. 54, no. 1-2, pp. 531–543, 2011.

[13] S. L. Jeng, W. H. Chieng, and H. P. Lu, "Estimating speed using a side-looking single-radar vehicle detector," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 607–614, 2014.

[14] C. Lu and J. Dong, "Estimating freeway travel time and its reliability using radar sensor data," *Transportmetrica B: Transport Dynamics*, vol. 6, no. 2, pp. 97–114, 2018.

[15] P. Chakraborty, Y. O. Adu-Gyamfi, S. Poddar, V. Ahsani, A. Sharma, and S. Sarkar, "Traffic congestion detection from camera images using deep convolution neural networks," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2672, no. 45, pp. 222–231, 2018.

[16] A. Fedorov, K. Nikolskaia, S. Ivanov, V. Shepelev, and A. Minbaleev, "Traffic flow estimation with data from a video surveillance camera," *Journal of Big Data*, vol. 6, no. 1, p. 73, 2019.

[17] D. C. Luvizon, B. T. Nassu, and R. Minetto, "A video-based system for vehicle speed measurement in urban roadways," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, pp. 1393–1404, 2017.

[18] M. Bernas, B. Płaczek, W. Korski, P. Loska, J. Smyła, and P. Szymała, "A survey and comparison of low-cost sensing technologies for road traffic monitoring," *Sensors*, vol. 18, no. 10, p. 3243, 2018.

[19] Y. Nam and Y. C. Nam, "Vehicle classification based on images from visible light and thermal cameras," *EURASIP Journal on Image and Video Processing*, vol. 2018, no. 1, 2018.

[20] J. Kato, T. Watanabe, S. Joga, J. Rittscher, and A. Blake, "An HMM-based segmentation method for traffic monitoring movies," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1291–1296, 2002.

[21] Y. Wang, Y. Zou, H. Shi, and H. Zhao, "Video image vehicle detection system for signaled traffic intersection," in *Proceedings of the 2009 9th International Conference on Hybrid Intelligent Systems*, vol. 1, pp. 222–227, Shenyang, China, August 2009.

[22] R. Cucchiara, M. Piccardi, and P. Mello, "Image analysis and rule-based reasoning for a traffic monitoring system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 2, pp. 758–763, 1999.

[23] J. Zhou, D. Gao, and D. Zhang, "Moving vehicle detection for automatic traffic monitoring," *The IEEE Transactions on Vehicular Technology*, vol. 56, no. 1, pp. 51–59, 2007.

[24] Y. K. Ki and D. Y. Lee, "A traffic accident recording and reporting model at intersections," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, pp. 188–194, 2007.

[25] J. P. Lin and M. T. Sun, "A YOLO-based traffic counting system," in *Proceedings of the 2018 Conference on Technologies and Applications of Artificial Intelligence TAAI 2018*, pp. 82–85, Taichung, Taiwan, December 2018.

[26] C. S. Asha and A. V. Narasimhadhan, "Vehicle counting for traffic management system using YOLO and correlation filter," in *Proceedings of the 2018 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, Bangalore, India, March 2018.

[27] K. J. Kim, P. K. Kim, Y. S. Chung, and D. H. Choi, "Multi-scale detector for accurate vehicle detection in traffic surveillance data," *IEEE Access*, vol. 7, pp. 78311–78319, 2019.

[28] F. Zhang, C. Li, and F. Yang, "Vehicle detection in urban traffic surveillance images based on convolutional neural networks with feature concatenation," *Sensors*, vol. 19, no. 3, p. 594, 2019.

[29] Z. Xu, H. Shi, N. Li, C. Xiang, and H. Zhou, "Vehicle detection under UAV based on optimal dense YOLO method," in *Proceedings of the 2018 5th International Conference on Systems and Informatics ICSAI 2018*, pp. 407–411, Nanjing, China, November 2019.

[30] A. Forero and F. Calderon, "Vehicle and pedestrian video-tracking with classification based on deep convolutional neural networks," in *Proceedings of the 2019 22nd Symposium on Image, Signal Processing and Artificial Vision*, Bucaramanga, Colombia, April 2019.

[31] D. Ka, D. Lee, S. Kim, and H. Yeo, "Study on the framework of intersection pedestrian collision warning system considering pedestrian characteristics," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2673, no. 5, pp. 747–758, 2019.

[32] J. Zhang, X. Jin, J. Sun, J. Wang, and K. Li, "Dual model learning combined with multiple feature selection for accurate visual tracking," *IEEE Access*, vol. 7, pp. 43956–43969, 2019.

[33] J. Zhang, Y. Wu, W. Feng, and J. Wang, "Spatially attentive visual tracking using multi-model adaptive response fusion," *IEEE Access*, vol. 7, pp. 83873–83887, 2019.

[34] J. Zhang, X. Jin, J. Sun, J. Wang, and A. K. Sangaiah, "Spatial and semantic convolutional features for robust visual object tracking," *Multimedia Tools and Applications*, vol. 79, no. 21-22, pp. 15095–15115, 2020.

[35] X. Chen, X. Xu, Y. Yang, H. Wu, J. Tang, and J. Zhao, "Augmented ship tracking under occlusion conditions from maritime surveillance videos," *IEEE Access*, vol. 8, pp. 42884–42897, 2020.

[36] X. Chen, Z. Li, Y. Yang, L. Qi, and R. Ke, "High-resolution vehicle trajectory extraction and denoising from aerial videos," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 5, pp. 3190–3202, 2021.

[37] L. Wang, L. Zhang, and Z. Yi, "Trajectory predictor by using recurrent neural networks in visual tracking," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3172–3183, 2017.

[38] D. Koller, J. Weber, T. Huang, and J. Malik, "Towards robust automatic traffic scene analysis in real-time," in *Proceedings of 12th International Conference on Pattern Recognition*, vol. 4, Jerusalem, Israel, 1994.

[39] J. Li, S. Chen, F. Zhang, E. Li, T. Yang, and Z. Lu, "An adaptive framework for multi-vehicle ground speed estimation in airborne videos," *Remote Sensing*, vol. 11, no. 10, p. 1241, 2019.

[40] S. Seong, J. Song, D. Yoon, J. Kim, and J. Choi, "Determination of vehicle trajectory through optimization of vehicle bounding boxes using a convolutional neural network," *Sensors*, vol. 19, p. 4263, 2019.

Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 5513552, 15 pages
https://doi.org/10.1155/2021/5513552

Research Article

The Implications of Weather and Reflectivity Variations on Automatic Traffic Sign Recognition Performance

Mudasser Seraj, Andres Rosales-Castellanos, Amr Shalkamy, Karim El-Basyouny, and Tony Z. Qiu

Department of Civil and Environmental Engineering, University of Alberta, Edmonton T6G 2R3, Canada

Correspondence should be addressed to Mudasser Seraj; seraj@ualberta.ca

Received 11 January 2021; Revised 29 June 2021; Accepted 31 July 2021; Published 12 August 2021

Academic Editor: Wen LIU

Automatic recognition of traffic signs in complex, real-world environments has become a pressing research concern with rapid improvements of smart technologies. Hence, this study leveraged an industry-grade object detection and classification algorithm (You-Only-Look-Once, YOLO) to develop an automatic traffic sign recognition system that can identify widely used regulatory and warning signs in diverse driving conditions. Sign recognition performance was assessed in terms of weather and reflectivity to identify the limitations of the developed system in real-world conditions. Furthermore, we produced several editions of our sign recognition system by gradually increasing the number of training images in order to account for the significance of training resources in recognition performance. Analysis considering variable weather conditions, including fair (clear and sunny) and inclement (cloudy and snowy), demonstrated a lower susceptibility of sign recognition in the highly trained system. Analysis considering variable reflectivity conditions, including sheeting type, lighting conditions, and sign age, showed that older engineering-grade sheeting signs were more likely to go unnoticed by the developed system at night. In summary, this study incorporated automatic object detection technology to develop a novel sign recognition system to determine its real-world applicability, opportunities, and limitations for future integration with advanced driver assistance technologies.

1. Introduction

Traffic signs are used to regulate, warn, and guide traffic on roadways and facilitate coordinated road usage [1], and their placement, orientation, and visibility are crucial for road operation and safety. Given their importance, automatic recognition of roadway signs by smart transportation technology is a current research interest with numerous potential applications [2]. Automatic sign recognition could facilitate the interpretation of information received from the detected signs by assisted and autonomous driving systems. For instance, traffic operation and maintenance authorities could develop traffic sign inventories and conveniently identify traffic sign maintenance needs with the help of such automated systems [3, 4]. In addition, rapid and accurate recognition of traffic signs is important for improving traffic safety, the primary goal of Intelligent Transportation Systems and Vision Zero initiatives [5–7].

While significant effort to develop a robust sign recognition system has been made by both academics and industry practitioners [8–15], the sensitivity of the developed systems to recognize signs in varying real-world conditions is still hypothetical. Our descriptive research, then, concentrates on the question: how does the diversity of weather and reflectivity conditions of the physical world influence the recognition performance of a TSR system, given that the system is developed by incremental training resources? To address this crucial research question, our study leveraged a state-of-the-practise object detection algorithm to develop a robust TSR system that can successfully recognize a wide range of traffic signs. The objective was to measure the recognition efficiency of the developed TSR system in various weather and reflectivity conditions with the aim of assessing the implications of those variations on the recognition performance. The scope of this research is limited to identifying the changes of performance pattern due to

variations in real-world weather and reflectivity conditions resulting from an industry-level object detection system. Hence, comparing the detection performance in comparison to other state-of-the-art TSR system in standard lighting, weather and reflectivity conditions are beyond the scope of this study. Additionally, our study emphasised on detection sensitivity to TSR system training resources (i.e., images of traffic signs from favourable environmental conditions) which restrained us from analyzing asymmetrical impact due to training resources with varying lighting and environmental features.

The findings presented here will contribute to the overall body of knowledge in several ways. The key contributions of this study are listed below:

Besides intrinsic features of traffic signs (i.e., size, shape, and color), which are mostly focused on earlier studies, this study paid attention to external factors and their influence on sign identification efficiency.

Traffic sign images collected from favourable lighting and weather conditions are used as training resources in the development phase of the TSR system and expected to establish a more proficient system with gradually increased resources. The identified pattern of progression will facilitate future research by providing a benchmark for the correlation between training datasets and the expected efficiency of a TSR system that is developed with the same system architecture.

As mentioned before, the TSR system was only trained with images from favourable environmental condition which implies the proficiency of the developed system in identifying the signs in unfavourable condition without being trained for such conditions.

2. Literature Review

Researchers and practitioners have attempted to establish automatic traffic sign detection systems over the last few decades that have increased in complexity as technology has advanced. The most conventional form of research has focused on systems that attempt to extract signs from their environments for identification based on colour and shape. In terms of colour-based recognition, a valuable set of investigations used RGB space [16–18]. A clustering method in a colour space was developed by Tominaga [19] for sign detection. Ohlander et al. [20] also used a recursive region splitting method to achieve colour segmentation and applied Hue, Saturation, Intensity (HSI), and sign area. In terms of shape-based recognition, studies used the basic sign shapes of circles, triangles, or rectangles [9, 11, 21–26]. This method does overcome brightness issues found with colour-based methods. However, a significant difficulty for shape-based detection is the rotation angle and distortions of signs. All road signs in this method require a nonzero angle between the optical axis of each camera and the normal vector to the sign surface and should be as high as 30°, depending on the distance between the sign and the cameras. The complexity of automatic shape-based detection increases for signs with acute viewing angles as well as signs with torn corners and occluded parts.

Methods based on statistical machine learning and artificial intelligence (AI) have become prevalent, providing researchers with new tools to develop more efficient and reliable TSR systems using classification techniques, such as artificial neural networks (ANNs) [27, 28] k-nearest neighbour (KNN) [13, 29], support vector machine (SVM) [8, 18], and random forest [30, 31]. In 2005, Gil-Jimenez et al. [32] explored shape classification algorithms using SVMs for TSR. They found that using statistical classification methods such as SVM, colour, and shape, traffic signs can be roughly recognized. In 2006, Gao [33] did further research on shape and colour using human vision models, testing on 98 British traffic signs in various viewing conditions. The results were that the recognition rate increased to as high as 95%, especially for immobile traffic signs. In another study [34], unified visual saliency with Histograms of Oriented Gradients (HOG) featured learning for TSR. Here, the authors combined SVM with the HOG feather learning method and were able to achieve a high recognition rate. In 2010, Prisacariu et al. [35] proposed a real-time system that introduced region-based 3D tracking to single view detection, followed by adaptive boosting cascades and SVM to improve accuracy. Huang et al. [36] developed a method for detecting and recognizing speed-limit signs using only gray-level information. Hechri and Mtibaa [37] proposed and tested road sign detection using shape-filtering methods, with a multilayer, perception neural network classification model. Other studies have refined the details of real-time TSR. Sheng et al. [38] proposed treatment for recognizing signs of different colours, such as red/yellow/blue versus grayscale, and a probabilistic neural network to achieve final recognition. Li et al. [39] proposed a fuzzy shape recognizer to improve the robustness of traffic sign detection. Although it highlighted some success in sign detection and recognition, the authors emphasised the need to investigate and improve the overall performance in different weather and light conditions. However, while these studies were focused on developing TSR systems with higher accuracy through advanced techniques of machine learning and artificial intelligence, they were often silent on the implications of real-world variables on sign recognition.

A few examples of recent studies do couple the application of statistical machine learning and artificial intelligence with the influence of exogenous factors on sign detection and recognition. For instance, Sajjad et al. [40] developed a deep learning-based sign detection system as a part of an autonomous driving demonstration. Although the developed system performed well in a controlled environment, the detection and navigation accuracy are yet to be tested in real-world scenarios with complex challenges. Wang et al. [41] also developed an AI-based lightweight sign detection system that outperformed the Microsoft COCO benchmark [42]. However, the developed system was tested solely on images extracted from diverse real-world settings and not continuous detection from real-world driving condition videos that would include the added complexity from continuous sign tracking. The analysis performed by Muhammad et al. [43] on

multiple state-of-the-art approaches for sign detection from Swedish Traffic Sign Dataset [44] also suffered from similar limitations. Tabernik and Skocaj developed a convolutional neural network- (CNN-) based system capable of recognizing 200 categories of signs. While the literature demonstrated excellent performance on the tested dataset, the pair did not compare the developed TSR system's performance variations for different lighting and weather conditions. Garcia-Garrido et al. [45] tested a sign detection, classification, and tracking system under different weather and light conditions employing an algorithm designed to use a camera mounted on a vehicle's windscreen. While the authors stated that the developed TSR system was tested in different weather and reflectivity conditions, additional information regarding recognition performance in varying environmental scenarios was absent in the paper. Phu and Lwin Oo [46] introduced an RGB colour-based thresholding technique for sign detection and recognition, using an adaptive neuro-fuzzy inference system (ANFIS) to recognize different features points. Although they concluded that the system yields good results in sunny, cloudy, and rainy weather conditions, supporting analysis for this claim was not provided in the paper. Lim et al. [47] proposed a system for real-time recognition of speed limit signs in different illumination conditions using modified census transform (MCT) and support vector machine (SVM). High detection and recognition rates were obtained. However, the influence of incremental training on TSR system performance remained unaddressed. Hassaballah et al. [48] presented their conceptually similar research on detecting objects (i.e., vehicles) in adverse weather conditions. They restored visibility by improving raw image quality before object detection and tracking. The user's perspective of the testing dataset was static for all the different weather conditions collected from images. In real-world driving scenarios, the adverse weather conditions introduce added challenges of uninterrupted detection and tracking with continuously changing environments and backgrounds.

From our review, it is evident that numerous studies have been proposed to detect and recognize road traffic signs. While plenty of proposed methods rely on computer vision and artificial intelligence tools, challenges still exist in the field, including the effects of weather, reflectivity variations, limited classes of sign recognition, and human-machine interactivity. More importantly, the majority of established TSR systems have not been tested for real-world environmental challenges. To address these research gaps in sign recognition systems, we introduce a YOLO-based TSR system for detection and recognition of key regulatory and warning traffic signs in real time. Our study measured its performance in a variety of weather and reflectivity conditions and considered their influence on recognition ability. We contribute to the existing TSR system foundations by pairing it with an industry-level detection system and evaluating its performance in complex real-world scenarios. Furthermore, we developed several editions of the TSR system by training the system, introducing increasing numbers of traffic sign images to its training process. We went on to measure each edition's ability to provide direction on the necessary resources required to attain a certain level of success from TSR systems.

3. Data Collection

The data collection process played a pivotal role in our research progress as the diversity of collected data mandated the individuality of this research. In the beginning, a subset of available on-road traffic signs, coupled with signs used in earlier studies, was selected for investigation. Our initial selection criteria covered the majority of basic regulatory signs while expanding to include common warning signs. Ultimately, eight types of signs from two sign classes (i.e., regulatory and warning) were used, as listed in Table 1. Signs with different specifications (i.e., age and sheeting types) were collected from across Canada. Altogether, a total of twenty-eight signs were used for data collection. Fifteen of these signs were made of ASTM D4956 Type XI (usually known as and will be referred as diamond grade) sheeting with high reflectivity, and the remaining thirteen signs were made of Type I (usually known as and will be referred as engineering-grade) sheeting with low reflectivity. Out of twenty-eight signs, fifteen signs were new (age $\leq$1-year), while thirteen signs were three years or older. Although the collected sign inventory was diverse with respect to these features, we were unable to acquire each sign type in the different ages and sheeting types, proving to be an analysis limitation concerning reflectivity in sign recognition. Therefore, the analysis of reflectivity was performed by grouping signs into sign classes and comparing the performance of each sign class.

Once procured, the signs were installed on an access-controlled roadway segment (with a length of 750 meters in each direction) specifically built for research purposes at the University of Alberta, Edmonton, Alberta, Canada (Figure 1). A total of ten signposts were evenly spaced at designated spots on the test track, five for each direction of traffic. Each signpost was equipped with an adjustable holding mechanism to facilitate the placement and removal of traffic signs. On each round of data collection, ten signs were installed on the signposts. After installation, a vehicle equipped with a video recording camera drove along the roadway segment at 40 km/h recording the installed traffic signs. Although only ten signs were installed on the test track at any given time, all the available signs for this study were used by changing and/or moving them after each recording, thus generating different combinations for every round. For each combination of traffic signs, three recordings were made on each round to ensure the availability of superior quality videos for any given scenario. To maintain a standard recording environment, the elevations of the signs on their posts, as well as the position and angle of video camera inside the car, were kept consistent.

All videos recorded in the afternoon underwent weather impact analysis to filter for the influence of light. In total, 795 videos were recorded covering different weather and reflectivity conditions. Ninety seven (12%) of those were unusable due to exogenous factors (e.g., placement errors and obscured visuals). From the remaining videos, 15%, all with favourable conditions (i.e., daytime and sunny weather), were set aside for the image extraction required for TSR system development. The remaining videos were labelled and batched according to the weather and reflectivity conditions during recording.

Table 1: Selected traffic signs for this study.

Sign class	Sign type	Sign image
Regulatory signs	Speed limit	  
	Stop	
	Yield	
Warning signs	Speed limit change ahead	
	Signal ahead	
	Stop ahead	
	Yield ahead	
	Curve ahead	 

4. Sign Recognition System Development and Testing

In order to overcome the inherent limitations of self-created TSR systems, the search for an established system architecture based on artificial intelligence led us to YOLO. YOLO is an industry-grade object detection system that is extremely fast and accurate [49]. We chose version 3 (v3) for our study as it includes a new object classifier network that has outperformed earlier versions in different types of object detection. We then trained that classifier network to detect the traffic signs listed in Table 1 and subsequently tested its sign recognition ability in varying weather and reflectivity conditions.

Since adequate training datasets are the most important input for a deep learning approach, initial efforts went towards the extraction of video frames containing traffic signs, with the intention to gradually increase the number of images in the dataset. Each extracted image was labelled using Labellmg software [50] to extract individual regions of an image, create a bounding box, and generate annotation in a YOLOv3 compatible format, with a total of 4,445 annotations generated from the extracted frames. Once the annotations were generated, a python script was written to find corresponding images since image labelling software only develops individual annotations. Without finding the image for a specific annotation, the training process could not be executed. After developing a set of images with annotations, the images were further augmented to simulate potential variations of light, weather conditions, and physical distortions.

Using a python script, shape-based modification included rotating the captured frames at specific angles (i.e., 0°, 15°, 30°, and 45°) to imitate possible physical distortion of traffic signs that might be experienced on the video. Colours were also modified using the python script, to replicate various light scenarios by changing hue, saturation, and contrast of extracted image frames. While it would be impractical to account for all possible shape and colour-based inconsistencies experienced in real world, the augmented dataset provided some primary criterion for the algorithm to learn. Examples of data acquisition and shape and colour-based augmentation of image frames are provided in Figure 2.

Once the preprocessing of training data was finished, the images were fed into the YOLOv3 object detection system with a Darknet-53 feature extractor to develop our TSR system. YOLOv3 used a variant of Darknet [51] that originally had a 53-layer convolutional neural network (CNN) trained on ImageNet [52]. For the task of recognition, 53 additional CNN layers were stacked onto it, making it a 106-CNN layer underlying architecture for YOLOv3. As a result, the residual skip connection and upsampling features of the TSR system were enhanced. YOLOv3 performed recognition on feature maps of three different sizes at three different places in the network. Object recognition from images in different scales is a unique feature of YOLOv3, making it ideal for this study since traffic signs are small in relation to the larger image. Its recognition uses a detection kernel shaped as $1 \times 1 \times [B \times (5 + C)]$. Here, B is the number of possible predicted bounding boxes on a cell of the feature map, 5 is the four bounding box attributes and one object confidence, and C is the number of sign classes.

The first detection was made by the 82nd layer. For the first 81 layers, the image was down sampled by the network, such that the 81st layer had a stride of 32. For instance, if we had an image of 416×416 pixels, the resultant feature map would be 13×13. If one detection is made here using the 1×1 detection kernel, this gives us a detection feature map of $13 \times 13 \times 48$. The default value of $B = 3$ for YOLOv3 as it predicts 3 bounding boxes for every cell, where each bounding box specialises in detecting a certain kind of object; C is the number of classes, 11 in our case reflecting the number of detection signs. Hence, the kernel size is $1 \times 1 \times 54$ for each region on the image.

The feature map from layer 79 was subjected to a few convolutional layers before being upsampled by two-times to dimensions of 26×26. The map was then depth

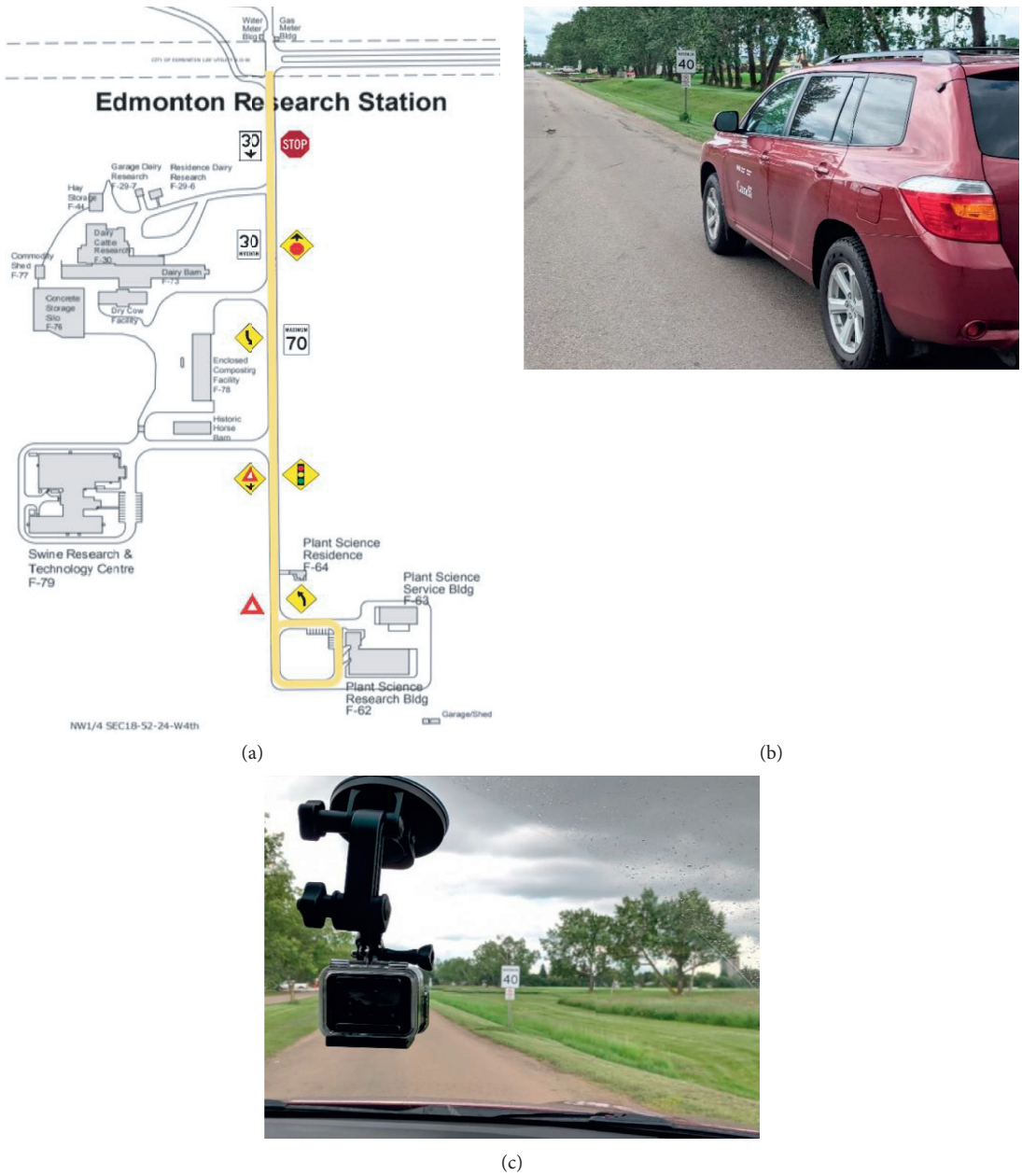


Figure 1: (a) Plan of data collection test track. (b) Vehicle. (c) Camera used for recording videos of traffic signs.

concatenated with the feature map from layer 61 and subjected to a few 1×1 kernel convolutional layers to fuse the features. The second detection was made on the 94^{th} layer, yielding a detection feature map of $26 \times 26 \times 255$. A similar procedure was followed where the feature map from layer 91 was subjected to convolutional layers before being depth concatenated with a feature map from layer 36. Like before, a few 1×1 kernel convolutional layers followed to fuse the information. The final detection was made at 106^{th} layer, yielding a feature map of $52 \times 52 \times 255$. YOLOv3 used nine anchor boxes, three for each image scale. Training YOLO on a custom dataset required using K-means clustering to generate these nine anchors. The anchors were arranged in descending order of dimension, assigning the three biggest anchors for the first scale, the next three for the second scale, and the last three for the third scale.

Training the TSR system began by feeding the annotated training images into the YOLOv3 network. Training images used for all versions of the TSR system were collected from favourable lighting (i.e., daylight) and weather (i.e., sunny) conditions. Continuous monitoring of average loss for each epoch was made from the beginning of the training process. The objective of observing the average loss value was to stop the training after it reached a certain threshold or a point where the loss value became stationary, and it could be assumed that the network had converged. The training usually converged at a loss rate of 0.03 over a certain number

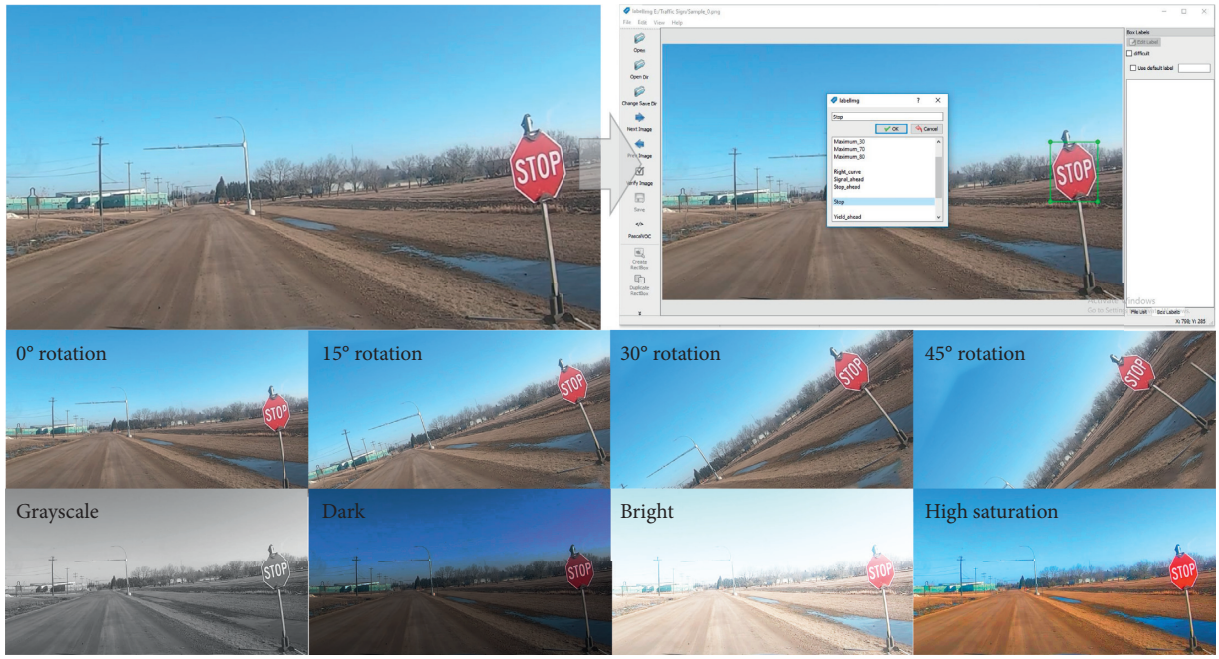


Figure 2: Different stages of data preprocessing.

of epochs (130 – 200 epochs). The training process involved iteratively updating the weights of the neural network based on the number of mistakes made on the training dataset. Weights represent the strength of the connection between neuron units, which helped to determine how much influence the input would have on the output. Every neuron of the network was given an input value and a bias value and was then multiplied by a weight value that was adjusted during each iteration of training.

Once the training task was completed by attaining a target average loss rate of 0.03, the most recent weight generated by the system was executed on Darknet to analyse the testing video dataset. These recorded videos contained the chosen traffic signs in different light, weather, and reflectivity scenarios. In order to obtain labelled videos, screenshots were captured during analysis by the developed TSR system. Additionally, a .csv file containing the information pertaining to the recognized traffic signs and image frames of labelled traffic signs from the analysed video was automatically stored in a secure server after each test video analysis was completed. An overview of the entire sign recognition system development and testing process is summarized in Figure 3.

In this study, five versions of the TSR system were developed by gradually increasing the number of training images, allowing us to explore the impact of training datasets on recognition performance. Each version, therefore, had a different training dataset but maintained the same parameters as outlined in Table 2. Training datasets for each version were prepared from extracted images recorded in bright, sunny conditions to maintain a standard training environment. In each version of the TSR system, 20% training images contained traffic signs and reminder of the training data were images without traffic signs. The number of annotated and augmented images of traffic signs used for the training and validation of each system version is listed in Table 2.

5. Analysis of TSR System Performance and Progress

To perform analysis of system performance, a set of parameters was chosen from the literature. A combination of these parameters was used exclusively to evaluate the identification accuracy of different versions of the developed TSR system. The recognition performance of each sign type was evaluated using three parameters: precision, recall, and $F1$-score. To measure these parameter values, the recognition criterion was defined by measuring the number of True Positives, False Positives, and False Negatives on each dataset. The analysis was based on the following definitions of the parameters.

True Positive: TSR system successfully recognized (i.e., identified and classified) the presence of a test sign

False Positive: TSR system incorrectly recognized (i.e., identified and/or classified) the presence of a test sign

False Negative: TSR system failed to recognize (i.e., identify and recognize) the presence of a test sign

Using the above definitions, three key parameter values were calculated, and the following definitions were used to determine the parameter values:

Precision is the fraction of correct recognition instances out of total successful recognitions (equation (1)).

Recall is the fraction of correct recognition instances retrieved over total expected recognitions (equation (2)).

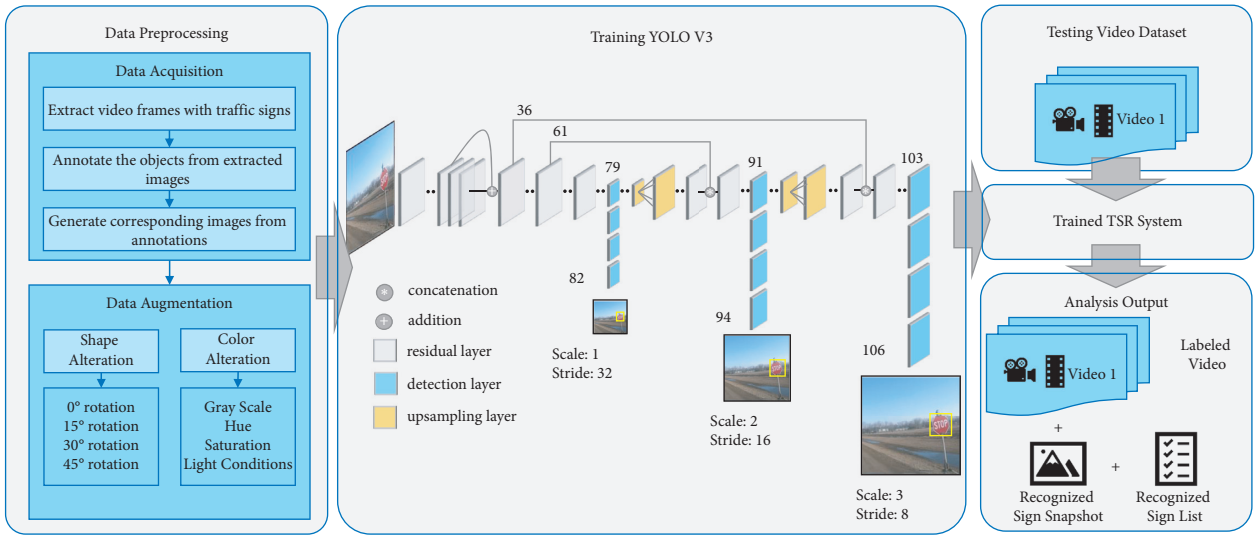


Figure 3: Flow chart of the developed TSR system.

Table 2: Information on training traffic sign image datasets for each version of the TSR system.

Groups	Number of images				
	Ver. 1	Ver. 2	Ver. 3	Ver. 4	Ver. 5
Max. 30 km/h	264	808	1594	1978	2232
Max. 70 km/h	120	502	750	1146	1408
Max. 80 km/h	291	963	1673	2257	2435
Stop	622	1388	2512	3732	4454
Yield	1272	3178	4408	5476	6046
Max. 30 km/h ahead	388	984	1578	2006	2192
Signal ahead	686	1724	2722	3232	3482
Stop ahead	932	2038	2962	4370	5014
Yield ahead	310	550	852	1192	1456
Right curve ahead	848	2196	4072	5138	5726
Left curve ahead	560	1078	1634	2110	2264
Annotated and unannotated image	5054	10231	18258	23215	26714
Training and validation annotated image	6293	15409	24757	32637	36709

$F1$-score is the weighted average of precision and recall. Since this takes both False Positives and Negatives into account, the $F1$-score provides a more useful interpretation of recognition performance (equation (3)).

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}, \quad (1)$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}, \quad (2)$$

$$F1 - \text{score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (3)$$

6. Implications of Weather Variations on TSR System Performance

The automatic sign recognition system's competence when introducing varying weather conditions was analysed by testing the videos through the established TSR system. Out of 370 videos considered (based on lighting conditions), 193 videos were recorded in varying weather conditions, but consistent daylight was chosen to keep results clear of any impact from variations in light conditions. As already outlined, the comparison of recognition performance was between fair and inclement weather conditions. Fair weather was represented on 104 videos among the tested datasets. Figure 4(a) primarily compared the recall performance of tested sign types in incremental versions of the developed TSR system for two types of weather scenarios, inclement weather and fair weather. Each horizontal bar represents the recall value of a specific sign for a specific version of the TSR system. The incremental intensity of green/blue color represents the higher version of the TSR system. For instance, the bottom-most horizontal bar of Figure 4(a) illustrated the recall value of Speed Limit signs for version 1 of the TSR system, which was much higher in fair weather (57.43%) than in inclement weather (17.77%). Table 3 outlines the version-specific parameter values irrespective of sign types. Since the impact of weather conditions was not clear from this illustration, we carried out several significance tests on the evaluation parameters.

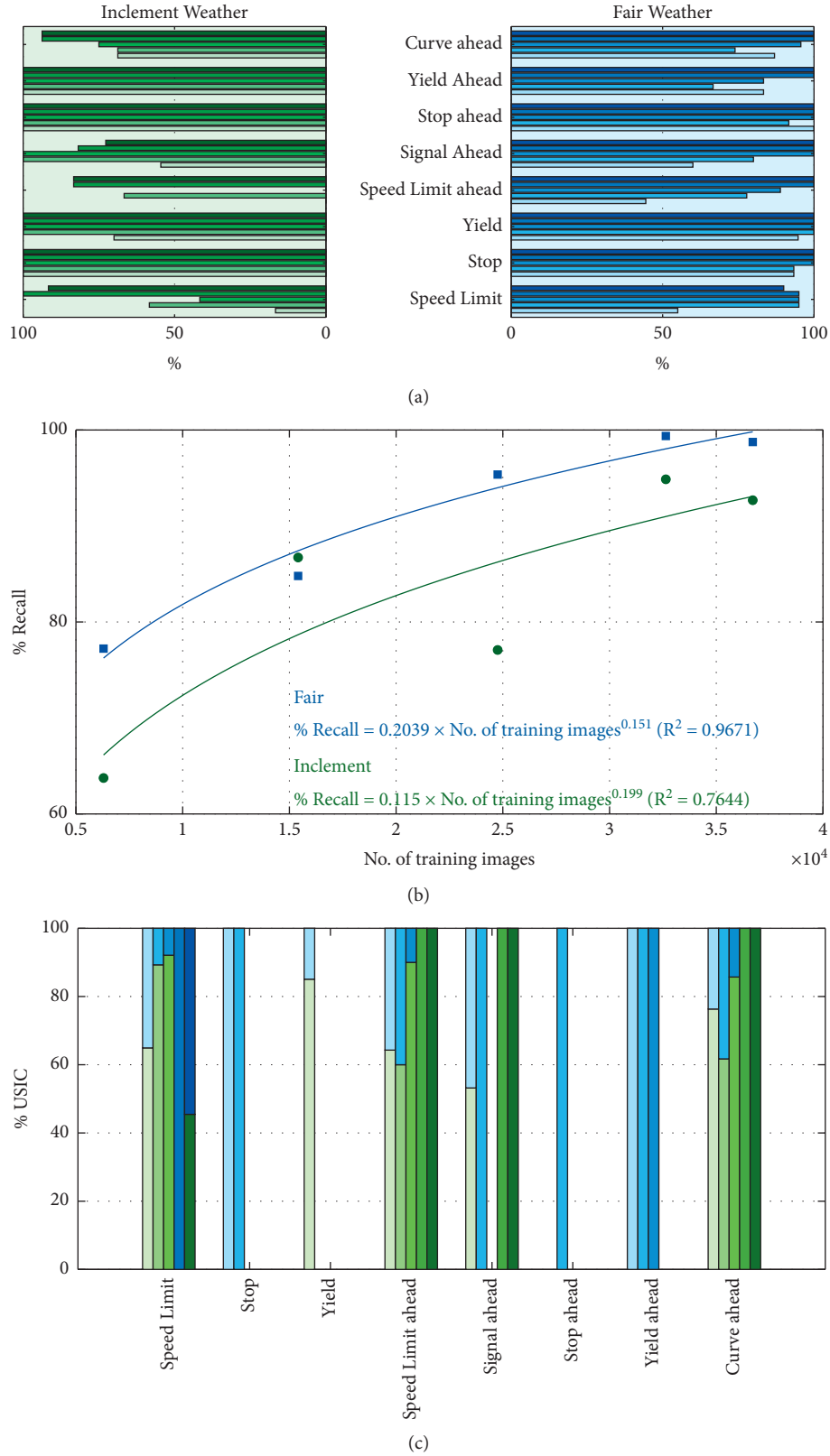


Figure 4: (a) Comparison of recall metric on studied weather conditions. (b) Progression of average recall values with an increasing number of training images. (c) Unidentified signs proportion of compared weather scenarios.

Table 3: Version-specific parameter values (i.e., mean and standard deviation) at different weather conditions.

Ver.	Precision		Recall		*F*1-score	
	Fair	Inclement	Fair	Inclement	Fair	Inclement
1	90.98% (7.15%)	80.14% (5.47%)	77.23% (20.98%)	63.75% (38.49%)	83.54% (12.67%)	71.01% (8.87%)
2	87.52% (10.68%)	75.18% (8.45%)	84.79% (11.80%)	86.72% (18.57%)	86.13% (13.21%)	80.54% (13.86%)
3	88.05% (13.98%)	67.70% (14.92%)	95.36% (6.24%)	77.08% (37.47%)	91.56% (10.63%)	72.09% (16.55%)
4	81.49% (16.16%)	58.19% (18.09%)	99.38% (1.77%)	94.86% (7.89%)	89.55% (5.19%)	72.13% (24.40%)
5	71.23% (24.76%)	40.02% (23.48%)	98.75% (3.54%)	92.68% (10.01%)	82.76% (8.93%)	55.90% (11.81%)

Significance test (two-sample *t*-test) results found that the recall parameter in fair weather was significantly higher, at 95% confidence level, than in inclement weather conditions for versions 1 and 2 of the TSR system. Based on the chi-square contingency table test, the precision metric was found to be significantly different in versions 1 through 3, whereas an insignificant difference was observed on the *F*1-score for versions 3 through 5. Hence, it can be stated that the devised TSR system did not experience significant performance variations due to weather on versions that were trained with more than 20,000 training and validation images.

Additional analysis of the average recall metric of different versions revealed a pattern of recognition performance progression as the number of images increased. This pattern revealed that the system required a higher number of images to be trained when faced with scenarios such as imperfect conditions (i.e., inclement weather) to match the performance for perfect conditions. For instance, to attain an 80% recall on signs in inclement weather, the TSR system required training on more than 17,000 images. By contrast, approximately 9000 training images would be adequate to attain a similar performance for signs in fair weather conditions. Finally, the analysis on the percent of unidentified signs in imperfect conditions (%USIC) revealed that the majority were set within inclement weather conditions (average = 56.46% and standard deviation = 41.34%). With the developed versions of the TSR system, the proportions of unidentified signs were evenly distributed between fair and inclement weather (Figure 4(c)), and several sign types were completely identified with the higher numbered versions, irrespective of weather conditions.

Since different inclement weather types bring different identification challenges, the TSR system performance was further analysed based on both inclement weather types (i.e., cloudy and snowy) compared to 89 videos. Figure 5(a) showed the comparative performance of recall parameters by the developed TSR systems in cloudy and snowy weather conditions. Figures 5(b) and 5(c) illustrate sample identification snapshots of one sign in cloudy and snowy weather conditions, respectively. The average recall value in cloudy weather conditions (84.11%) was slightly higher than snowy weather conditions (81.93%), considering all versions of the TSR system. Most of the signs showed similar performance in both inclement weather conditions, except signs with white backgrounds (i.e., speed limit signs and speed limit ahead signs). On average, identification of speed limit signs was 9.43% higher in cloudy weather conditions. Similarly, speed limit ahead signs in cloudy weather conditions experienced 5.06% higher identification rates.

7. Implications of Sign Reflectivity Variations on TSR System Performance

The analysis of sign reflectivity was designed to comprehensively consider the reflectivity factors that influence recognition performance. Hence, the signs were initially compared based on two sheeting categories, namely, engineering grade and diamond grade. Each sheeting type was further divided into two lighting conditions, daytime and night-time, presuming lighting conditions would play a significant role in detectability. Finally, two more categories for sign age were also taken into account, i.e., signs less than one year old and signs three years or older, presuming that reduction in reflectivity accumulates over the year due to weathering, natural abrasion, and other factors. The goal in this portion of the analysis was to establish the effect of these factors on the TSR system's accuracy.

The classification of the tests for this section was not made sign-specific but was rather classed in a higher-order due to the lack of availability of some specific signs in one of the two sheeting types or different age groups. As a result, signs were divided into two classes, namely, warning signs (e.g., curve ahead and signal ahead) and regulatory signs (e.g., stop and speed limit). Figure 6 shows the distribution of data samples in the testing dataset that contained characteristics related to sheeting type, lighting conditions, and sign age. The two sheeting types selected for this study possess considerably different reflectivity features. Engineering grade or Type I sheeting typically meets the requirements of ASTM D4956 and contains some basic reflectivity properties. On the other hand, diamond grade or Type XI sheeting is designed to reflect close to 50% of the available light to the driver, enabling them to better recognize signs and at a greater distance [53]. Lighting conditions were considered for the second level of reflectivity factors. All the videos for daytime lighting conditions were collected between 10 am to 4 pm, and night-time data were collected from 1 hour after sunset till 10 pm. Finally, the signs were sorted based on two predominant age groups of available signs.

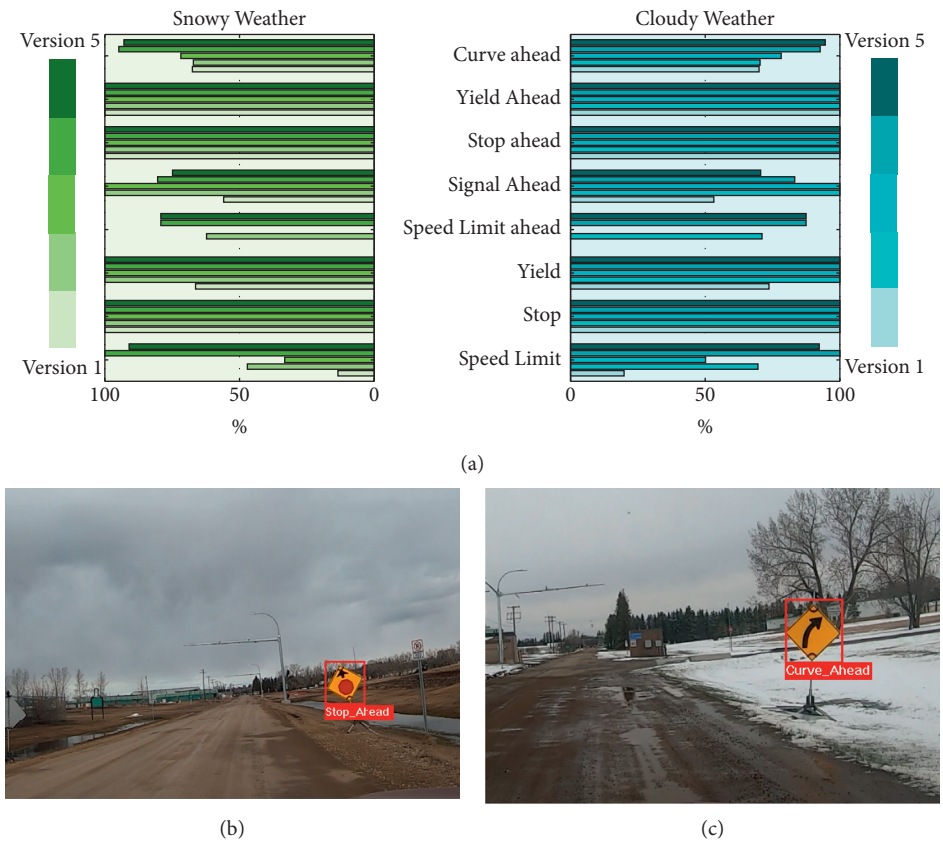


Figure 5: (a) Performance comparison in studied inclement weather scenarios and sample sign identification by the TSR system in (b) cloudy and (c) snowy weather.

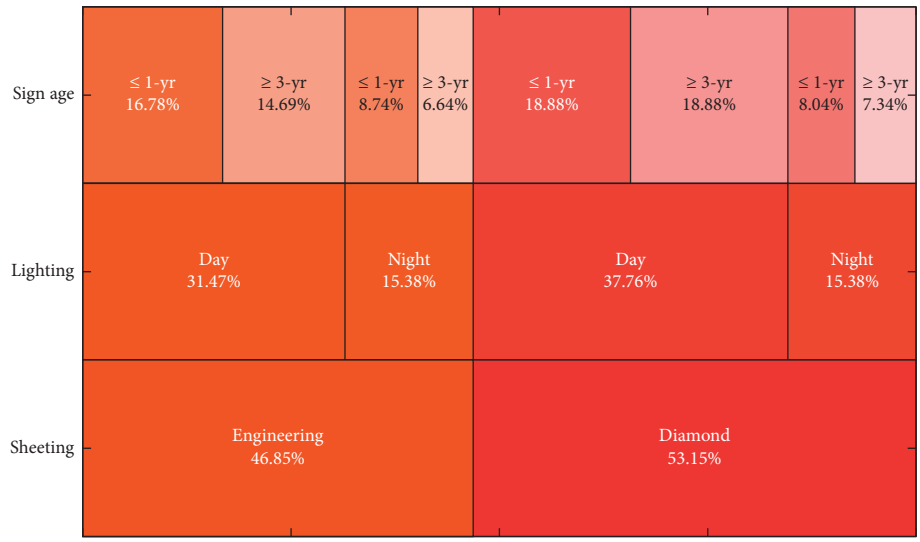


Figure 6: Share of samples according to sign age, sheeting type, and lighting conditions.

The performed analysis is summarized in Figure 7, which shows enhanced performance for the newer and diamond grade signs in night conditions. Figure 7(a) shows a comparison of recall parameters for both sheeting types obtained from different versions of the developed TSR system. A significance test on the recall metric for this part of the analysis showed a significant difference at a 95% confidence level for all versions of the developed TSR system, irrespective of sign class. Further analysis was performed to associate the influence of lighting on sign recognition for the two distinct sheeting types. In general, diamond grade signs were found to be similarly identifiable as engineering-grade signs in daylight (Figure 7(b)).

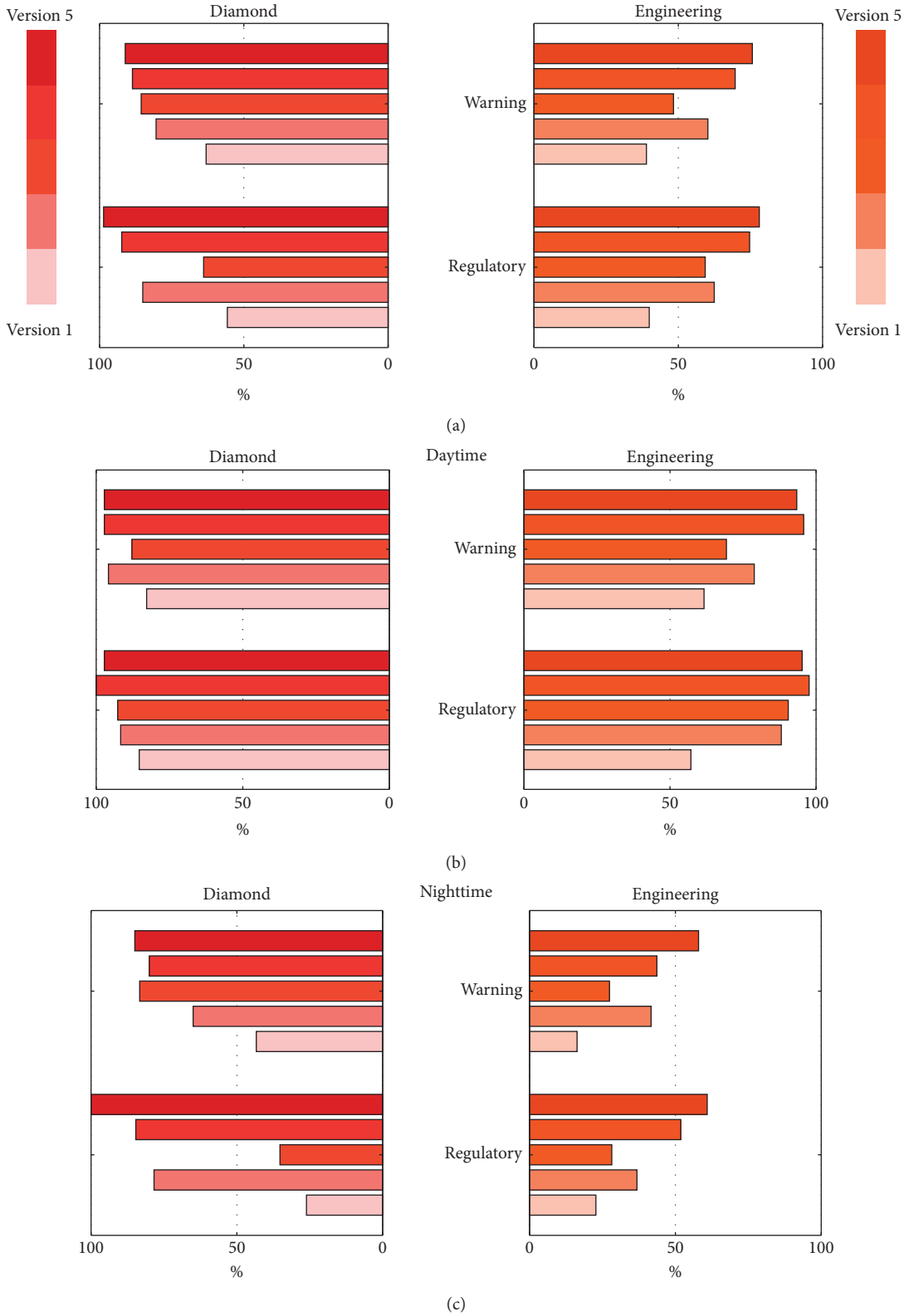


FIGURE 7: Recall parameter comparison between (a) sheeting types, (b) sheeting types with daytime lighting, and (c) sheeting types with night time lighting.

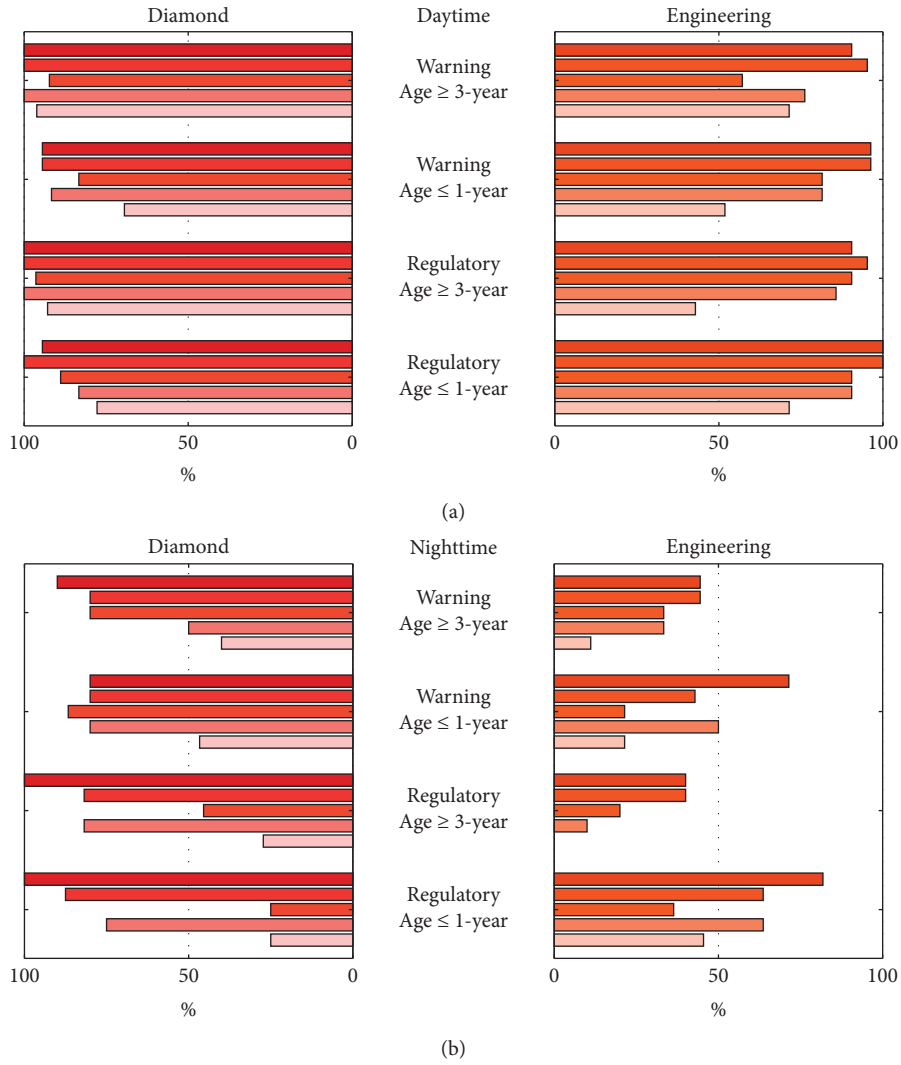


Figure 8: Recall parameter comparison between (a) sheeting types in daytime lighting and (b) sheeting types in night time lighting with different age groups.

Version 1 (Z value = 2.28) for regulatory signs and version 1 (Z value = 2.23), 2 (Z value = 2.01), and 3 (Z value = 2.08) for warning signs showed significantly higher levels of recognition for diamond grade signs during daytime. However, at night, diamond grade signs showed significantly enhanced recognizability for both sign classes in all versions (Figure 7(c)).

Figures 8 outlines daytime and night-time performance, respectively, via sign class-specific recall values in each version of the developed TSR system for the two sheeting type scenarios under scrutiny and further divided by sign age. At first glance, the daylight recall results seem to outperform night-time results, although both types of sheeting seem to produce a comparable outcome with slight favouring of diamond grade signs. Generally, the significance test results showed a significant difference between the two sheeting types, irrespective of sign class and age, for an earlier version of the TSR system at daylight condition (Figure 8(a)). More evolved forms of the TSR system varied less in responses to sign age. A striking difference in sign recognition performance was observed at night. For night-time samples (Figure 8(b)), the diamond grade signs provided all versions with a significantly higher recall rate. The comparison between different ages of engineering-grade signs showed significantly higher levels of recognition of new signs (age ≤1 year) as compared to older signs (age ≥3 years) in both sign classes (i.e., regulatory and warning). On the contrary, sign recognition was less sensitive to age for both sign classes when using diamond grade sheeting. This part of the analysis proved beyond a reasonable doubt that sheeting types of signs could play a significant role in recognition by the TSR system, exclusively during the night. Furthermore, older engineering-grade signs were more likely to be missed by a well-developed TSR system at night, in comparison to new engineering-grade signs, diamond grade signs of both age groups, and sign images taken in daylight.

8. Conclusion and Future Research

This study has contributed to the development of a new, robust automatic TSR system through the unique integration of industry-ready technology with an experimental

investigation of the TSR system's limitations and opportunities. The primary objective of the research was to study the influence of weather and reflectivity variations on the TSR system's performance, particularly as the system was improved consistently through the gradual increase in the number of training images. The results from the recognition performance analysis found that the impact of reflectivity conditions was far more significant than that of weather variations. However, scenarios with high reflectivity, fair weather, and generous lighting proved to perform better than their counterparts. Comprehensive significance testing was conducted on the evaluation parameters to identify statistically significant differences between the compared scenarios. This study also revealed that, even if the TSR system was trained with sign images from amenable natural conditions, it could attain a relatively comparable level of recognition based on imperfect real-world conditions. Furthermore, this research provides a benchmark for the resources required to train a TSR system of similar architecture with a specific level of accuracy. Our study not only developed a real-time TSR system with the capability of recognizing several sign types situated in real-world scenarios with varying environmental conditions but it also specified the scope and restrictions of the established system.

While this study focused primarily on the performance variations caused by changes of two factors (i.e., weather and reflectivity), the implications of other factors such as recognition distances, deterioration due to damage, and other exogenous variations will be explored in future studies. We believe that this study will assist in the sustainable and consistent growth of TSR system development as a part of advanced driver assistance systems. In the broader field, interested industry partners can work towards overcoming the identified limitations of TSR systems, while researchers can identify additional applications of this technology to improve traffic operation and the safety of all road users.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Disclosure

The contents of this paper reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of Alberta Innovates and 3M Canada.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors acknowledge the contributions by Kushal Hosahalli Mahalingaiah, Md. Nafize Sadik, and Soumik Das in developing the sign recognition system. This research work was jointly supported by 3M Canada, and Alberta Innovates.

References

[1] Federal Highway Administration, *Manual on Uniform Traffic Control Devices for Streets and Highways*, Federal Highway Administration, Washington, D.C., USA, 2012.

[2] U.S. Department of Transportation, *Preparing for the Future of Transportation: Automated Vehicle 3.0*, U.S. Department of Transportation, Washington, D.C., USA, 2018.

[3] L. Yang, K. Kwon, K. Moon, S. Lee, and S. Kwon, "Broken traffic sign recognition based on local histogram matching," in *Proceedings of the 2012 Computing, Communications and Applications Conference*, Hong Kong, China, January 2012.

[4] S. Maldonado-Bascón, S. Lafuente-Arroyo, P. Siegmann, H. Gómez-Moreno, and F. J. Acevedo-Rodríguez, "Traffic sign recognition system for inventory purposes," in *Proceedings of the 2008 IEEE Intelligent Vehicles Symposium*, pp. 590–595, Eindhoven, Netherlands, June 2008.

[5] B. Johansson, "Road sign recognition from a moving vehicle," Master thesis, Uppsala University, Uppasala, Swedan, 2002.

[6] B. Höferlin and K. Zimmermann, "Towards reliable traffic sign recognition," in *Proceedings of the 2009 IEEE Intelligent Vehicles Symposium*, pp. 324–329, Xi'an, China, June 2009.

[7] J. Abele, C. Kerlen, S. Krueger, H. Baum, T. Geißler, and S. Grawenhoff, "Exploratory study on the potential socio-economic impact of the introduction of intelligent safety systems in road vehicles," Final report, Publications Office of the European Union, Luxembourg, 2005.

[8] J. Greenhalgh and M. Mirmehdi, "Real-time detection and recognition of road traffic signs," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1498–1506, 2012.

[9] W. Gang, K. S. Ishwar, and S. Susanta, "Traffic sign detection and recognition for safe driving," *Intelligent Engineering Systems Through Artificial Neural Networks*, vol. 9, pp. 821–826, 1999.

[10] H. Guan, W. Yan, Y. Yu, L. Zhong, and D. Li, "Robust traffic-sign detection and classification using mobile LiDAR data with digital images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 5, pp. 1715–1724, 2018.

[11] J. Miura, T. Kanda, S. Nakatani, and Y. Shirai, "An active vision system for on-line traffic sign recognition," *IEICE - Transactions on Info and Systems*, vol. 85, no. 11, pp. 1784–1792, 2002.

[12] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: the German traffic sign detection benchmark," in *Proceedings of the 2013 International Joint Conference on Neural Networks (IJCNN)*, Dallas, TX, USA, August 2013.

[13] M. Gu and Z. Cai, "Traffic sign recognition using dual tree-complex wavelet Transform and 2D independent component analysis," in *Proceedings of the 10th World Congress on Intelligent Control and Automation*, Beijing, China, July 2012.

[14] S. Maldonado-Bascon, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno, and F. Lopez-Ferreras, "Road-sign detection and recognition based on support vector machines," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, pp. 264–278, 2007.

[15] M. Soilán, B. Riveiro, J. Martínez-Sánchez, and P. Arias, "Traffic sign detection in MLS acquired point clouds for geometric and image-based semantic inventory," *ISPRS*

Journal of Photogrammetry and Remote Sensing, vol. 114, pp. 92–101, 2016.

[16] H. Kamada, S. Naoi, and T. Gotoh, "A compact navigation system using image processing and fuzzy control," in *IEEE Proceedings on Southeastcon*, New Orleans, LA, USA, April 1990.

[17] R. Janssen, W. Ritter, F. Stein, and S. Ott, "Hybrid approach for traffic sign recognition," in *Proceedings of the Intelligent Vehicles'93 Symposium*, Tokyo, Japan, July 1993.

[18] H. Gomez-Moreno, S. Maldonado-Bascon, P. Gil-Jimenez, and S. Lafuente-Arroyo, "Goal evaluation of segmentation algorithms for traffic sign recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 4, pp. 917–930, 2010.

[19] S. Tominaga, "Color image segmentation using three perceptual attributes," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 628–630, Miami Beach, FL, USA, June 1986.

[20] R. Ohlander, K. Price, and D. R. Reddy, "Picture segmentation using A recursive region splitting method," *Computer Graphics and Image Processing*, vol. 8, no. 3, pp. 313–333, 1978.

[21] D. M. Gavrila, "Traffic sign recognition revisited," in *Proceedings of the Mustererkennung 1999: 21st DAGM Symposium*, Bonn, Germany, September 1999.

[22] C.-Y. Chiung-Yao Fang, S.-W. Sei-Wang Chen, and C.-S. Chiou-Shann Fuh, "Road-sign detection and tracking," *IEEE Transactions on Vehicular Technology*, vol. 52, no. 5, pp. 1329–1341, 2003.

[23] X. Gao, D. Shaposhnikov, V. Gusakova, K. Hong, and L. Podladchikova, "The foveal system for traffic signs recognition (FOSTS)," in *Proceeding of the 14 International Conference on Neurocybernetics*, Rostov-on-Don, Russia, 2005.

[24] G. Bae, J. Ha, J. Y. Jeon, S. Y. Jo, and H. Jeong, "LED traffic sign detection using rectangular hough transform," in *Proceedings of the 2014 International Conference on Information Science & Applications (ICISA)*, Seoul, Republic of Korea, May 2014.

[25] M. A. Garcia-Garrido, M. Ocana, D. F. Llorca, M. A. Sotelo, E. Arroyo, and A. Llamazares, "Robust traffic signs detection by means of vision and V2I communications," in *Proceedings of the 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Washington, DC, USA, October 2011.

[26] R. Belaroussi and J. Tarel, "Angle vertex and bisector geometric model for triangular road sign detection," in *Proceedings of the 2009 Workshop on Applications of Computer Vision (WACV)*, Snowbird, UT, USA, December 2009.

[27] S. KumarSaha, D. Chakraborty, and M. Al-Amin Bhuiyan, "Neural network based road sign recognition," *International Journal of Computer Applications*, vol. 50, no. 10, pp. 35–41, 2012.

[28] R. Rajesh, K. Rajeev, K. Suchithra, V. P. Lekhesh, V. Gopakumar, and N. K. Ragesh, "Coherence vector of oriented Gradients for traffic sign recognition using neural networks," in *Proceedings of the 2011 International Joint Conference on Neural Networks*, San Jose, CA, USA, July-August 2011.

[29] F. Zaklouta and B. Stanciulescu, "Warning traffic sign recognition using a HOG-based K-d tree," in *Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IV)*, Baden-Baden, Germany, June 2011.

[30] A. Z. Kouzani, "Road-sign identification using ensemble learning," in *Proceedings of 2007 IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey, June 2007.

[31] J. Greenhalgh and M. Mirmehdi, "Traffic sign recognition using MSER and random forests," in *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, Bucharest, Romania, August 2012.

[32] P. Gil-Jiménez, S. Lafuente-Arroyo, S. Maldonado-Bascón, and H. Gómez-Moreno, "Shape classification algorithm using support vector machines for traffic sign recognition," in *Proceedings of the 8th international conference on Artificial Neural Networks: computational Intelligence and Bioinspired Systems*, Barcelona, Spain, June 2005.

[33] X. W. Gao, L. Podladchikova, D. Shaposhnikov, K. Hong, and N. Shevtsova, "Recognition of traffic signs based on their colour and shape features extracted using human vision models," *Journal of Visual Communication and Image Representation*, vol. 17, no. 4, pp. 675–685, 2006.

[34] Y. Xie, L. Liu, C. Li, and Y. Qu, "Unifying visual saliency with HOG feature learning for traffic sign detection," in *Proceedings of the 2009 IEEE Intelligent Vehicles Symposium*, Xi'an, China, June 2009.

[35] V. A. Prisacariu, R. Timofte, K. Zimmermann, I. Reid, and L. Van Gool, "Integrating object detection with 3D tracking towards a better driver assistance system," in *Proceedings of the 2010 20th International Conference on Pattern Recognition*, Istanbul, Turkey, August 2010.

[36] Y. Huang, Y. Le, and F. Cheng, "A method of detecting and recognizing speed-limit signs," in *Proceedings of the 2012 Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Piraeus-Athens, Greece, July 2012.

[37] A. Hechri and A. Mtibaa, "Lanes and road signs recognition for driver assistance system," *IJCSI international journal of computer science*, vol. 8, no. 6, 2011.

[38] Y. Sheng, K. Zhang, C. Ye, C. Liang, and J. Li, "Automatic detection and recognition of traffic signs in stereo images based on features and probabilistic neural networks," in *Proceedings of the SPIE Photonics Europe*, Strasbourg, France, April 2008.

[39] L. Li, J. Li, and J. Sun, "Robust traffic sign detection using fuzzy shape recognizer," in *Proceedings of the Sixth International Symposium on Multispectral Image Processing and Pattern Recognition*, M. Ding, B. Bhanu, F. M. Wahl, and J. Roberts, Eds., p. 74960Z, Yichang, China, November 2009.

[40] M. Sajjad, M. Irfan, K. Muhammad et al., "An efficient and scalable simulation model for autonomous vehicles with economical hardware," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1718–1732, 2021.

[41] Z. Wang, J. Wang, Y. Li, and S. Wang, "Traffic sign recognition with lightweight two-stage model in complex scenes," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2020.

[42] T.-Y. Lin, M. Maire, S. Belongie et al., "Microsoft COCO: common objects in context," in *Proceedings of the 13th European Conference on Computer Vision -- ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., pp. 740–755, Springer International Publishing, Zurich, Switzerland, September 2014.

[43] K. Muhammad, A. Ullah, J. Lloret, J. D. Ser, and V. H. C. de Albuquerque, "Deep learning for safe autonomous driving: current challenges and future directions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4316–4336, 2021.

[44] F. Larsson and M. Felsberg, "Using fourier descriptors and spatial models for traffic sign recognition," in *Proceedings of the 17th Scandinavian Conference on Image Analysis*, A. Heyden and F. Kahl, Eds., pp. 238–249, Springer, Ystad, Sweden, May 2011.

[45] M. A. Garcia-Garrido, M. A. Sotelo, and E. Martin-Gorostiza, "Fast traffic sign detection and recognition under changing lighting conditions," 2006.

[46] K. T. Phu and L. Lwin Oo, "Traffic sign recognition system using feature points," in *Proceedings of the 2018 12th International Conference on Research Challenges in Information Science (RCIS)*, pp. 1–6, Nantes, France, May 2018.

[47] K. Lim, T. Lee, C. Shin, S. Chung, Y. Choi, and H. Byun, "Real-time illumination-invariant speed-limit sign recognition based on a modified census transform and support vector machines," in *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication-ICUIMC'14*, pp. 1–5, Siem Reap, Cambodia, January 2014.

[48] M. Hassaballah, M. A. Kenk, K. Muhammad, and S. Minaee, "Vehicle detection and tracking in adverse weather using a deep learning framework," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4230–4242, 2021.

[49] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018.

[50] Tzutalin. Labellmg. Git Code. https://github.com/tzutalin/labelImg. Accessed Jun. 6, 2018.

[51] J. D. Redmon, "Open source neural networks in C," 2018, https://pjreddie.com/darknet/.

[52] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," 2009.

[53] A. Halstuch and Y. Yitzhaky, "Properties of light reflected from road signs in active imaging," *Applied Optics*, vol. 47, no. 22, p. 4031, 2008.

Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 8153474, 15 pages
https://doi.org/10.1155/2021/8153474

Research Article

Moving Camera-Based Object Tracking Using Adaptive Ground Plane Estimation and Constrained Multiple Kernels

Tao Liu [1,2,3] **and Yong Liu** [1,2,3]

[1] *Beijing Key Laboratory of Network System Architecture and Convergence, Beijing University of Post and Telecommunications, Beijing, 100876, China*
[2] *Beijing Laboratory of Advanced Information Networks, Beijing University of Post and Telecommunications, Beijing 100876, China*
[3] *School of Information and Communication Engineering, Beijing University of Post and Telecommunications, Beijing 100876, China*

Correspondence should be addressed to Tao Liu; tony6@uw.edu

Received 22 April 2021; Revised 7 July 2021; Accepted 13 July 2021; Published 21 July 2021

Academic Editor: Wen Liu

Moving camera-based object tracking method for the intelligent transportation system (ITS) has drawn increasing attention. The unpredictability of driving environments and noise from the camera calibration, however, make conventional ground plane estimation unreliable and adversely affecting the tracking result. In this paper, we propose an object tracking system using an adaptive ground plane estimation algorithm, facilitated with constrained multiple kernel (CMK) tracking and Kalman filtering, to continuously update the location of moving objects. The proposed algorithm takes advantage of the structure from motion (SfM) to estimate the pose of moving camera, and then the estimated camera's yaw angle is used as a feedback to improve the accuracy of the ground plane estimation. To further robustly and efficiently tracking objects under occlusion, the constrained multiple kernel tracking technique is adopted in the proposed system to track moving objects in 3D space (depth). The proposed system is evaluated on several challenging datasets, and the experimental results show the favorable performance, which not only can efficiently track on-road objects in a dashcam equipped on a free-moving vehicle but also can well handle occlusion in the tracking.

1. Introduction

Currently, video-based traffic surveillance plays an important role in intelligent transportation systems (ITSs). And as more and more people use the dashcam during driving, video analysis based on dascam has thus become a very important research area, and object tracking such as pedestrians and vehicles is a crucial and unavoidable task in this field. By tracking pedestrians or vehicles, their movement trajectories can be collected in the video for advanced analysis, such as human or vehicle flow estimation, collision avoidance of abnormal behavior, and criminal tracking. Therefore, researchers are motivated to develop an effective tracking system, which not only can track objects in the scene but also is able to collect the information for higher-level analysis.

Tracking vehicle and pedestrian in moving cameras is quite challenging due to several reasons. First, the appearance of these objects may change greatly due to nonrigid deformation, different viewing perspectives, and other visual attributes. Second, frequent occlusion by other objects in the scene will cause severe identity switches. Last but not least, object tracking in moving camera is more challenging than that in static cameras, because of the combined effects of rapidly changing lighting conditions, blur, and the issues mentioned above. Moreover, many robust and effective object tracking techniques used in static cameras cannot be directly applied in moving camera, such as background subtraction and constant ground plane assumption, thus making the problem more difficult. Unlike using background-based methods to extract moving objects blobs under static cameras, object detection is widely used in video

analysis under moving camera. Therefore, the challenge becomes to successfully detect objects in the moving cameras and then apply tracking techniques to track the detected ones, which are so-called tracking-by-detection schemes. However, when the object is partially or fully occluded, the detection cannot work well and thus affect the tracking result. Hence, the constrained multiple kernel (CMK) tracking technique was further adopted in the proposed system and facilitated with the estimated ground plane and Kalman filter, to overcome the occlusion issue during the tracking.

In this paper, we extend our previous work [1] and propose an efficient and robust 3D object tracking system based on adaptive ground plane estimation, which also successfully integrates structure from motion (SfM), object detection, CMK tracking, and Kalman filter framework. The proposed system begins with object detection and structure from motion for estimating camera pose. Then, the adaptive ground planes are estimated based on the camera motions, and the 3D location of the objects relative to the cameras can be inferred. By taking 3D information into account, the CMK tracking method is used to overcome the occlusion issue during the tracking. Hence, the proposed system can not only handle the occlusion but also estimate a reliable ground plane simultaneously. Figure 1 shows an example of the tracked objects on the estimated ground plane (the red squares on the ground). The number above the bounding box represents the distance of the detected objects from the camera.

The remaining of this paper is organized as follows: Section 2 gives a brief survey on the related work. In Section 3, we describe the proposed tracking system. The depth CMK tracking which includes depth map construction, CMK tracking, hypothesized association, and Kalman filter are described in Section 4, and Section 5 depicts the adaptive ground plane estimation algorithm. The experimental results are demonstrated in Section 6. Finally, the conclusion of this work is given in Section 7.

2. Related Work

Recently, ground plane estimation-based tracking methods [2–6] have attracted a lot of attention. By applying the ground plane estimation method to each frame of a video sequence for detecting a reliable ground plane, the relative 3D location of the camera and the objects can be inferred, thereby making the object tracking more robust.

In general, the existing ground plane estimation approaches can be roughly divided into two categories: 2D or 3D approaches based on the sensor type. Within 2D approaches, homography is the most popular approach for ground plane estimation, which based on feature correspondence to calculate every pair of consecutive frames and the first requisite is to find a set of reliable feature points lying on the ground plane. Usually, corner detectors such as Harris are used to extract features, followed by a robust estimation technique in which the dominant homography is estimated. Arróspide et al. [7] used Kalman filtering and Conrad and DeSouza [8] used modified expectation maximization to build confidence in the ground plane transformation across successive frames. Both of the two methods assumed the camera can only see the ground plane with objects above it, and the roll angle of sensors is zero. With the homography decomposition results combined with contour searching [9] or a Bayes filter [10] to estimate the ground plane in 2D images, homography has also been successfully used as a first step. However, again the ground plane is assumed to be the area in front of the camera, or the single color ground plane is assumed to occupy the majority of the FOV. The other 2D approaches used depth-image data or histogram of the disparity map [11] instead of traditional RGB image data [12, 13], and Jin et al. [14] proposed a ground plane detection method based on depth map driven, which grows a plane from the largest area having similar depth values in the depth map, and the largest plane is considered to be the ground plane. Kircali and Tek [15] estimated the ground plane by comparing the depth map of new coming frame with a precalibrated depth map in which the ground plane was predefined. Skulimowski et al. [16] used the gradient of the V-disparity pixel values to detect ground plane which has an arbitrary camera roll angle. Furthermore, Cherian et al. [6] reconstruct the depth map from a single RGB image by applying multiple texture-based filters with a Markov random field and estimate the ground plane based on texture-based searching segmentation. Due to the intrinsic features of the algorithm, this approach assumed the ground plane has a unique texture and the camera is parallel to the ground plane. Dragon et al. [17, 18] formulate the ground plane estimation problem as a hidden Markov model (HMM) based on temporal sampling and decomposing of homography. The decomposition of the homography with the highest probability indicates the orientation and ego motion of the camera's movement. Man et al. [19] develop a ground plane estimation approach based on monocular images with a predefined region of interest, which requires a known pitch angle of the camera.

The ground plane estimation method in 3D commonly utilizes the depth sensors as LIDAR [20] or TOF [21] to get the 3D point cloud data, which can provide the 3D structure of the environment and then be used as an effective way to estimate the ground plane. Borrmann et al. [22] use all points of 3D point cloud to calculate, which has high computation cost. RANSAC-like approaches [23, 24], which can then be used as an effective way to estimate the ground plane, are unlimited to number of iteration. Thus, processing time cannot be guaranteed. A less expensive alternative to

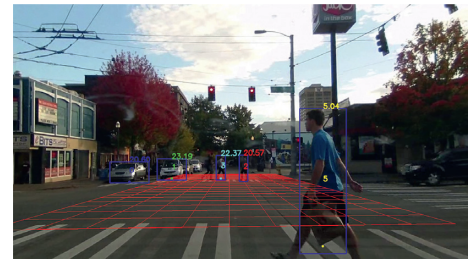


Figure 1: The ground plane estimation and 3D tracking of pedestrians and vehicles based on our system.

generate 3D point clouds is the use of a stereo camera in which the ground plane can be estimated from disparity [25]. Assuming that the scene is static, monocular approaches for simultaneous localizing and mapping (SLAM) can also be used to extract the 3D shape and then the ground plane can be estimated [26, 27]. Zhang and Czarnuch [28] proposed a perspective ground plane estimation approach which combines the robustness of 2D and 3D data analysis. Other 3D approaches [29–31] use the 3D normal vector for each raw data point rather than estimation of the raw points directly. However, we assume that the camera roll and pitch angles are zero. More recently, machine learning technique has been used in ground plane estimation, which requires minimal orientation variations (i.e., $0 \sim 15^{\circ}$) [32].

Although the above approaches can successfully detect the ground plane and achieve good experimental results, they are specifically designed to only produce one single ground plane based on the available data and not suitable for the unpredictability of dynamic road conditions. In addition, these approaches do not utilize the estimated camera pose information. In addition, the camera's pose is the most significant factor for representing the ground plane in the scene. The reliability and accuracy of the ground plane estimation can thus be improved by taking advantage of the camera pose information.

Our proposed tracking system is inspired by the approach in [33], which also has mounted the monocular dashcam on a free-moving vehicle. However, due to the driving road condition is continuously changing, if the ground plane is only estimated in the beginning may not be applicable for the entire video sequence, therefore, it is very useful to take advantage of the camera's pose information estimated from the essential matrix calculation phase. In contrast to the most existing ground plane estimation methods, our approach introduces the estimated camera yaw angle as a feedback to estimate ground plane adaptively, which aims to overcome the deficiency of the previous methods caused by fixed frame window for smoothing the results. Based on the reliably estimated ground plane, we can locate the detected objects in 3D space and combine CMK tracking with the 3D information, so as to deal with the partial or fully occlusion issues during tracking.

3. Overview of the Proposed System

The proposed tracking system is shown in Figure 2. After converting the video from the dashcam to image sequences, there are two parallel procedures launched simultaneously. In the structure from motion phase, the proposed system extracts the Harris corner features in the current image at time step t and matches them to the features observed in the previous N frames. By using the singular value decomposition (SVD), we can estimate the camera's essential matrix for each image frame. Then, according to the camera essential matrix, the ground plane for the entire image sequences can thus be estimated adaptively, where we assume the dashcam is mounted on the vehicle with a fixed height. Meanwhile, a pretrained object detector is adopted to detect desired objects such as vehicle and pedestrian in the image sequences. In the pose estimation stage, the 2D locations of detected objects can be back-projected to 3D locations by using the estimated ground plane. Once the 3D locations of the detected objects is obtained from the pose estimation stage, the depth CMK tracking is applied to track them in the Kalman filter framework. First, for each target, the 3D locations of its candidate are predicted by the Kalman filter predication. Then, the CMK tracking is applied to relocate the candidate's 3D locations by maximizing the similarity between candidates and target. The Kalman filter continually updates and finally gets the reliable tracking result. Besides, based on the object's 3D information relative to the camera motion, a depth map can be constructed to represent the relative 3D locations of all the detected objects. Therefore, with the help of depth information between the targets, the proposed system not only is able to effectively track objects but also can overcome occlusion during the tracking.

3.1. Robust Feature Extraction. The ideal ground plane estimation largely depends on the selected image feature detector, which should contain the invariance of rotation, scale, and image noise. Scale-invariant feature transform (SIFT) [34] feature is a very effective scale-space feature, but it can be very time-consuming for real-time applications. As for the speeded-up robust features (SURFs) with lower computational complexity, its stability is a major problem because it often detects unstable features even after edge suppression as a post treatment. The Harris corner feature detector is thus introduced to solve the above issues, which has also been widely studied in the previous works [35–38]. Firstly, its feature extraction execution speed can be used in real-time applications with reasonable robustness in accuracy. Secondly, to robustly estimate the ground plane, more corner points on the ground plane are welcome to participate in the calculation of the camera parameters. Figure 3 shows an example of using the Harris corner feature detector to extract feature points. The detected feature points in the current image are marked with green crosses. Feature points that are detected as outliers during the processing are marked with red crosses. These points can be matched from one image frame to the next by choosing matches that have the highest cross-correlation of image intensity for regions surrounding the points. The paths of the feature points are drawn in orange here.

3.2. Essential Matrix Calculation. Camera pose plays a crucial role in the ground plane estimation for the entire image sequences, and the computation of the camera yaw angle θ is the key to calculate camera pose. According to the study in [39], there are three camera parameters used to describe two relative poses of a camera moving on a planar surface, i.e., the polar coordinates (ρ, φ_c) and yaw angle θ of the second position c_2 relative to the first position c_1 (see Figure 4).

In addition, we can set $\rho = \nu \cdot \Delta t$, where ν is the velocity of the vehicle and Δt is the transition time between the two end positions c_1 and c_2. Therefore, only two parameters (φ, θ) need to be calculated. In addition, according to the

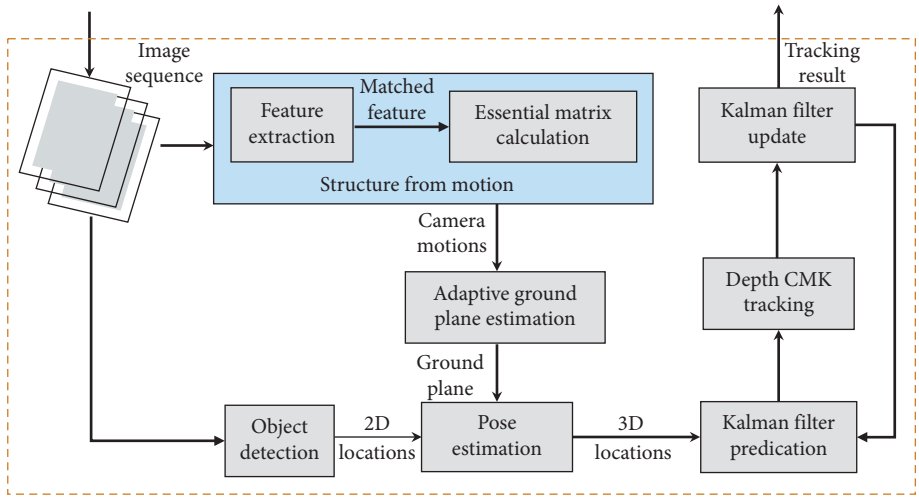


Figure 2: Overview of the proposed system.

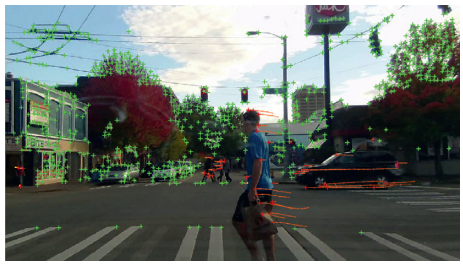


Figure 3: Example of Harris corner feature point extraction.

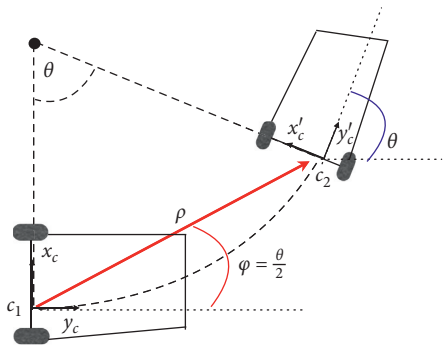


Figure 4: Rotation between camera axes in circular motion.

Ackermann steering principle, a circular motion called the instantaneous center of rotation (ICR) can be used to describe the motion of a camera mounted on a vehicle. The linear driving can be represented along with a circle of infinite radius. With this assumption, we can easily get $\varphi = \theta/2$. Thus, there is only one parameter, and the camera yaw angle θ needs to be calculated.

As we all know, the essential matrix can be represented by the rotation matrix R and the translation matrix T, which are related to the camera pose. Then, we have

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad T = \rho \cdot \begin{bmatrix} \cos\varphi \\ \sin\varphi \\ 0 \end{bmatrix}, \tag{1}$$

where we consider the camera moves on the (x, y) plane and rotates around the z axis. Given two coplanar points, p and p', which are represented as $p = [\, x \;\; y \;\; z \,]^T$ and $p' = [\, x' \;\; y' \;\; z' \,]^T$ in the image coordinates, they must meet the epipolar constraint:

$$p'^T EP = 0, \tag{2}$$

where E is the essential matrix defined as $E = [T]_\times R$. Note that R is the rotation matrix defined in (1) and $[T]_\times$ denotes the skew symmetric matrix:

$$[T]_\times = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_X & 0 \end{bmatrix}. \tag{3}$$

Then, using the constraint $\varphi = \theta/2$ and equations (1) and (3), we can obtain the expression of the essential matrix of the camera moving on a planar surface:

$$E = \rho \cdot \begin{bmatrix} 0 & 0 & \sin\frac{\theta}{2} \\ 0 & 0 & -\cos\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} & 0 \end{bmatrix}. \tag{4}$$

By replacing (4) into (2), we can notice that every image points contribute the following homogeneous equation:

$$\sin\frac{\theta}{2}\cdot\left(x'z+z'x\right)+\cos\frac{\theta}{2}\cdot\left(y'z-z'y\right)=0. \tag{5}$$

The rotation angle θ between a pair of successive images can be obtained from (5) as

$$\theta=-2\arctan\left(\frac{y'z-z'y}{x'z+z'x}\right). \tag{6}$$

Conversely, given m consecutive image points, θ can be estimated indirectly by solving linearly for the vector $[\sin(\theta/2),\ \cos(\theta/2)]$ using SVD. To this end, a $m\times 2$ data matrix D is first formed, where each row is formed by the two coefficients of equation (5), as follows:

$$\left[\left(x'z+z'x\right),\left(y'z-z'y\right)\right]. \tag{7}$$

Then, the matrix D is decomposed by using SVD:

$$D_{m\times 2}=U_{m\times 2}\Lambda_{2\times 2}V_{2\times 2}, \tag{8}$$

where the columns of $V_{2\times 2}$ contain the eigenvectors e_i of D^TD. And the eigenvector $e^*=[\sin(\theta/2),\ \cos(\theta/2)]$ corresponding to the minimum eigenvalue minimizes the sum of squares of the residuals, subject to $\|e^*\|=1$. Finally, the yaw angle of the camera θ can be estimated from e^*.

3.3. Object Detection. Object detection is the first step in the tracking-by-detection schemes, and accurate object detection can roughly determine the quality of the tracking system. Unlike detecting objects under static camera, object detection under moving cameras is more challenging due to the dynamic background, illumination changes, and so on. Because the background is constantly changing, the method based on background extraction is no longer applicable for mobile cameras. Therefore, the pretrained object detectors are widely studied in recent years. The work in [40] proposes a human detector by using histogram of gradient (HOG) as the features, which can effectively represent the shape of human. The deformable part model (DPM) [41] extends the concept of [40], which uses a root and several part templates to describe different partitions of the object, and the part templates are spatially connected with the root template according to the predefined geometry, thereby accurately depicting the object. In the latest research, the convolution neural network (CNN)-based object detector has drawn increasing attention and has achieved favorable performance, which can detect hundreds of objects with a high detection accuracy.

In this paper, the objects to be detected and tracked are mainly focusing on the pedestrians and vehicles, which should move on the estimated ground plane. In fact, these objects can be any objects on the road, such as bicycles and animals. In order to avoid detecting other false objects in the field of view, we adopt the state-of-the-art pretrained YOLOv3 detector [42], which uses the most advanced CNN technology to help detecting pedestrians and vehicles. The detector can be embedded independently in the proposed system, so as to functionally perform object detection. To efficiently track the object, the tracking procedure is launched only when the object has been detected in five consecutive image frames; otherwise, the detection is considered as a false alarm. Furthermore, the detected objects are refined by morphological operations to accurately locate their positions.

4. Depth CMK Tracking

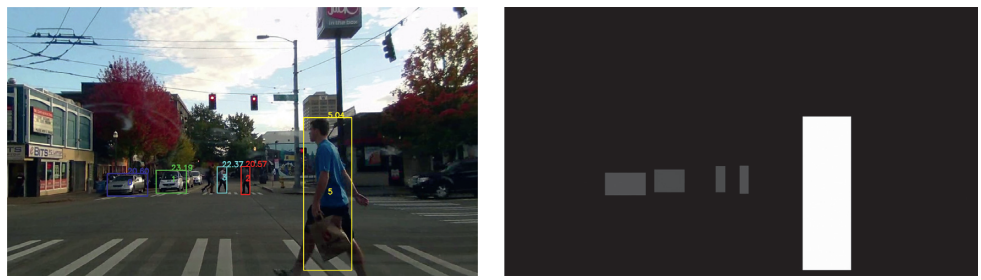
In this section, we mainly describe how to track objects with constrained multiple kernels (CMKs) in 3D space under the framework of the Kalman filter. The depth CMK tracking is triggered to track the objects when its 3D locations are obtained from the pose estimation stage (see Figure 2). In other words, we associate the objects in the current frame with the detected objects in the next frame facilitated with the Kalman filtering. On the other hand, with the help of the depth information, we can get the relative 3D locations between the objects to overcome the occlusion in the tracking. By effectively combining depth information and CMK tracking into the Kalman filter framework, the proposed system can not only track objects effectively but also well handle occlusion problems during tracking.

4.1. Depth Map Construction. A depth map can be constructed based on the 3D location of the detected objects, which represent the relative 3D location of all the tracked objects. Figure 5 shows an example of the depth map, where Figure 5(a) shows the result of detect objects and Figure 5(b) shows the corresponding depth map. The depth map depicts the relative distance between the detected object and the camera. The higher intensity (brighter) means that the detected object is closer to the camera. By using the depth map, we can roughly assess whether an object is occluded by other objects based on the visibility $v_i\in[0,1]$:

$$v_i=\frac{\text{visible area of the } i^{\text{th}} \text{ target}}{\text{total area of the } i^{\text{th}} \text{ target}}, \tag{9}$$

and if $v_i=1$, it means the i^{th} target is totally visible; if $0<v_i<1$, it implies the i^{th} target is partially occluded; otherwise, it is fully occluded by other targets. As shown in Figure 5(a), all of the five objects are totally visible. So, the visibility should be set to $v_i=1$.

4.2. CMK Tracking. In traditional kernel-based tracking, a histogram including spatial and color information is usually used to represent the target and candidate model. During the histogram extraction, the contribution of a pixel is determined by the distance between the pixel and the kernel center. In [43], the tracking problem for maximizing the similarity $\text{simi}(\mathbf{x})$ is formulated as locating $\mathbf{x}$ that maximizes the probability density function (pdf) $f(\mathbf{x})$:



(a) (b)

Figure 5: Example of the depth map, showing (a) tracked objects and (b) the relative depth map.

$$f(\mathbf{x}) = \frac{\sum_{i=0}^{N_h} \omega_i k\left(\left\|(\mathbf{x} - \mathbf{z_i})/h\right\|^2\right)}{\sum_{i=0}^{N_h} k\left(\left\|(\mathbf{x} - \mathbf{z_i})/h\right\|^2\right)}, \tag{10}$$

where $\mathbf{x}$ is the kernel center; the subscript $\mathbf{i}$ represents each pixel location inside the kernel; $k(\cdot)$is a kernel function with a convex and monotonic decreasing kernel profile. $\mathbf{z_i}$ and ω_i are the position to be considered and the weight of a pixel, respectively; h is the bandwidth of the kernel.

After back-projecting the 2D locations to 3D locations of the detected object in the pose estimation stage, we use the depth CMK tracking technique to track them. The objective of depth CMK tracking is to find the candidate model that has the highest similarity to the target model, which is composed of multiple kernels with prespecified constraints in 3D space. For an object described by N_k kernels, the total cost function $J(X)$ is defined as the sum of N_k individual kernel cost functions $J_k(X)$, which is inversely proportional to the similarity:

$$J(X) = \sum_{k=1}^{N_k} J_k(X), \quad \frac{J_k(X) \propto 1}{\text{simi}_k(X)}, \tag{11}$$

where $\text{simi}_k(X)$ is the similarity function at the location $X \in R^3$. In addition, the constraint function $C(X)$ is used to confine the kernels according to their spatial interrelationships, and in order to maintain the relative location of each kernel, the constraint function needs to be set by $C(X) = 0$. Thus, the problem is further formulated as

$$\widehat{X} = \text{argmin}_X J(X), \quad \text{subject to } C(X) = 0. \tag{12}$$

However, when the object is occluded by other objects, not all the kernels in the object can be used for matching. To overcome this issue, we assigned an adaptively adjustable weight w_k to each kernel within the object. So, the cost function for the i^{th} target is as follows:

$$J^i(X) = \sum_{k=1}^{N_k} w_k^i \cdot J_k^i(X). \tag{13}$$

Taking the depth information into account, the visibility of each object can be set as a weight to handle global optimization. In other words, the total cost function in (11) becomes to

$$J(X) = \sum_{i=1}^{N_q} v_i \cdot J^i(X) = \sum_{i=1}^{N_q} v_i \cdot \left(\sum_{k=1}^{N_k} w_k^i \cdot J_k^i(X)\right), \tag{14}$$

where N_q is the number of the objects in the q^{th} image frame and w_k^i is a weight which is proportional to the similarity for the i^{th} target of each kernel N_k.

At the same time, the constraint functions $C(X) = 0$ must be considered to maintain the relative locations of the kernels. Figure 6(a) shows an example of the object was described by 2-kernel layouts in 2D space.

Unlike the work in [44] sets the constraints in 2D space, the constraints set in this paper are based on the 3D geometry. Without loss of generality, we discussed the 2-kernel case as shown in Figure 6(b), but it can be easily extended to the multikernel case. To represent an object in the 3D space, we define an object plane $(-n_q, \pi_q)$ for the object in the q^{th} image frame, where n_q is the normal vector of the q^{th} image frames, and π_q for the offset of the plane. In order to set the constraints properly, we start to estimate two auxiliary vectors, which are $u_q = -n_q \times g_q$ and $u_{1,2} = X_1 - X_2$. First, the distance between two kernel centers should be remained the same initial distance L, which implies

$$\left\|\mathbf{u}_{1,2}\right\|^2 = (L)^2. \tag{15}$$

Second, the angle ϕ_q between the vector u_q and $u_{1,2}$ and the angle ς_q between $-n_q$ and $u_{1,2}$ should be kept constant as well:

$$\begin{cases} \dfrac{u_q \cdot u_{1,2}}{\left\|u_q\right\|\left\|u_{1,2}\right\|} = \cos\left(\phi_q\right), \\ \dfrac{-n_q \cdot u_{1,2}}{\left\|-n_q\right\|\left\|u_{1,2}\right\|} = \cos\left(\varsigma_q\right). \end{cases} \tag{16}$$

These constraints can bind the kernels of the object to each other in the 3D space during the tracking. As shown in Figure 7(a), the constraint ϕ_q restricts the left-right movement of the kernels, and the constraint ς_q restricts the forward-backward movement of the kernels which is shown in Figure 7(b).

In order to gradually decrease the total cost function and maintain the constraints satisfied during the candidate model searching, the projected gradient method in [45] is

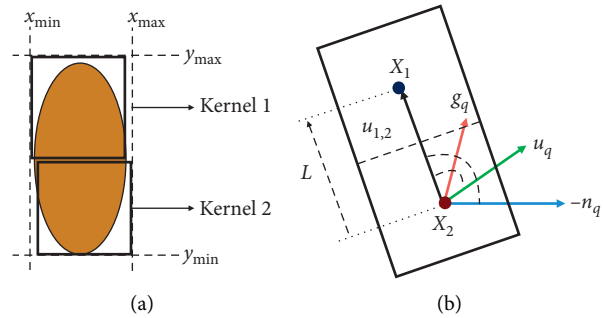


Figure 6: (a) Layout of an object with two kernels in 2D space. (b) Illustration of the 3D-based constraints in a 2-kernel layout.

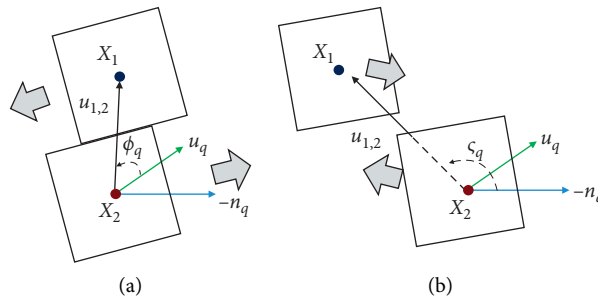


Figure 7: Constraints for binding two kernels in 3D space along the (a) left-right direction and the (b) forward-backward direction.

adopted to iteratively solve the constrained optimization problem. The basic concept of the method is to project the movement vector δ_X, i.e., the gradient vector of the $J(x)$, onto two orthogonal spaces. One is associated with decreasing the total cost function, and the other is responsible for satisfying the constraint function $C(X) = 0$:

$$\begin{aligned}\delta_X &= \alpha\left(-I + C_X\left(C_X^T C_X\right)^{-1} C_X^T\right) VWJ_X \\ &\quad + \left(-C_X\left(C_X^T C_X\right)^{-1} C_X\right) \\ &= \delta_X^A + \delta_X^B,\end{aligned} \tag{17}$$

where α is the size of searching step; I is a $3N_q \times 3N_q$ identity matrix,$\mathbf{C}(\mathbf{x}) = [c_1(\mathbf{x}), \ldots, c_m(\mathbf{x})]^T$ consists of m constraint functions, and $c_j(X)$: $\mathbb{R}^{3\cdot N_q \cdot N_k} \longrightarrow \mathbb{R}$ is the j^{th} constraint function; $\mathbf{V} = \begin{bmatrix} v_1 I_v & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & v_{N_q} I_v \end{bmatrix}$, where I_v is an $3N_k \times 3N_k$ identity matrix, which represents the visibility of kernels in the object; $\mathbf{W} = \begin{bmatrix} w_1 I_w & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & w_{N_k} I_w \end{bmatrix}$, where I_w is an $3N_q \times 3N_q$ identity matrix, which represents the similarity of the object.

As proved in [44], $\delta_{\mathbf{x}}^A$ and $\delta_{\mathbf{x}}^B$ have the following three characteristics. The first one is that $\delta_{\mathbf{x}}^A$ and $\delta_{\mathbf{x}}^B$ are orthogonal to each other. The second one is that moving along the $\delta_{\mathbf{x}}^A$ will decrease the total cost function $J(X)$ while keeping the same values of the constraint function $C(\mathbf{x})$. The last one is that moving along the $\delta_{\mathbf{x}}^B$ can lower the absolute values of constraint function $C(\mathbf{x})$. Owing to these three characteristics, the optimal solution can be reached in an iterative manner. The iteration is stopped until either the cost function and the absolute values of constraint are both lower than some given thresholds ε_j and ε_c, respectively, or the iteration count is larger than a threshold T (Algorithm 1 in [44]).

4.3. Hypothesized Association. Due to the occlusion or unreliable detection, objects may not be detected within a few frames. Therefore, some tracked targets cannot be successfully associated with the detections in subsequent frames. A hypothesized association which has been located by the CMK tracking with the best color similarity was inserted to consistently track a nonassociated target. By inserting hypothetical associations, it not only can improve the detection rate, but it also helps to continuously track the target. When an object is occluded, we can predict the 3D location by taking advantage of its 3D information, and a hypothesized association is thus used to pretend a possible detection. On the other hand, if a tracked target cannot be successfully associated to detection for several frames (empirically set as five frames in this work), then this target is considered as a missed target.

4.4. Kalman Filter Prediction and Update. Kalman filter is a traditional unscented transform-based state estimation method, which is used to approximate the mean and covariance of random variables after a nonlinear conversion. Most of tracking problems can be formulated as a state estimation problem. The tracking target can be regarded as a

Input:
Output: (g_k, φ_k)
(1) Initial frame number $N = 30$.
(2) Load a new frame f_k, k is the number of input frames.
(3) If $k < N$, set $D = [(g_1, \varphi_1)^T, \ldots, (g_k, \varphi_k)]$, go to step 6.
(4) If $\theta_{\text{rotation}} = |\theta_{k-1} - \theta_{k-N}| > \theta_{\text{threshold}}$, using $N^* = |1 - (2/\pi) \cdot \theta_{\text{rotation}}| \cdot N$ frames to estimate ground plane. Set $D = [(g_{k-N^*}, \varphi_{k-N^*})^T, \ldots, (g_{k-1}, \varphi_{k-1})^T]$ else set $D = [(g_{k-N}, \varphi_{k-N})^T, \ldots, (g_{k-1}, \varphi_{k-1})^T]$. Go to step 6.
(5) If the $\theta_{\text{rotation}} > (\pi/2)$ go to step 1.
(6) Input the D to RPCA and output the final (g_k, φ_k).

Algorithm 1: Adaptive ground plane estimation.

state, and the tracking problem is to predict and locate where the target (state) will appear in the next time. For this reason, the Kalman filter is widely used to solve tracking problems. The traditional Kalman filter is defined as follows:

$$\begin{aligned} \mathbf{x}_t &= \mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{w}_{t-1}, \\ \mathbf{y}_t &= \mathbf{H}_t \mathbf{x}_t + \mathbf{v}_t, \end{aligned} \tag{18}$$

where $\mathbf{x}_t \in R^n$ and $\mathbf{y}_t \in R^m$ denote the state and measurement vector at the time step t, respectively; $\mathbf{F}_t$ is the state transition matrix; $\mathbf{H}_t$ is measurement matrix; $\mathbf{w}_{t-1} \sim N(0, Q)$ and $\mathbf{v}_t \sim N(0, R)$ are the system and measurement noise, and these two random variables are uncorrelated Gaussian white-noise sequence, with their covariance matrix Q and R, respectively.

In the stage of prediction, the predictions for state and error covariance are as follows:

$$\widehat{x}_t = \mathbf{F}_t \mathbf{x}_{t-1}, \tag{19}$$

$$\widehat{P}_t = \mathbf{F}_t \mathbf{P}_{t-1} \mathbf{F}_t^T + \mathbf{Q}_{t-1}. \tag{20}$$

After completing the measurement, the Kalman filter will be updated as follows:

$$\begin{aligned} \mathbf{K}_t &= \widehat{P}_t \mathbf{H}_t^T \left(\mathbf{H}_t \widehat{P}_t \mathbf{H}_t^T + \mathbf{R}_t \right)^{-1}, \\ \mathbf{x}_t &= \widehat{x}_t + \mathbf{K}_t \left(\mathbf{y}_t - \mathbf{H}_t \widehat{x}_t \right), \\ \mathbf{P}_t &= \left(\mathbf{I} - \mathbf{K}_t \mathbf{H}_t \right) \widehat{P}_t. \end{aligned} \tag{21}$$

The implementation of the Kalman filter algorithm is formulated as follows.

4.4.1. Initialization. For each object, the state vector is defined as $\mathbf{x}_t = [\, u_t \;\; v_t \;\; \dot{u}_t \;\; \dot{v}_t \;\; a_t \;\; b_t \,]^T$ and the measurement vector is defined as $\mathbf{y}_t = [\, u_t \;\; v_t \;\; a_t \;\; b_t \,]^T$, where (u_t, v_t), $(\dot{u}_t, \dot{v}_t)$, and (a_t, b_t) denote the object position, velocity, and size, respectively. Hence, the initial for the state transition matrix $\mathbf{F}_t$ and the measurement matrix $\mathbf{H}_t$ are defined as

$$\begin{aligned} \mathbf{F}_t &= \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \\ \mathbf{H}_t &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \end{aligned} \tag{22}$$

4.4.2. State Transition Matrix Update. In addition, the size of object in the image sequence will probably change when it is moving toward or away from the camera, and the extracted color histogram used for similarity measurement is highly dependent on the kernel size. On the other hand, when the multiple kernel tracking is performed, the result of segmentation might be no longer reliable for estimating the similarity due to occlusion. Hence, the state transition matrix needs to be modified adaptively to reflect the potential size changes. So, we embed the factor of kernel size into the matrix $\mathbf{F}_t$:

$$\mathbf{F}_t = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 + \dfrac{\beta \nabla f(h_x)}{a_{t-1}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 + \dfrac{\beta \nabla f(h_y)}{b_{t-1}} \end{bmatrix}, \tag{23}$$

where β is the step size which also contains the smoothing factor; $\nabla f(h)$ is the derivative of the pdf with the kernel bandwidth h. Hence, the predict size of the object becomes to

$$\begin{bmatrix} \widehat{a}_t \\ \widehat{b}_t \end{bmatrix} = \begin{bmatrix} a_{t-1} + \beta \nabla f(h_x) \\ b_{t-1} + \beta \nabla f(h_y) \end{bmatrix}. \tag{24}$$

If the object is occluded so much that the average similarity value of all the kernels is lower than a certain threshold, the mechanism of state transition matrix update stops and $\mathbf{F}_t$ returns to the default setting as (22).

4.4.3. Measurement Noise Covariance Matrix Update. We use the object tracking result as a measurement to update the Kalman filter during the tracking. Although the system is robust under occlusion by using multiple kernels tracking, it still needs a mechanism to avoid the errors caused by incorrect measurements. It can be seen from (19) and (20) that not only does the Kalman gain $\mathbf{K}_t$ control the tradeoff between using the prediction and the measurement, but also it is inversely proportional to the measurement noise covariance matrix R. Hence, we can adaptively adjust the portion measurement contribution to avoid errors by changing the covariance matrix as follows:

$$\mathbf{R} = \begin{bmatrix} \sigma^2 \times J(X) & 0 & 0 & 0 \\ 0 & \sigma^2 \times J(X) & 0 & 0 \\ 0 & 0 & w^2 \times J(X) & 0 \\ 0 & 0 & 0 & h^2 \times J(X) \end{bmatrix}, \tag{25}$$

where $J(X)$ is the total cost function of all kernels; σ^2 is the predefined variance value, and w and h are the width and height of the kernel, respectively. With the help of the adaptively covariance matrix, if the total similarity between the candidate and the target is high, the diagonal term of the covariance matrix will be small. In this way, the Kalman gain will have a larger value, which will make the updated state closer to a reliable measurement.

5. Adaptive Ground Plane Estimation

Due to the unpredictability of driving road conditions, the ground plane estimated in the beginning may not be suitable for the entire image sequences. Therefore, the ground plane needs to be continuously reestimated based on the dynamic road conditions. In [33], the ground plane is reestimated and parameter smoothened every $f_g = 200$ frames to mitigate the adverse impact by the camera calibration noises. However, using a fixed number of frames for estimating the ground plane can affect the measurement accuracy when the camera is moving on a curve. In this paper, we propose to update the ground plane every single frame, based on an adaptively chosen N frames for parameter smoothing, by taking advantage of the camera rotation yaw angle calculated in the essential matrix calculation phase. The adaptive ground plane estimation algorithm is shown as follows.

In the algorithm, θ_k is the camera yaw angle at the k^{th} frame; (g_k, φ_k) is the ground plane at the k^{th} frame;$g_k \in R$ is the normal vector; and $\varphi_k \in R$ is the offset of the plane. D is a single $4 \times f_N$ matrix, and its elements are f_N ground planes, which is estimated by each pair of consecutive frames:

$$D = \left[\left(g_q, \varphi_q\right)^T, \ldots, \left(g_{q+f_N}, \varphi_{q+f_N}\right)^T \right]. \tag{26}$$

Due to the noisy camera calibrations and the unpredictability of road conditions, some ground planes (g_q, φ_q) may be unreliable; therefore, the robust principle component analysis (RPCA) [46] is applied to decompose a low-rank $4 \times f_N$ matrix A from D. The low-rank matrix's mean vector (g_k, φ_k) is considered to be our final ground plane, which is more robust to the noise contributed from the camera calibration and essential matrix calculation stage (see Section 3.2), derived from those f_N consecutive frames. Figure 8 shows an example of using a set of ground planes $\left\{ \left(g_q, \varphi_q\right)^T | q = 1, \ldots, f_N \right\}$ to estimate the final ground plane (g_k, φ_k). The gray planes are the image sequences converted from the driving recorder, and H is the camera height. The final ground plane for f_N consecutive frames (dot-line plane) is obtained from a set of ground planes (solid planes).

6. Experiment Results

In this section, we show experimental results of the proposed system on the Kitti datasets [47], which are taken with high quality dash cameras with motion pose ground truth and GPS information available. We test eight sequences (see Figure 9(a)), which are relatively short, and most of them are driving on a curvy road. Figure 9(b) shows the relative ground plane estimation results by applying our proposed method. We also test two of self-recorded video sequences captured around the University of Washington (UW) campus using a driving recorder mounted on a fixed height 1650 mm. And a more complex scenario in the ETHMS dataset, which includes multiple pedestrians on one scene, is also tested, and Table 1 shows the configurations of the tested videos.

6.1. The Relative Angular and Distance Errors. To demonstrate the accuracy of our proposed adaptive ground estimation, we compare the performance on the Kitti dataset with the following three different methods: the method in [4] is a stereo algorithm based on graphical model; the method in [17] formulates the ground plane estimation as a state continuous hidden Markov model where the hidden state contains ground plane; the method in [33] adopted the simultaneous localization and mapping (SLAM) technique to estimate the ground plane by using constant frames.

As in the method [17], the average relative angular error and distance error of the camera's motion are applied to evaluate the accuracy of the ground plane estimation. For the performance measurement, we calculate the camera poses and compare them with the given camera pose ground truth. The average relative angular and distance errors, which are

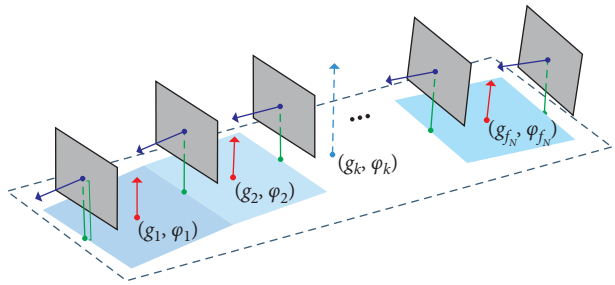
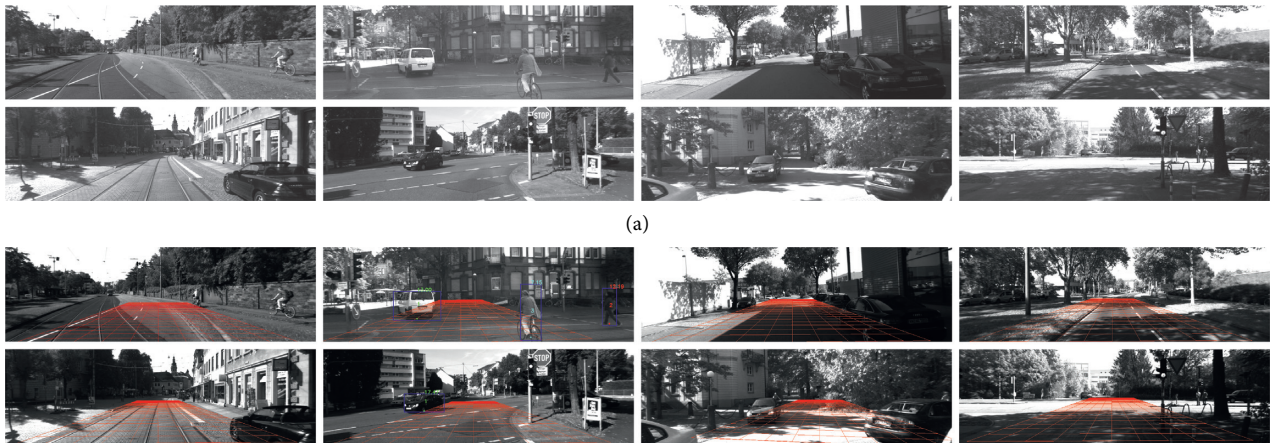


Figure 8: Example of the ground plane estimation.



(a)

(b)

Figure 9: Overview of the 8 sequences from the Kitti dataset and their relative ground plane estimation results: (a) the sequences 1–8 (row-wise starting top left), taken from the Kitti dataset; (b) the relative ground plane estimation result.

Table 1: Configurations of the datasets.

Sequence	Resolution	#Frames	Frame per second
Dataset seq#1	1242 × 375	77	15
Dataset seq#2	1242 × 375	155	15
Dataset seq#3	1242 × 375	447	15
Dataset seq#4	1242 × 375	233	15
Dataset seq#5	1242 × 375	154	15
Dataset seq#6	1242 × 375	384	15
Dataset seq#7	1242 × 375	87	15
Dataset seq#8	1242 × 375	106	15
UWcamp#1	1920 × 1080	1100	30
UWcamp#2	1920 × 1080	200	30
ETHMS #4	640 × 480	450	15

normalized by the path length, are given in Tables 2 and 3 separately.

Tables 2 and 3 show that the performance of our approach is better than the method [33] in both relative angular errors and comparable relative distance errors. That is because the estimated ground plane becomes more reliable after applying the adaptive ground plane estimation algorithm. Unlike the method in [33] that uses a constant number of frames to estimate the ground plane, our proposed method takes advantage of the estimated yaw angle in the camera pose to fight the adverse effects of the changing road conditions. Compared to the method used

Table 2: The average relative angular errors (DEG).

Dataset	Our method	Method [33]	Method [17]	Method [4]
1	0.06	0.11	0.02	0.8
2	0.05	0.20	0.07	1.22
3	0.03	0.08	0.04	0.27
4	0.01	0.03	0.23	0.92
5	0.06	0.06	0.01	0.41
6	0.03	0.08	0.07	0.39
7	0.05	0.10	0.20	3.06
8	0.10	0.59	0.11	1.68

in [17], our proposed scheme also shows better performance, except for the angular error in datasets 1 and 5, similarly except for the distance error in dataset 6 when compared with the method used in [4]. The major reason of the better performance is that our method can be well contributed by the noise reduction from the camera calibration and the unpredictability of road conditions as facilitated by taking advantage of the characteristics of adaptive-length RPCA.

6.2. Detection Performance. To demonstrate the detection performance of our proposed system, we compared it with three methods [33, 48, 49] with different human detectors on the ETHMS dataset, in terms of the detection rate and false

Table 3: The average relative distance errors (%).

Dataset	Our method	Method [33]	Method [17]	Method [4]
1	0.01	0.01	0.59	0.69
2	0.02	0.02	0.75	0.40
3	0.03	0.03	0.72	0.23
4	0.01	0.01	1.99	0.33
5	0.01	0.01	0.34	0.41
6	0.40	0.40	0.74	0.28
7	0.01	0.01	1.65	4.95
8	0.01	0.01	2.13	1.11

Table 4: Comparison of detection rate and FPPI.

Method	Detector	Detection rate (%)	FPPI
Method [48]	ISM	47	1.5
Method [48]	HOG	67.5	1
Method [49]	DPM	49.53	0.93
Method [49]	SP	51.86	0.92
Method [33]	DPM	75.58	0.89
Our method	DPM	75.71	0.82
Our method	YOLOv3	78.10	0.19

positive per image (FPPI). This shows the performance of inserting hypothesized association during tracking. The test results of the ETHMS dataset are shown in Table 4. The result shows that both the proposed method and the method in [33] are superior to the method in [48, 49]. Both methods further utilize the 3D information of the detected object, instead of only using 2D information in [48, 49]. They can effectively handle the occlusion issues. When compared with the method [33] with the DPM detector, the proposed method performs much better because it performs better in the tracking with the adaptive ground plane estimation, which results in increasing the detection rate and decreasing the FPPI. And compared with DPM and YOLOv3 detectors, the proposed method with YOLOv3 has a better performance due to the low false positive detection rate in the YOLOv3. Thanks to the proper insertion of hypothesized associations and the successive tracking, the detection rate of the proposed method can achieve about 78%. This implies that missing detection can be improved by the tracking techniques, and thus better detection results benefit the tracking performance.

6.3. Multiple Object Tracking Result. To demonstrate the tracking performance of our proposed system, we compare the performance with the following three different tracking methods: the method in [44] is a kernel-based human-tracking system which tracks a human in 2D space and without estimating the ground plane. The method in [50] uses the tracking-by-detection scheme to associate the detected objects by calculating their similarity. The method in [33] is a human tracking system which uses a constant number of frames to estimate the ground plane. To fairly evaluate the tracking performance for each method, we manually labeled 7302 locations as ground truth which includes 31 moving vehicles and 89 pedestrians across 3393 frames and also adopt the following metrics which are widely used in multiple object tracking (MOT) challenge [51].

(i) Multiple object tracking accuracy (MOTA): the measurement of tracking accuracy combines three sources of errors: false positive, false negative, and identity switches.

(ii) Multiple object tracking precision (MOTP): the measurement of object localization precision.

(iii) False positive (FP): the number of times of the system detects an object but the ground truth is not present in the image frame.

(iv) False negative (FN): the number of times of the system failed to detect an object but the ground truth is present in the image frame.

(v) ID switches (IDSs): the number of times two trajectories switch their IDs.

The comparison of the experimental results is shown in Table 5. The proposed method achieved the best performance in all of the metrics except for FN. The reason is that the CNN-based tracking by detection retains more foreground around the object regions. However, the extra extracted background information will also cause the increase in FP and IDS. The ability of the proposed depth CMK to deal with occlusion issues can be learned from the fact that there is less identity switching, while the other methods are tending to generate new object identities when occlusion occurs. To facilitate the comparison of experimental results, the red entries in Table 5 indicate that the best results in the corresponding columns and blue italics are the second best.

An additional typical example of performance comparison is shown in Figures 10 and 11, which both extract five continuous frames from 175 to 179 from the UW campus sequence 1. Figure 10 shows the tracking results in the method [33], which use a constant number to estimate ground plane. Figure 11 shows the tracking result in the proposed method, which takes advantage of the yaw angle from the camera pose to estimate the ground plane adaptively. From Figure 10, we can see that the camera mounted on the driving vehicle starts to change direction in the frame 175, and in the frame of 177, the distance of the vehicle to the camera sharply changed from 10.31 to 7.98, and then back to 8.31 in the frame 179. The estimated ground plane remains the same even when the vehicle starts to turn. Figure 11 shows the tracking performance of the proposed method using adaptive ground plane estimation, we can see that the distance of the vehicle gradually reduces from 10.51 to 8.44, and the ground plane keeps changing with the direction of the vehicle adaptively. It can be observed that the proposed method can track objects more continuously and effectively by using the adaptive ground plane estimation. Several object tracking results with estimated ground plane are shown in Figures 12–14, which show the tracking results on the UW campus sequence 2, Kitti datasets, and ETHMS datasets, respectively. The results show favorable performance of the proposed system, which not only can successively track objects but also estimate a reliable ground plane adaptively.

Table 5: The tracking performance between different methods.

Methods	MOTA (%)	MOTP (%)	FP	FN	IDS
Our method	79.7	95.8	143	1313	29
Method [33]	76.2	92.1	268	1416	53
Method [44]	63.9	82.6	590	1955	91
Method [50]	7.8	90.7	316	1223	82

(a) (b) (c) (d) (e)

Figure 10: Tracking results in the method [33] without adaptive ground plane estimation on UW campus sequence #1. (a) Frame 175. (b) Frame 176. (c) Frame 177. (d) Frame 178. (e) Frame 179.

(a) (b) (c) (d) (e)

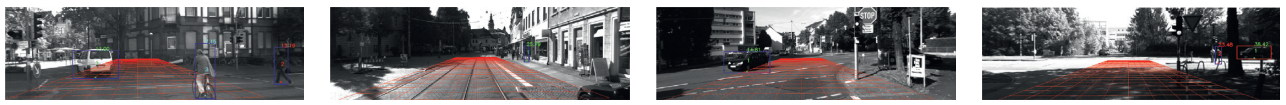
Figure 11: Tracking result in our method with adaptive ground plane estimation on UW campus sequence #1. (a) Frame 175. (b) Frame 176. (c) Frame 177. (d) Frame 178. (e) Frame 179.

(a) (b) (c) (d)

Figure 12: Tracking results with the estimated ground plane on UW campus sequence #2. (a) Frame 3. (b) Frame 4. (c) Frame 5. (d) Frame 6.

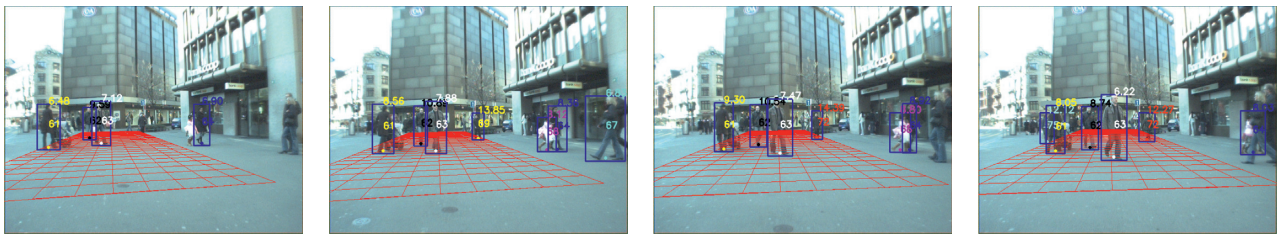
6.4. Runtime Performance. Apart from the detectors, all the experiments are processed on a laptop with an Intel Core i7, 2.2 GHz CPU with 8GB DDR. The implementation is constructed by C/C++, and the experimental settings are described as follows: in the structure from motion phase, the proposed system uses the Harris corner detector to extract 1000 features initially, which are tracked by a KLT tracker. And these corresponding feature points are used to estimate the camera pose. In object detection, the pretrained YOLOv3 detectors are independently used in the proposed system to detect objects such as human and vehicle. In the depth CMK tracking, a depth map is constructed to describe the relative 3D locations of all the tracked objects firstly, and the histogram of objects is constructed based on the HSV color space with a roof kernel; then, the K-L distance is used for all the similarity-related measurements. Table 6 shows the running time of the proposed system on different datasets with different image resolutions.

6.5. Discussion. In this paper, we proposed an adaptive ground plane estimation algorithm-based tracking system. Existing ground plane estimation methods are required to meet significant assumptions, such as the ground plane is the largest plane in the scene and the ground plane is constant in color or texture. These assumptions are not practical in cluttered or dynamic environments, especially not suitable for driving environments. Our method can robustly estimate



(a) (b) (c) (d)

Figure 13: Tracking results with the estimated ground plane on the Kitti datasets. (a) Kitti dataset 2. (b) Kitti dataset 5. (c) Kitti dataset 6. (d) Kitti dataset 8.



(a) (b) (c) (d)

Figure 14: Tracking results with the estimated ground plane on the ETHMS datasets. (a) Frame 175. (b) Frame 185. (c) Frame 191. (d) Frame 196.

Table 6: Runtime on different image resolutions.

Dataset	Resolution	Average runtime (fps)
Kitti	1242 × 375	0.87
UW campus	1920 × 1080	0.65
ETHMS	640 times 480	0.95

the ground plane on a moving camera with nonrestrictive assumption: the camera is mounted on a fixed height of the vehicle.

Combining the adaptive ground plane estimation, object detection, Kalman filter framework, and efficient depth CMK tracking techniques, the proposed tracking system can not only track the object effectively but also robustly handle occlusion during tracking. Nevertheless, several limitations are still existed. First, the proposed approach adopts the tracking-by-detection scheme to detect and then track objects, and this implies that the method highly relies on the detection results. However, if the quality of video sequences is not sufficient for the object detectors, the proposed tracking system is not able to perform well on the poor detection results. More specifically, the positive detection of a target can always trigger the tracking of a specific object. In other words, the proposed method may not work well at night or some cases of insufficient lighting. Second, the proposed method effectively estimates ground planes based on certain video frames when the vehicle moves on flat roads, but if the roads are severely bumpy, it will produce less reliable estimation, resulting in larger error of the object back-projection and impacting accuracy of the reprojected 3D information. Hence, the proposed method is not reliable for the unmanned aerial vehicle, because its height dynamically changes and then infers unreliable 3D information of objects.

In the future, we will focus on improving the performance of the algorithm by enhancing the accuracy of the object detection algorithms. In addition, we will also test our algorithms on video sequences that have higher outdoor complexity and more objects visible in the scene.

7. Conclusion

We propose a robust object tracking system and ground plane estimation simultaneously in a dashcam mounted on a free-moving vehicle. The proposed system effectively integrates the object detection, ground plane estimation, CMK tracking, and Kalman filter framework to relocate the objects in 3D space, and the estimated camera yaw angle has been adopted into the adaptive ground plane estimation. With the depth CMK tracking, the 3D positions of the detected targets are updated on the more reliable ground plane and occlusion issue is also handled in the tracking system. The experimental result shows that the proposed method greatly improved the tracking performance. Such tracking system can be regarded as a key component for high-level applications, such as video analysis in a large scale of the mobile network. Besides, the proposed framework can also be futher applied to the advanced driver assistance system (ADAS).

Data Availability

The Kitti dataset used to support the findings of this study may be released upon application to the KITTI Vision Benchmark Suite, which is a project of Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago. The dataset can be downloaded for free at this web page http://www.cvlibs.net/datasets/kitti/raw_data.php. The ETHMS dataset can be downloaded on the following web page https://data.vision.ee.ethz.ch/cvl/aess/dataset/#pami09. Requests for self-recorded UW data, 6/12 months, after the publication of this article, will be considered by the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the 111 Project, under grant B17007, and in part by the China Scholarship Council Funding.

References

[1] T. Liu, Y. Liu, Z. Tang, and J.-N. Hwang, "Adaptive ground plane estimation for moving camera-based 3d object tracking," in *Proceedings of the 2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–6, IEEE, London, UK, October 2017.

[2] L. Ladickỳ, P. Sturgess, C. Russell et al., "Joint optimization for object class segmentation and dense stereo reconstruction," *International Journal of Computer Vision*, vol. 100, no. 2, pp. 122–133, 2012.

[3] D. Maier and M. Bennewitz, "Appearance-based traversability classification in monocular images using iterative ground plane estimation," in *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4360–4366, IEEE, Vilamoura, Portugal, October 2012.

[4] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3D reconstruction in real-time," in *Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IV)*, pp. 963–968, IEEE, Baden-Baden, Germany, June 2011.

[5] C. Yuan and G. Medioni, "3D reconstruction of background and objects moving on ground plane viewed from a moving camera," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pp. 2261–2268, IEEE, New York, NY, USA, June 2006.

[6] A. Cherian, V. Morellas, and N. Papanikolopoulos, "Accurate 3D ground plane estimation from a single image,," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, pp. 2243–2249, IEEE, Kobe, Japan, May 2009.

[7] J. Arróspide, L. Salgado, M. Nieto, and R. Mohedano, "Homography-based ground plane detection using a single on-board camera," *IET Intelligent Transport Systems*, vol. 4, no. 2, pp. 149–160, 2010.

[8] D. Conrad and G. N. DeSouza, "Homography-based ground plane detection for mobile robot navigation using a modified em algorithm," in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, pp. 910–915, IEEE, Anchorage, AK, USA, May 2010.

[9] P. Ke, C. Meng, J. Li, and Y. Liu, "Homography-based ground area detection for indoor mobile robot using binocular cameras," in *Proceedings of the 2011 IEEE 5th International Conference on Robotics, Automation and Mechatronics (RAM)*, pp. 30–34, IEEE, Qingdao, China, September 2011.

[10] S. Kumar, A. Dewan, and K. M. Krishna, "A bayes filter based adaptive floor segmentation with homography and appearance cues," in *Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing*, pp. 1–8, Mumbai, India, December 2012.

[11] R. Labayrade, D. Aubert, and J.-P. Tarel, "Real time obstacle detection in stereovision on non flat road geometry through "v-disparity" representation," in *Proceedings of the Intelligent Vehicle Symposium, 2002*, pp. 646–651, IEEE, Yangzhou, China, June 2002.

[12] Y. Lang, H. Wu, T. Amano, and Q. Chen, "An iterative convergence algorithm for single/multi ground plane detection and angle estimation with rgb-d camera," in *Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP)*, pp. 2895–2899, IEEE, Québec City, Canada, September 2015.

[13] J. Zhao, J. Katupitiya, and J. Ward, "Global correlation based ground plane estimation using v-disparity image," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 529–534, IEEE, Roma, Italy, April 2007.

[14] Z. Jin, T. Tillo, and F. Cheng, "Depth-map driven planar surfaces detection," in *Proceedings of the 2014 IEEE Visual Communications and Image Processing Conference*, pp. 514–517, IEEE, Valletta, Malta, December 2014.

[15] D. Kırcalı and F. B. Tek, "Ground plane detection using an rgb-d sensor," in *Proceedings of the Information Sciences and Systems 2014*, pp. 69–77, Springer, Shenzhen, China, April 2014.

[16] P. Skulimowski, M. Owczarek, and P. Strumiłło, "Ground plane detection in 3D scenes for an arbitrary camera roll rotation through "v-disparity" representation," in *Proceeding of the 2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 669–674, IEEE, Prague, Czech Republic, September 2017.

[17] R. Dragon and L. V. Gool, "Ground plane estimation using a hidden markov model," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4026–4033, Columbus, OH, USA, June 2014.

[18] R. Dragon, B. Rosenhahn, and J. Ostermann, "Multi-scale clustering of frame-to-frame correspondences for motion segmentation," in *Proceedings of the European Conference on Computer Vision*, pp. 445–458, Springer, Florence, Italy, October 2012.

[19] Y. Man, X. Weng, X. Li, and K. Kitani, "Groundnet: monocular ground plane normal estimation with geometric consistency," in *Proceedings of the 27th ACM International Conference on Multimedia*, pp. 2170–2178, Nice, France, October 2019.

[20] M. W. McDaniel, T. Nishihata, C. A. Brooks, and K. Iagnemma, "Ground plane identification using lidar in forested environments," in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, pp. 3831–3836, IEEE, Anchorage, AK, USA, May 2010.

[21] F. Mufti, R. Mahony, and J. Heinzmann, "Robust estimation of planar surfaces using spatio-temporal ransac for applications in autonomous vehicle navigation," *Robotics and Autonomous Systems*, vol. 60, no. 1, pp. 16–28, 2012.

[22] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter, "The 3d hough transform for plane detection in point clouds: a review and a new accumulator design," *3D Research*, vol. 2, no. 2, p. 3, 2011.

[23] D. Scaramuzza, F. Fraundorfer, and R. Siegwart, "Real-time monocular visual odometry for on-road vehicles with 1-point ransac," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, pp. 4293–4299, IEEE, Kobe, Japan, May 2009.

[24] X. Qian and C. Ye, "Ncc-ransac: A fast plane extraction method for 3-D range data segmentation," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2771–2783, 2014.

[25] S. Se and M. Brady, "Ground plane estimation, error analysis and applications," *Robotics and Autonomous Systems*, vol. 39, no. 2, pp. 59–71, 2002.

[26] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," in *Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2531–2538, IEEE, Nice, France, September 2008.

[27] B. Micusik and J. Kosecka, "Piecewise planar city 3d modeling from street view panoramic sequences," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2906–2912, IEEE, Miami Beach, FL, USA, June 2009.

[28] C. Zhang and S. Czarnuch, "Perspective independent ground plane estimation by 2D and 3D data analysis," *IEEE Access*, vol. 8, pp. 82024–82034, 2020.

[29] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, "Real-time plane segmentation using rgb-d cameras," in *Robot Soccer World Cup*, pp. 306–317, Springer, Berlin, Germany, 2011.

[30] S. Choi, J. Park, J. Byun, and W. Yu, "Robust ground plane detection from 3D point clouds," in *Proceedings of the 2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)*, pp. 1076–1081, IEEE, Seoul, Korea, October 2014.

[31] T. Liu and Y. Liu, "Deformable model-based vehicle tracking and recognition using 3-D constrained multiple-Kernels and Kalman filter," *IEEE Access*, vol. 9, 2021.

[32] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1106–1112, IEEE, San Juan, Puerto Rico, June 1997.

[33] K.-H. Lee, J.-N. Hwang, G. Okopal, and J. Pitton, "Ground-moving-platform-based human tracking using visual slam and constrained multiple kernels," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3602–3612, 2016.

[34] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[35] B. Liang, N. Pears, and Z. Chen, "Affine height landscapes for monocular mobile robot obstacle avoidance," *In Proceedings of Intelligent Autonomous Systems*, vol. 8, pp. 863–872, 2004.

[36] J. Zhou and B. Li, "Homography-based ground detection for a mobile robot platform using a single camera," in *Proceedings 2006 IEEE International Conference on Robotics and Automation*, pp. 4100–4105, Beijing, China, June 2006.

[37] J. Zhou and B. Li, "Robust ground plane detection with normalized homography in monocular sequences from a robot platform," in *Proceedings of the 2006 International Conference on Image Processing*, pp. 3017–3020, Varzim, Portugal, September 2006.

[38] N. Simond and M. Parent, "Obstacle detection from ipm and super-homography," in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4283–4288, San Diego, CA, USA, November 2007.

[39] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, June 2016.

[40] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pp. 886–893, IEEE, San Diego, CA, USA, June 2005.

[41] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2009.

[42] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018.

[43] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

[44] C.-T. Chu, J.-N. Hwang, H.-I. Pai, and K.-M. Lan, "Tracking human under occlusion based on adaptive multiple kernels with projected gradients," *IEEE Transactions on Multimedia*, vol. 15, no. 7, pp. 1602–1615, 2013.

[45] J. A. Snyman, *Practical Mathematical Optimization*, Springer, Berlin, Germany, 2005.

[46] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma, "Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization," in *Advances in Neural information Processing Systems*, pp. 2080–2088, Springer, Berlin, Germany, 2009.

[47] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, IEEE, Providence, Rhode Island, June 2012.

[48] A. Ess, B. Leibe, K. Schindler, and L. Van Gool, "Robust multiperson tracking from a mobile platform," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, pp. 1831–1846, 2009.

[49] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *Proceedings of the CVPR 2011*, pp. 1201–1208, IEEE, Colorado Spring, CO, USA, June 2011.

[50] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[51] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "Mot16: a benchmark for multi-object tracking," 2016, http://arxiv.org/abs/1603.00831.

Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 5531965, 14 pages
https://doi.org/10.1155/2021/5531965

Research Article

Research on Road Adhesion Condition Identification Based on an Improved ALexNet Model

QiMing Wang,[1] **JinMing Xu**,[1] **Tao Sun**,[1] **ZhiChao Lv**,[2] **and GaoQiang Zong**[1]

[1]*School of Mechanical Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China*
[2]*School of Automotive Studies, Tongji University, Shanghai 201804, China*

Correspondence should be addressed to QiMing Wang; wang.qiming2008@163.com

Received 5 February 2021; Revised 10 June 2021; Accepted 28 June 2021; Published 17 July 2021

Academic Editor: Wen LIU

Automotive intelligence has become a revolutionary trend in automotive technology. Complex road driving conditions directly affect driving safety and comfort. Therefore, by improving the recognition accuracy of road type or road adhesion coefficient, the ability of vehicles to perceive the surrounding environment will be enhanced. This will further contribute to vehicle intelligence. In this paper, considering that the process of manually extracting image features is complicated and that the extraction method is random for everyone, road surface condition identification method based on an improved ALexNet model, namely, the road surface recognition model (RSRM), is proposed. First, the ALexNet network model is pretrained on the ImageNet dataset offline. Second, the weights of the shallow network structure after training, including the convolutional layer, are saved and migrated to the proposed model. In addition, the fully connected layer fixed to the shallow network is replaced by 2 to 3, which improves the training accuracy and shortens the training time. Finally, the traditional machine learning and improved ALexNet model are compared, focusing on adaptability, prediction output, and error performance, among others. The results show that the accuracy of the proposed model is better than that of the traditional machine learning method by 10% and the ALexNet model by 3%, and it is 0.3 h faster than ALexNet in training speed. It is verified that RSRM effectively improves the network training speed and accuracy of road image recognition.

1. Introduction

As car ownership has risen continuously, traffic jams, delays, and accidents spiraled upward. According to statistics [1, 2], 16.12% of traffic accidents on highways are attributed to slippery road conditions and the driver's response to changes in terrain caused by road damage. To improve vehicle safety, research on vehicle safety control has gradually changed focus from passive safety to active safety. As an important part of the vehicle's perception of the surrounding environment, road surface type recognition plays an important role not only in the power, smoothness, and comfort of intelligent driving vehicles, but also in vehicle safety.

In the 1960s, Wiesel and Hubel [3] found that their unique network structure could effectively reduce the complexity of the feedback neural network when they studied the neurons used for local sensitivity and direction selection in the cortex of cats and proposed a convolutional neural network (CNN). Lecun et al. [4] made a great breakthrough in optical character recognition and computer vision by using a CNN, which promoted the development of computer vision. In recent years, CNNs have been widely used in many fields and have shown excellent performance in image target detection [5–7] and classification [8, 9]. The appearance of a CNN also provides a new solution for road condition recognition.

Several researchers and institutions have focused on pavement type identification and adhesion coefficient prediction. Chen [10] extracted the feature parameters of the gray-level cooccurrence texture matrix of the pavement image, studied the selection of pavement texture features, and achieved certain results. However, this method has the disadvantages of fewer image features and lower recognition

accuracy. Bekhtike and Kobayashi [11] used a camera to collect pavement images and evaluated the texture attributes obtained from the fractal dimension using Gaussian process regression for function approximation and predicted road types by fusing road texture features and vibration data received from motion. This method still has some limitations. For instance, when the background lighting changes obviously, motion blur occurs, or if the road is covered by rain, snow, or ice, it is difficult to accurately identify the road type.

Ward and Iagnemma [12] successfully classified asphalt, paved, and gravel roads with acceleration sensors. This method has drawbacks when the road surface roughness is similar, and it is obviously insufficient to use acceleration data to distinguish the road type. Alonso et al. [13] proposed a real-time acoustic pavement state recognition system based on tire noise, using a noise measurement system and a signal processing algorithm to classify the pavement state, and achieved accurate classification of wet and dry pavement states.

Neupane and Gharaibeh [14] proposed a method for detecting pavement types based on heuristic lidar and identified the pavement type by the mean and variance of the laser reflection intensity. This method is mainly used for asphalt pavement. Jonsson et al. [15] proposed road classification based on near infrared camera image spectral analysis, using KNN and support vector machine (SVM) methods to classify dry, wet, icy, and snowy roads and achieved certain results. Bystrov et al. [16] used automotive ultrasonic sensors to analyze reflected ultrasonic signals for road classification, with a recognition accuracy of up to 89%.

Meng [17] proposed a method based on the basic principles of machine learning to classify pavement types by combining data from vertical acceleration sensor signals and camera features. The accuracy of using an acceleration sensor or image data to identify road type was only 62% and 88%, respectively. When the two were combined, because of the small sample size, accuracy reached only 90%. Wang [18] classified and discriminated road images based on high-dimensional features and RBF neural networks and performed recognition experiments on eight different road images with an accuracy of approximately 78.4%. Based on the SVM, Zhao et al. [19] obtained the best classification model by PSO parameter optimization, classified the road types, and improved the recognition accuracy of the test image, achieving an accuracy rate of over 90% for the five basic road types.

Casselgren et al. [20] studied the light performance of asphalt pavements covered by water, ice, or snow. They conducted a detailed study on the changes in light intensity with the angle of incidence and spectrum changes and proposed two different wavebands to classify road conditions. Linton and Fu [21] described a networked vehicle-based winter road condition (RSC) monitoring solution that combines vehicle-based image data with data from road weather information systems. Jokela et al. [22] presented a method and evaluation results to monitor and detect road conditions (ice, water, snow, and dry asphalt).

The developed device is based on light polarization changes when reflected from the road surface. The recognition capability has been improved with texture analysis, which estimates the contrast content of an image, but the results show that the proposed solution does not currently adapt to different conditions perfectly well. Yeong [23] and Yu and Salari [24] developed a pothole detection system and method using 2D LiDAR. Caltagirone [25] developed a method for road detection in point cloud top-view images using fully CNN. However, according to the material presented in [26, 27], even LIDAR, which is the safest laser, can cause damage to the human eye during longer exposure (e.g., cataracts and burn of the retina). In the future, with the popularity of smart cars, this type of laser may be a problem. We consider a method to improve road condition recognition through image vision.

To summarize, most road recognition algorithms are based on traditional machine learning. Traditional machine learning extracts artificial image features as algorithm input. It was found that the process had a certain randomness, and the whole process including the classification algorithm was complex. To solve these problems, this paper proposes a road surface condition identification method based on an improved ALexNet model, namely, the road surface recognition model (RSRM).

Therefore, the main contributions of this paper can be summarized as follows:

(1) The ALexNet [28] network model is pretrained on the ImageNet [29] dataset offline. The weights of the shallow network structure after training, including the convolutional layer, are saved and migrated to the proposed model. In addition, the fully connected layer fixed to the shallow network is replaced by 2 to 3, which improves the training accuracy and shortens the training time.

(2) The traditional machine learning and improved ALexNet model are compared, focusing on adaptability, prediction output, and error performance, among others.

2. Research Method for Identifying Road Surface Conditions Based on Improved ALexNet Model (RSRM)

The traditional road type identification method has some limitations, such as a complex extraction process, weak adaptability, poor light robustness, low recognition accuracy, and difficulty in practical application. Meanwhile, the rapid development of artificial neural networks has also given birth to the progress of deep learning [30] in recent years. Common deep learning networks include autoencoders [31], deep belief networks [32], and CNNs. In deep learning, CNNs play a key role in image recognition. Road condition recognition belongs to the field of image recognition; therefore, in this study, the road image recognition model is built by combining CNNs and deep learning theory. With the help of CNN's self-learning and training of road image features, the actual road types can be identified.

2.1. Convolutional Neural Networks (CNN). CNN [33] has high efficiency and accuracy in image recognition, which is due to the shared parameters of convolutional kernels in the hidden layer and sparsity of interlayer connections. A CNN model is generally formed by alternately stacking convolutional layers and pooling layers, and the specific operation for input data is saved in the weight of this layer. The loss function is used to evaluate the difference between the output and target values. The optimizer uses the difference between the target value and the output value as the feedback signal to update the weight value through the back-propagation algorithm [34] and finally reduces the loss value corresponding to the current target, which makes the network prediction more accurate. The feature values of the last layer of the pooling layer generate a list of vectors through the fully connected layer and input them to SoftMax [35], for classification and recognition. The CNN training process is shown in Figure 1.

2.1.1. Convolutional Layers. The convolutional layers principally perform convolution operation on the image or feature map, which is input into the convolution layer, to extract feature and output the convoluted feature map. Therefore, as shown in equation (1), each feature map of convolution layer is obtained by combining and calculating multiple feature maps output from the previous layer:

$$X_n^l = f\left(\sum_{i\in M_n} X_i^{l-1} * K_{in}^l + b_n^l\right), \tag{1}$$

where M_n is the feature map set filtered from the input feature map, X_n^l is the n th feature map in the l th layer, K_{in}^l is the i th element of the n th convolution kernel in the l th layer, b_n^l is the n th offset of the l th layer, and " $*$ " is the process of convolution.

2.1.2. Pooling Layers. The pooling layer, also known as the lower sampling layer, is mainly used to reduce the calculation amount of feature extraction. The pooling layer retains the number of feature maps but changes the size of the feature maps; equation (2) represents the calculation process of the sampling layers.

$$X_n^l = f\left(\beta_n^l \operatorname{down}\left(X_n^{l-1}\right) + b_n^l\right), \tag{2}$$

where down $(\cdot)$ is the lower sampling (pooling) function, β_n^l is the n th multiplication offset of the l th layer, and b_n^l is the n th offset of the l th layer.

The lower sampling function is largely divided into mean-pooling and max-pooling. Mean-pooling is to calculate the average of all elements in the pooling area.

$$p_n = \frac{1}{|R_n|}\sum_{i\in R_n} c_n. \tag{3}$$

The max-pooling is to select the maximum element in the pool area.

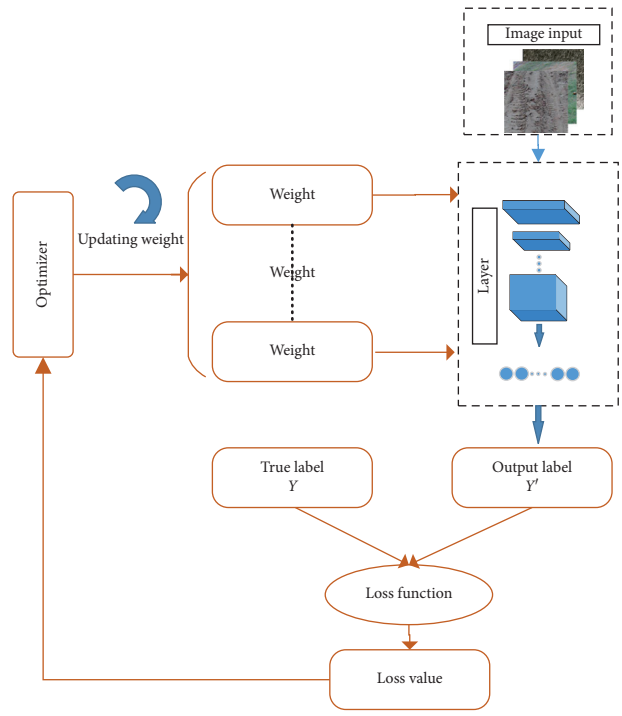


Figure 1: The training process of road surface image based on deep learning.

$$p_n = \max_{i\in R_n} c_i, \tag{4}$$

where R_n is the n th pooling area in the feature map and c_i is the i th pixel value in R_n.

2.1.3. Fully Connected Layers. The fully connected layers generally locate at the last part of the hidden layers in CNN. The fully connected layers form a multilayer perceptron like the shallow neural network, which nonlinearly combines the feature vectors output by the convolutional layer and the pooling layer to get the output.

2.1.4. Output Layers. The output layers in CNN are usually behind the fully connected layers. For image classification problems, the output layers use a logical function or a normalized exponential function (SoftMax function) to output classification labels. The range of the multi-classification label y in SoftMax regression is $y \geq 2$. The training sample set is composed of k labeled samples:

$$T = \left\{\left(x_{(1)}, y_{(1)}\right)\left(x_{(2)}, y_{(2)}\right), \ldots, \left(x_{(k)}, y_{(k)}\right)\right\}, \tag{5}$$

where $y_{(i)} \in \{1, 2, \ldots, k\}$ is the classification labels, and $x_{(i)}$ is the sample set. j represents different classifications, and it is estimated probability value. The probability that a single sample is classified into class K is

$$P(y = j|x), \quad (j = 1, 2, \ldots, k). \tag{6}$$

The regression sample set is transformed into a k-dimensional probability vector, and it is given by

$$h(x_i|\theta) = \begin{bmatrix} p(y_{(i)} = 1|x_{(i)}, \theta \\ p(y_{(i)} = 2|x_{(i)}, \theta \\ \vdots \\ p(y_{(i)} = k|x_{(i)}, \theta \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e_j^T x_{(i)}} \begin{bmatrix} e^{\theta_1^T x_{(i)}} \\ e^{\theta_2^T x_{(i)}} \\ \vdots \\ e^{\theta_k^T x_{(i)}} \end{bmatrix}, \tag{7}$$

where $\theta = [\theta_1^T, \theta_2^T, \ldots, \theta_k^T]$ is learning parameters, and $\theta_1^T, \theta_2^T, \ldots, \theta_k^T \in \mathbf{R}^{n+1}$; the purpose of $(1/\sum_{j=1}^{k} e^{\theta_j^T x_{(i)}})$ is to normalize the probabilities and make the sum of the probabilities be 1.

Through the training of sample set, the optimizer adjusts parameters to minimize model loss function value, and its loss function formula is defined as

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\{y_{(i)} = j\} \log \frac{e^{\theta_j^T x_{(i)}}}{\sum_{i=1}^{k} e^{\theta_I^T x_{(i)}}}\right]. \tag{8}$$

2.2. ALexNet Network. The typical CNN models usually include GoogLeNet [36], VGG-16 [37], and ALexNet. GoogLeNet and VGG-16 have 22 and 16 layers, respectively. Theoretically, the deeper the number of model layers, the better the classification effect.

As a classic model, ALexNet accelerates the development of deep learning, which is a milestone in image recognition. Before the research, we have done the comparison between ALexNet and VGG, GoogleNet, and other networks, and ALexNet network can reach a higher recognition accuracy in a shorter time.

Second, theoretically, the deeper the model layer is, the better the classification effect is. However, the training process of deep convolution network is extremely difficult. For example, many parameters lead to the disappearance of backpropagation gradient and overfitting. At the same time, the deeper network often needs to consume more computing resources. ALexNet can meet the accuracy of road image recognition, while reducing computer resources. So, the diversity of road images is low, and ALexNet can achieve higher recognition accuracy and occupy less computer resources.

Third, the road image is relatively simple, and the latest network is usually to solve more complex image classification problems. ALexNet has been able to solve the problem of road condition image recognition extremely well.

This is due to several advantages of the ALexNet network:

(1) In the training process, dropout is used to randomly ignore some neurons to avoid overfitting the model.

(2) Samples are data augmented [38] to expand the samples with insufficient training images.

(3) Rectified Linear Units (ReLUs) [39] are used as the excitation function of the network, which improves its nonlinearity and solves the problem of gradient dispersion. To solve the problem of gradient dispersion, ALexNet adopts the ReLU activation function. ReLU is defined as follows:

$$\mathrm{ReLU}(x) = \max(x, 0). \tag{9}$$

In Figure 2, comparing ReLU and sigmoid [40] activation function curves, it shows that when x is greater than 0, the ReLU gradient value is always a constant of 1. The derivative of the sigmoid function is like the curve shape of the Gaussian function, but not constant. The derivative at both ends of the sigmoid curve becomes smaller. Therefore, the network with ReLU as an activation function converges quickly, which is helpful in accelerating training.

Figure 3 and Table 1 show the structure and parameters of ALexNet. The model is mainly composed of five convolutional layers and three fully connected layers. The number of convolution kernels in five convolution layers is 96, 256, 384, 384, and 256, respectively. The role of the pooling layer is mainly to reduce the size of the feature image after convolution. The nodes of the three fully connected layers are 4096, 4096, and 1000, respectively. SoftMax can classify 1000 categories.

2.3. Road Surface Recognition Model Based on RSRM. The ALexNet network was pretrained on the ImageNet database with at least one million images offline, and the weights and parameters of each layer were obtained after training. The trained network has a strong ability to learn features, especially curves, edges, and contours of an image. To improve the efficiency of network training and reduce the training time, this study takes the trained ALexNet network as the pretrained model and transfers its parameters to the RSRM using fine-tuning transfer learning [41]. ALexNet, SVM, and BP use the classic structure. SVM algorithm is based on the characteristics of the road image for road color and texture feature extraction experiments.

Similarly, RSRM consists of a convolutional layer, pooling layer, fully connected layer, and SoftMax classification layer. By analyzing the characteristics of actual pavement images, nine typical pavement types are selected, as shown in Figure 4, focusing on nine typical road surface types; therefore, 9-label SoftMax is used to replace the original classifier in the ALexNet network. In addition, as shown in Figure 5, two fully connected layers are trained on the actual road pavement test set and to replace the original three fully connected layers. The number of nodes in the two fully connected layers are 4096 and 1000, respectively.

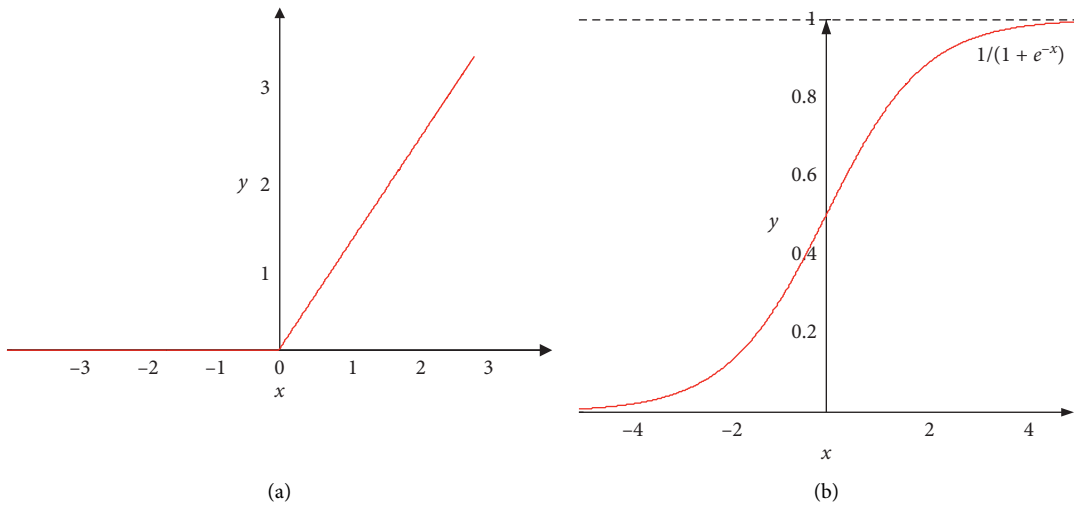


Figure 2: Activation function. (a) ReLU function image. (b) Sigmoid function image.

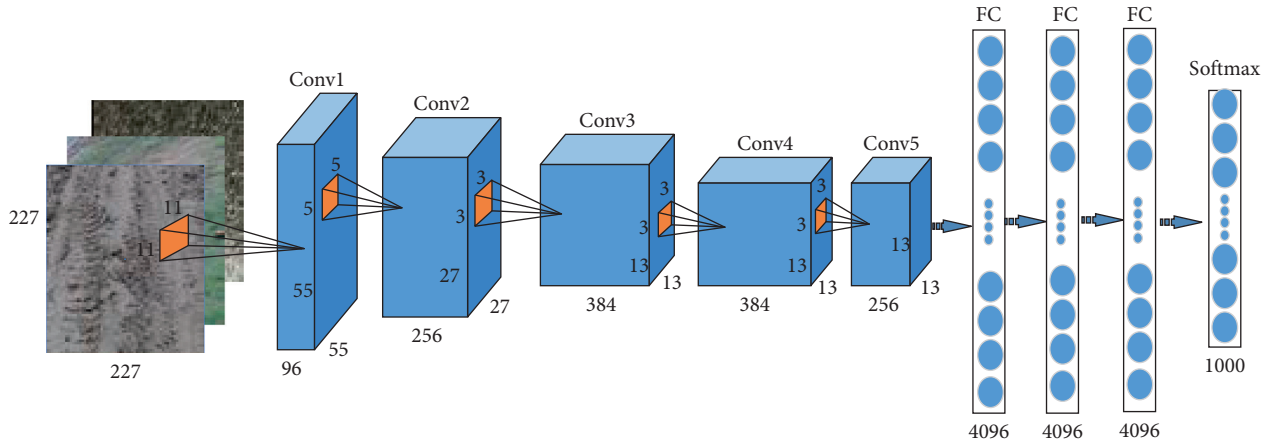


Figure 3: The structure of ALexNet network models.

Table 1: ALexNet structure parameters.

Name	Parameter	Stride	Number of convolution kernels
Conv 1	$11 \times 11 \times 3$	4	96
Pool 1	$1 \times 3 \times 3 \times 1$	2	—
Conv 2	$5 \times 5 \times 48$	1	256
Pool 2	$1 \times 3 \times 3 \times 1$	2	—
Conv3	$3 \times 3 \times 256$	1	384
Conv4	$3 \times 64 \times 384$	1	384
Conv5	$3 \times 3 \times 384$	1	256
Pool5	$1 \times 3 \times 3 \times 1$	2	—
Fc6	4096	—	—
ReLU6	ReLU	—	—
Dropout	0.5	—	—
Fc7	4096	—	—
ReLU7	ReLU	—	—
Dropout	0.5	—	—
Fc8	1000	—	—
Prob	SoftMax	—	—
Output	1000	—	—

Through the above steps, the problem of road surface image classification and recognition is solved.

3. Experimental Settings

3.1. Road Surface Data Acquisition System. The road collection test vehicle was a sedan with a length of 3564 mm, width of 1620 mm, and height of 1527 mm. Its wheelbase was 2340 mm. The camera model was LeTMC-520. As shown in Figure 6, the camera was installed at the air intake grille in the front of the vehicle, at an angle of −10° from the horizontal grille. The installation height from the ground was 350 mm. In this study, considering the complex weather conditions in the actual driving process, three typical weather conditions, namely, cloudy, sunny, and rainy, were selected for road image data collection. Note that the images of the actual road test set are all taken by the vehicle during driving.

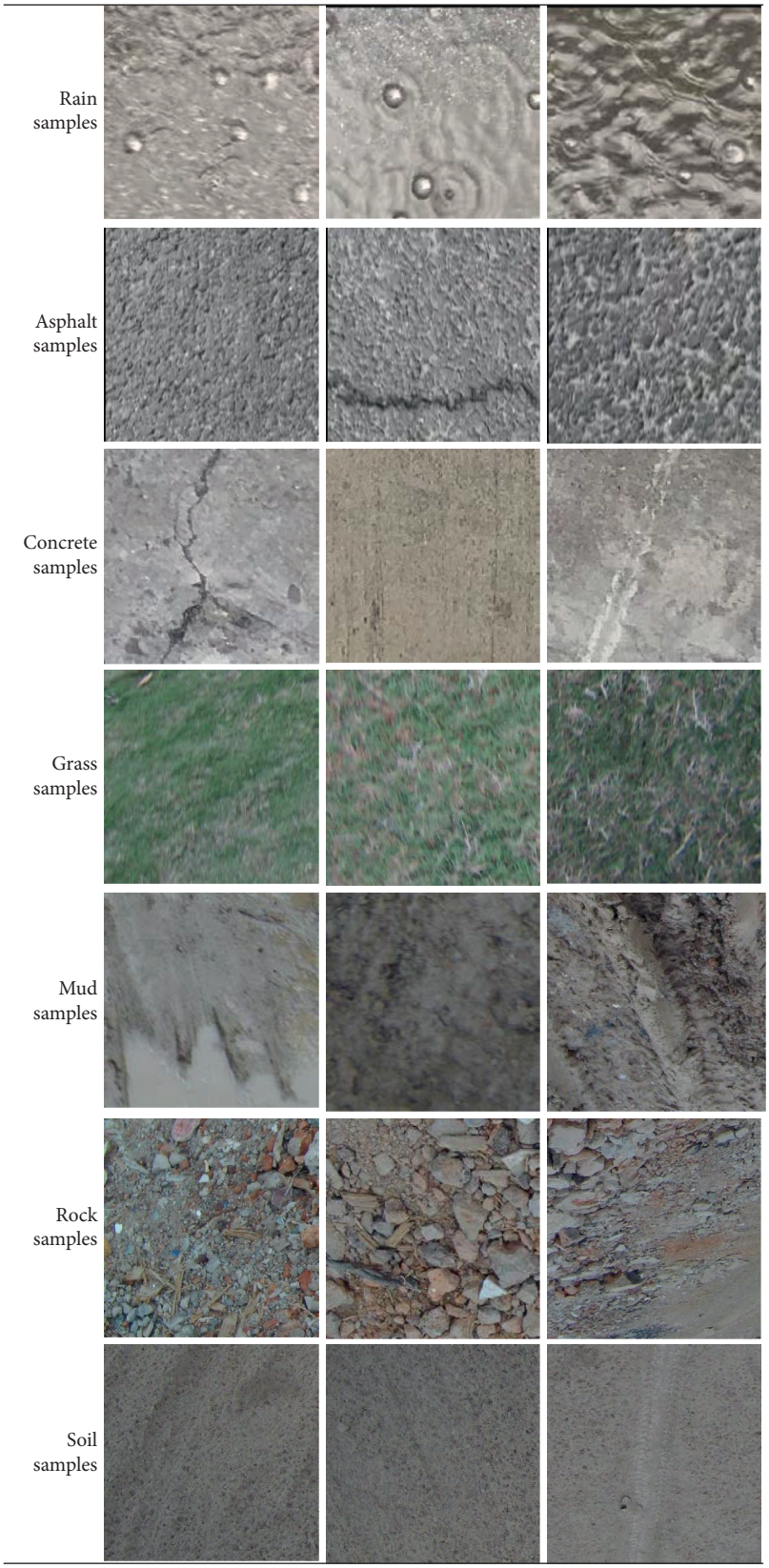


Figure 4: Image samples of road surface state.

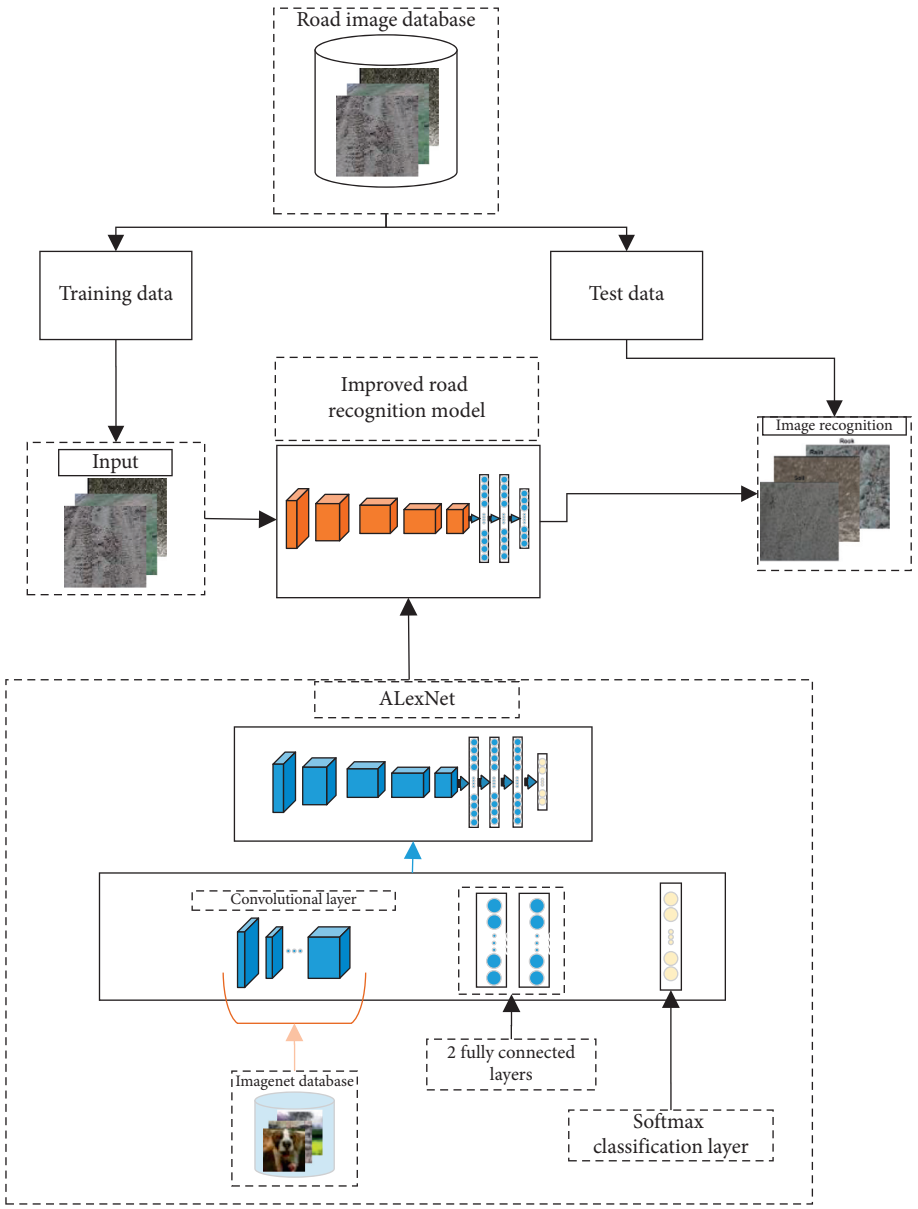


Figure 5: Structure of the road surface recognition model.

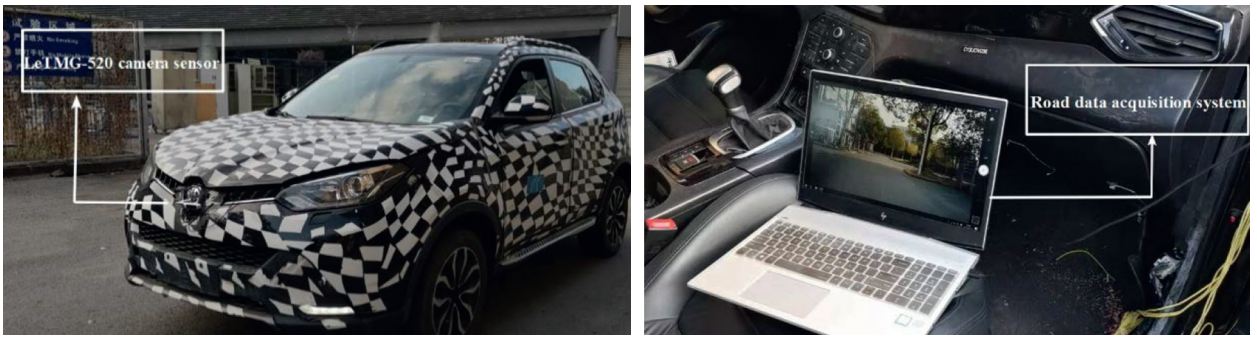


Figure 6: Camera installation and data acquisition.

In addition, a road surface data analysis system server configuration was performed on a desktop computer with a 64 bit operating system, 16 GB of memory, an AMD Ryzen 5 3600 6-Core Processor, and a GeForce GTX 1660 graphics processing unit.

3.2. Establishment of Road Surface Image Database. The image standards were selected according to the typical pavement types: asphalt, concrete, grass, mud, rain, rock, soil, wet asphalt, and wet concrete, and the images with clear quality were used for the road surface image database (RSID). The sample size of each pavement was 2000, in which the training set and test set were divided in a ratio of 7 : 3.

3.3. Experimental Procedure

Step 1: Image preprocessing.

Scaling and cropping operations are performed on all road surface images to ensure a uniform image size, which can meet the requirements of the neural network module in MATLAB.

Step 2: Building the training set and test set.

RSID is divided into the training set and test set in a ratio of 7 : 3.

Step 3: Building the RSRM.

Focusing on nine typical road surface types, the 9-label SoftMax is used to replace the original classifier in the ALexNet network. The next step is to use the trained ALexNet network as the pretrained model and transfer its parameters to the RSRM using fine-tuning transfer learning.

Step 4: Model training

Model training that uses the stochastic approach initializes the model parameters; sets the momentum parameters, learning rate, and training time; and freezes the parameters of the five convolutional layers and pooling layers. Through the above, we replace the parameters of the two fully connected layers and 9-label SoftMax with a fresh new one.

Step 5: Model testing.

The remaining 30% of the RSID was used as a test set to verify the accuracy and speed of the RSRM.

4. Results and Analysis of Experiment

4.1. Experiment of Road Image Feature Extraction. The role of the convolution layer is to extract features by performing a convolution operation on the image or feature map. First, we pretrained the improved ALexNet network model (5 convolutional layers) on the ImageNet dataset. Second, the weights of the shallow network structure after training were saved and transferred to the RSRM. Finally, to observe the feature extraction effect of RSRM more clearly, taking the mud image as an example, the output features of each convolution layer were visualized. Figure 7 shows the mud pavement image after preprocessing.

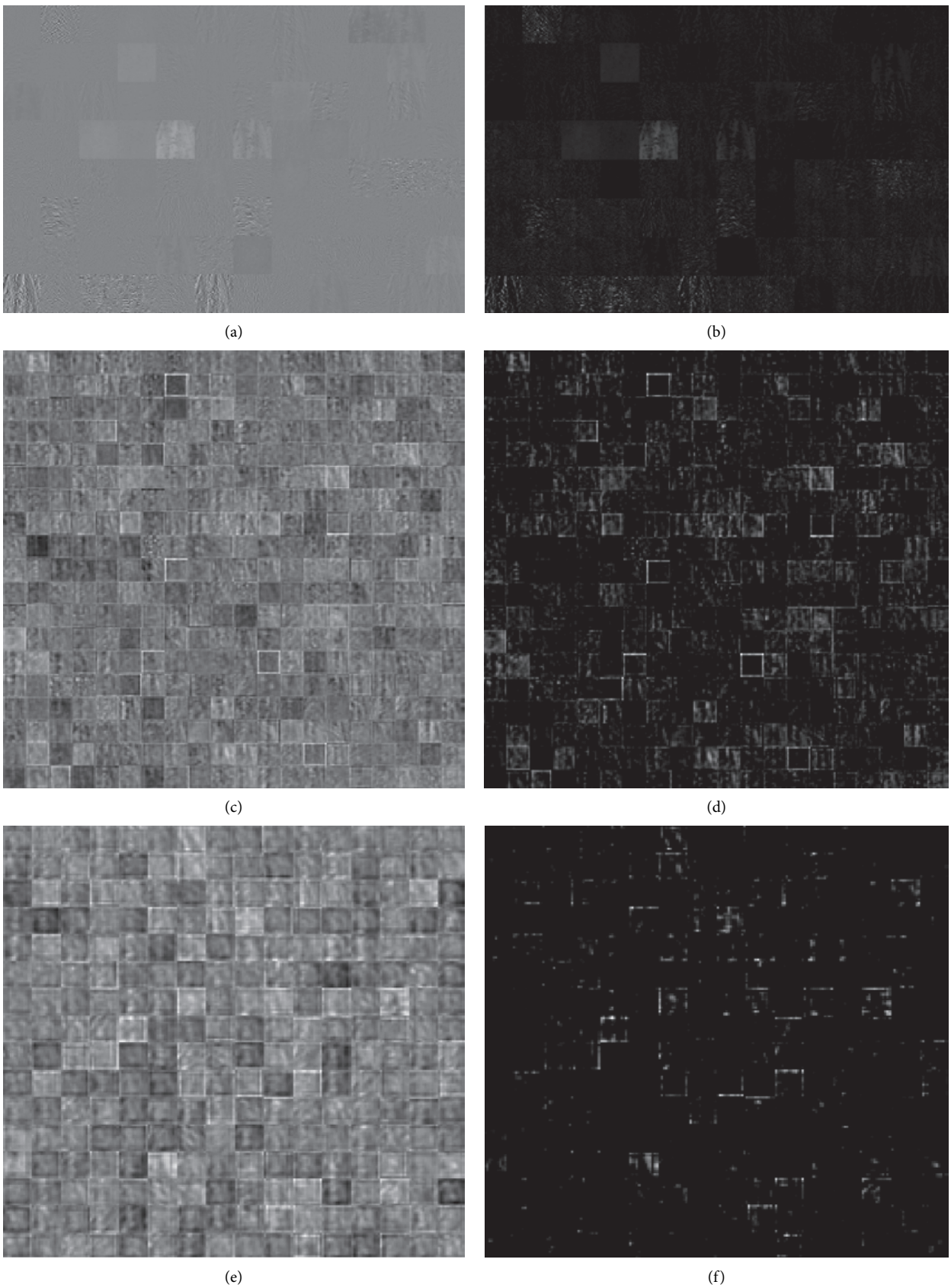
Figure 7: Image of mud pavement after preprocessing.

As shown in Figure 8(a), the preprocessed mud image is extracted with 96 feature maps through Conv1. The convolution layer mainly extracts edges and details of the image. After several convolution kernel operations in the convolution layer, the image retains most of the information of the original image. As shown in Figure 8(b), the convolved image is processed by the ReLU1 activation function, and the edge information and detailed information of the mud surface road image are more obvious. Figures 8(c) and 8(d) show the feature map after Conv3 and relu3. It can be seen from the figure that the convolution kernel can extract more edge information, and the outline of the mud road surface image is clearer. Figures 8(e) and 8(f) show the feature map after Conv5 and relu5. It also reveals that as the number of convolutional layers increases from the first layer to the fifth layer, the resolution ratio of the image decreases, and the image output from the convolutional kernel becomes increasingly abstract.

According to the above image feature extraction experiments, the convolution layer integrates shallow features or underlying features to form more abstract features. This makes the expression of road information more comprehensive and can also use high-level abstract features for pavement classification and recognition.

4.2. Experiments of Road Surface Type Recognition Based on RSRM. RSID contains 18000 images of nine pavement types, such as asphalt, wet asphalt, rain, concrete, wet concrete, soil, mud, grass, and rock. To verify the validity of the RSRM proposed in this paper, 70% of RSID were randomly selected as the training set, with a total of 12600 pieces, and the remaining 30% was used as the test set. There were 600 images for each type of pavement in the test set, for a total of 5400 pavement images.

Table 2 shows the RSRM training parameter setting. The test tolerance is the number of iterations that the loss of test set before network training stops can be greater than or equal to the previously smallest loss. This can stop training by setting the test tolerance when test loss is no longer



(a) (b) (c) (d) (e) (f)

Figure 8: Feature extraction results of different Conv and ReLu for mud pavement images. (a) Conv1. (b) ReLu1. (c) Conv3. (d) ReLu3. (e) Conv5. (f) ReLu5.

Table 2: Training parameter setting.

Verification frequency	Minibatch size	Epoch	Learning rate	Test tolerance
20/iteration	128/images	25	0.001	5/iteration

reduced, to avoid overfitting, save computer memory and improve training speed.

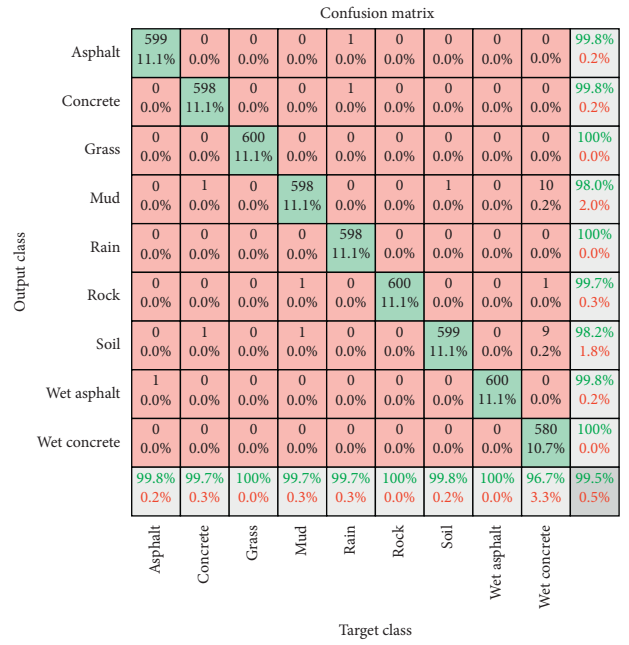
Table 3 shows the classification results of test samples based on RSRM in this research. Table 4 lists the recognition results of some image based on RSRM.

According to Step 2, there are 600 samples per category in the test set; this is true for the asphalt pavement type. As can be seen in Table 3, an asphalt pavement image (600 samples) was misidentified, and the recognition accuracy was 99.8%. This is because part of the asphalt pavement presents a dry-wet state, which makes it extremely similar to the image characteristics of asphalt pavement; thus, it is misidentified as wet asphalt. A total of 598 concrete pavement samples were correctly identified, and the remaining two were identified as soil and mud pavements, with an accuracy rate of 99.7%. The reason is that the color and image texture of some concrete, soil, and mud pavements are similar under dry conditions. The number of rain pavement samples correctly identified is 598, with the remaining two misidentified as wet asphalt and wet concrete pavements; meanwhile, the identification accuracy rate was 99.7%. For soil pavement, 599 samples were correctly identified, and the remaining one was classified as mud. A wet soil road surface often forms the mud surface, and the high probability of these two pavement features cooccurring in a single image is the main factor leading to false positives. The total number of wet concrete surfaces is 600, of which 580 are correctly recognized, 9 are identified as soil, one is identified as rock pavement, and the last 10 are identified as mud pavement. Thus, the recognition accuracy of wet concrete pavement is 96.7%. This is because the color of the wet concrete pavement is brown-gray after being wet. The recognition accuracy of grass, rock, and wet asphalt are higher than other surfaces, which is due to the significant difference in color and texture features compared to other road images.

4.3. Experiments of Classification Method Comparison. In this study, RSRM is compared against the ALexNet model, support vector machines (SVM), and backpropagation (BP) neural networks. The results are shown in Table 5. According to previous research, color and texture are the main features of road images. The SVM [42] classification model needs to extract road image features manually. In the three-color spaces of HSV, RGB, and YCM, there are nine color components of the road image, namely H, S, V, *R*, G, B, Y, C, and M. The gray-level cooccurrence matrix is used to extract four texture similar information of road surface images, such as contrast, correlation energy, and entropy. The BP neural network [43–45] has five layers, the number of nodes in each layer is 100, and the optimization algorithm uses stochastic gradient descent.

In this section, RSRM is compared with the BP neural network, SVM, and ALexNet models, focusing on the analysis of model prediction output, error performance, training time, and detection time.

Table 3: The classification results of test samples based on RSRM.



As shown in Figures 9 and 10 , RSRM significantly improves accuracy of road surface identification compared to ALexNet. Specifically, RSRM converged at 216 iterations, realizing an accuracy of 96.38%. However, the ALexNet network has yet to converge after 500 iterations. Therefore, transfer learning and optimization of the fully connected layer can effectively improve the training efficiency and accuracy of the model. The ALexNet model requires a longer training time and larger dataset to match the accuracy of RSRM.

Figure 11 and Table 5 illustrate the identification accuracy of different methods. The average recognition accuracy of BP, SVM, ALexNet, and RSRM was 92.84%, 89.59%, 97.57%, and 99.48%, respectively. The accuracy of RSRM and ALexNet is more than 95%, which shows the superiority of deep learning methods. Traditional machine learning methods, such as SVM and BP neural networks, are not suitable for representing variations in illumination intensity due to their manual features. The SVM classifier is suitable for small datasets, which is why it has not achieved good results in road datasets. Table 6 shows the average time taken by each learning model to classify a test image. The test times of all models for a given road image are almost the same. The results show that the accuracy of the deep learning model is higher than that of the traditional machine learning method.

Table 4: Results of the test sample classification.

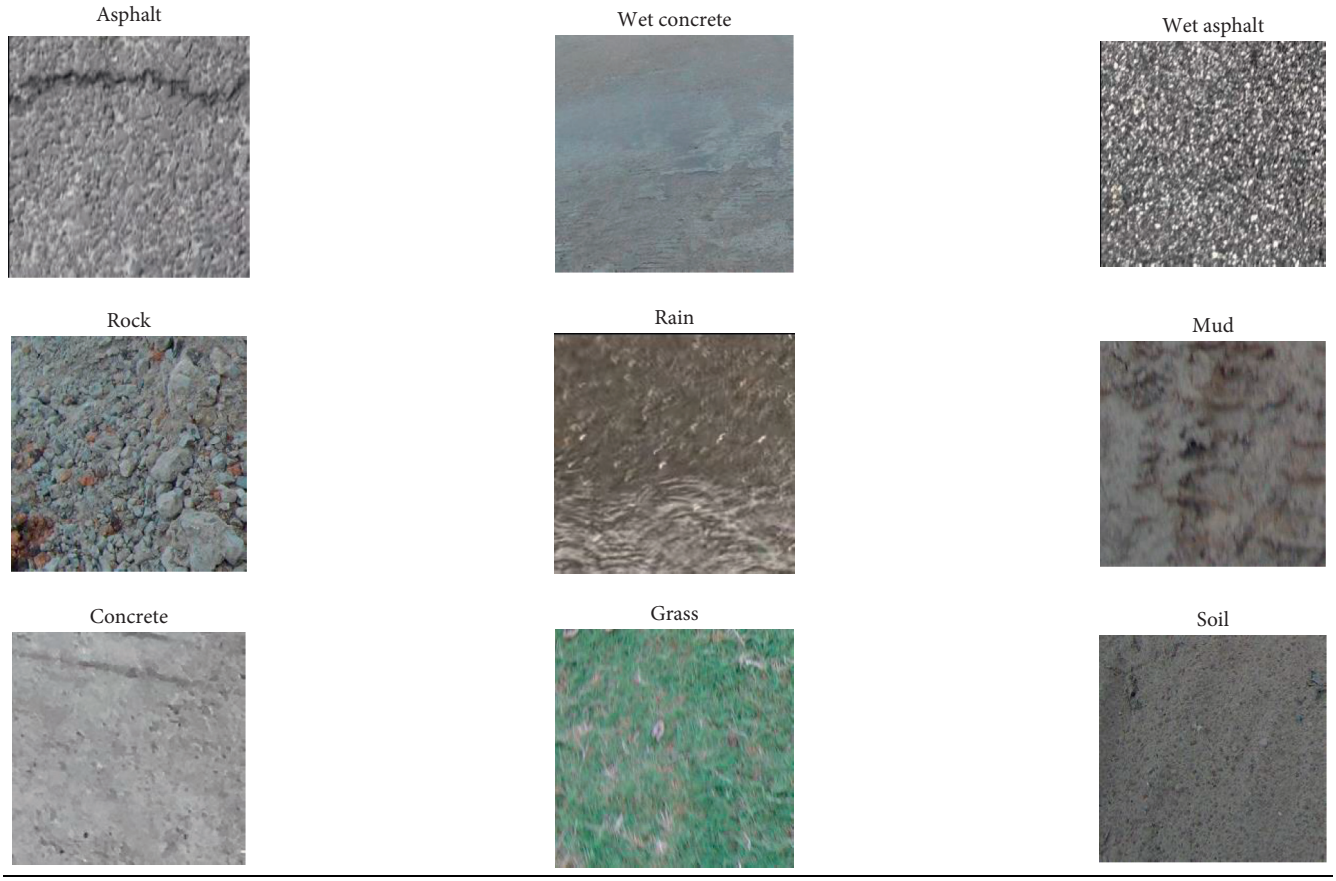


Table 5: The accuracy of different model identification.

Model (%)	SVM	BP	ALexNet	RSRM
Training accuracy	91.72	93.36	98.36	100
Test accuracy	89.59	92.84	97.57	99.48

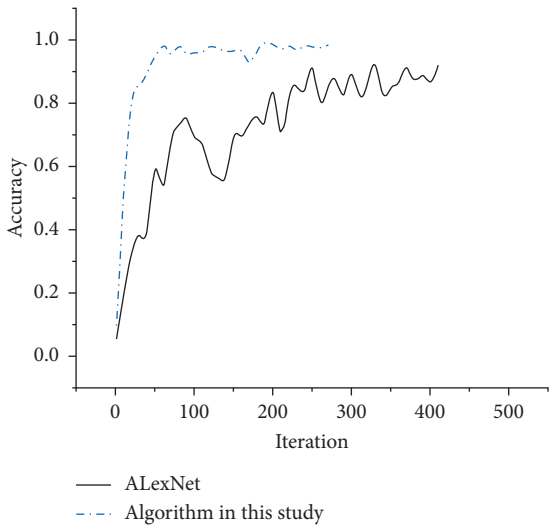


Figure 9: Accuracy during validation.

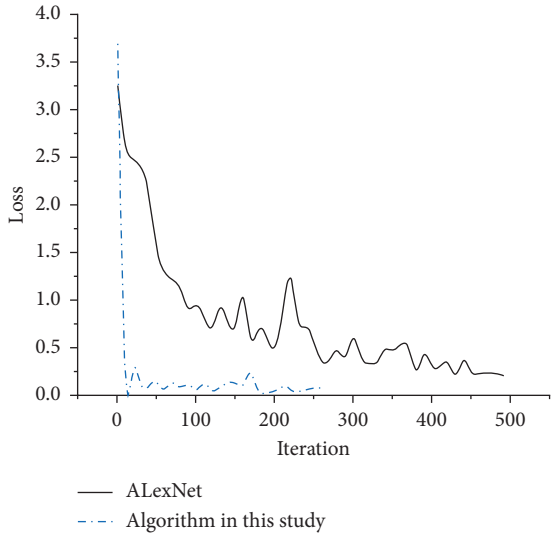


Figure 10: Loss during validation.

SVM is effective in dealing with small-scale datasets, which is difficult to implement for large-scale training samples. In the process of image classification based on BP, the upper layer of neurons and the next layer of neurons are fully connected, which leads to excessive

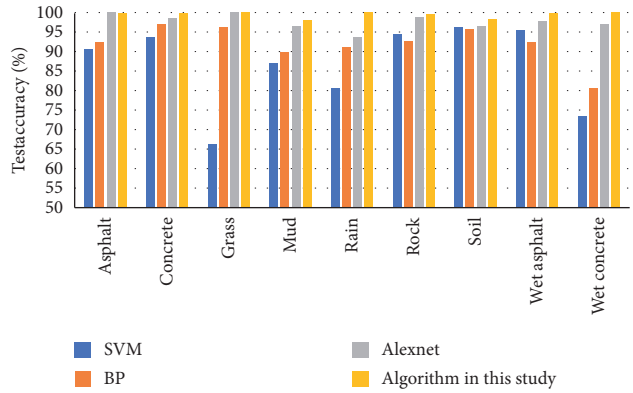


Figure 11: Test results of different pavement identification models.

Table 6: Training time and individual image recognition time for different models.

Model	SVM	BP	ALexNet	RSRM
Training time/h	1.6	2.2	1	0.4
Recognition time/s	0.15	0.16	0.15	0.14

training weight and overfitting. However, a CNN can effectively reduce the training weight and improve the training speed using a convolution operation.

In addition, this study proposes a method for testing tolerance thresholds to stop model training and reduce the number of fully connected layers. Based on this, the SoftMax classifier for nine labels is designed. The training time for the proposed method was 1.6 h, BP was 2.2 h, ALexNet was 1 h, and RSRM required 0.4 h (RSRM does not include pretraining time), and it took 0.14 s to classify a test image. The training speed of RSRM is four times that of SVM and five times that of BP. Meanwhile, the recognition accuracy was 1.91% higher than that of ALexNet, 6.64% higher than BP, and 9.89% higher than SVM. RSRM can effectively improve the training efficiency and accuracy of the model.

In summary, the BP neural network is not suitable for recognizing multiple ranges of road image databases because of the large number of neurons, the number of network layers cannot be too large, and the computing time is long, which can easily lead to overfitting and inconvenience in processing high-dimensional data. SVM feature extraction is complex and only suitable for small datasets. The proposed method not only achieves fast and high-precision recognition of road surface types in a short training time but also meets the perception requirements of actual road conditions.

5. Conclusion

This paper presents a pavement identification method based on an improved ALexNet model. First, the ALexNet network model is pretrained on the ImageNet dataset offline. Second, the weights of the shallow network structure after training, including the convolutional layer, are saved, and migrated to the proposed model. In addition, the fully connected layer fixed to the shallow network is replaced by 2 to 3, which improves the training accuracy and shortens the training time, and the 9-label SoftMax replaces the original classifier in the ALexNet network. In addition, the proposed method is compared with the BP neural network, SVM, and ALexNet models, focusing on the prediction output, error performance, and rapidity of the model. The results show that the recognition accuracy of RSRM is 99.48%, which is higher than that of ALexNet, BP, and SVM by 1.91%, 6.64%, and 9.89%, respectively. Moreover, this paper proposes a method for testing tolerance thresholds to stop model training and reduce the number of fully connected layers, which can save 0.6 h of training time and increase the training speed to four times that of SVM and five times that of BP. In conclusion, the deep learning model not only has higher accuracy than the traditional machine learning method but also can achieve higher recognition accuracy in a shorter time, which can meet the perception requirements of actual road conditions. The research method is not only suitable for road recognition, but also suitable for human-vehicle-road collaborative perception of the vehicle environment.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded by the National Natural Science Foundation of China, under grant no. 51575232, and the Shanghai Youth Science and Technology Talents Sailing Project, under grant no. 19YF1434600.

References

[1] X. Sun, *Research on Risk Coupling of Highway Traffic Safety Based on Catastrophe Theory*, Beijing Jiaotong University, Beijing, China, 2015.

[2] S. D. Gleave, R. Frisoni, and F. Dionori, "EU road surfaces: economic and safety impact of the lack of regular road

maintenance, study, transport and tourism," *European Union*, vol. 24, no. 135, pp. 897–906, 2014.

[3] T. N. Wiesel and D. H. Hubel, "Effects of visual deprivation on morphology and physiology of cells in the cat's lateral geniculate body," *Journal of Neurophysiology*, vol. 26, no. 6, pp. 978–993, 1963.

[4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[5] R. Girshick, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, Columbus, OH, USA, June 2014.

[6] W. Liu, D. Anguelov, D. Erhan, C. Szegedy et al., *SSD Single Shot Multibox Detector, European Conference on Computer Vision*, Springer, Cham, Berlin, Germany, 2016.

[7] J. Redmon, "You only look once: unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, Las Vegas, NV, USA, June 2016.

[8] D. Lan, "Target detection method based on convolutional neural network for SAR image," *Journal of Electronics Information Technology*, vol. 38, no. 12, pp. 3018–3025, 2016.

[9] H. Zhang, L. I. Yumei, and F. U. Yingying, "Research on natural scene image classification based on haar-CNN model," *Journal of Sichuan Normal University (Natural Science)*, vol. 40, no. 1, pp. 119–126, 2017.

[10] Y. Chen, "Image analysis applied to black ice detection," *Applications of Artificial Intelligence IX. International Society for Optics and Photonics*, vol. 1468, pp. 551–562, 1991.

[11] M. A. Bekhti and Y. Kobayashi, "Prediction of vibrations as a measure of terrain traversability in outdoor structured and natural environments," *Image and Video Technology*, vol. 34, no. 8, pp. 282–294, 2015.

[12] C. C. Ward and K. Iagnemma, "Speed-independent vibration-based terrain classification for passenger vehicles," *Vehicle System Dynamics*, vol. 47, no. 9, pp. 1095–1113, 2009.

[13] J. Alonso, J. M. López, I. Pavón, M. Recuero et al., "On-board wet road surface identification using tyre/road noise and support vector machines," *Applied Acoustics*, vol. 76, pp. 407–415, 2014.

[14] S. R. Neupane and N. G. Gharaibeh, "A heuristics-based method for obtaining road surface type information from mobile lidar for use in network-level infrastructure management," *Measurement*, vol. 131, pp. 664–670, 2019.

[15] P. Jonsson, J. Casselgren, and B. Thörnberg, "Road surface status classification using spectral analysis of NIR camera images," *IEEE Sensors Journal*, vol. 15, no. 3, pp. 1641–1656, 2014.

[16] A. Bystrov, E. Hoare, T. Y. Tran, N. Clarke, M. Gashinova, and M. Cherniakov, "Road surface classification using automotive ultrasonic sensor," *Procedia Engineering*, vol. 168, pp. 19–22, 2016.

[17] W. Meng, "Research on vehicle pavement type recognition technology based on machine learning," *Journal of Military Science and Technology*, vol. 38, no. 8, pp. 1642–1648, 2017.

[18] W. F. Wang, "Classification of slippery road images based on high-dimensional features and RBF neural network," *Journal of Transport Information and Safety*, vol. 31, no. 2, pp. 32–35, 2013.

[19] J. Zhao, H. Wu, and L. Chen, "Advances in traffic safety methodologies and technologies," *Journal of Advanced Transportation*, vol. 2017, Article ID 6458495, 21 pages, 2017.

[20] J. Casselgren, M. Sjödahl, and J. LeBlanc, "Angular spectral response from covered asphalt," *Applied Optics*, vol. 46, no. 20, pp. 4277–4288, 2007.

[21] M. A. Linton and L. Fu, "Connected vehicle solution for winter road surface condition monitoring," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2551, no. 1, pp. 62–72, 2016.

[22] M. Jokela, M. Kutila, and L. Le, "Road condition monitoring system based on a stereo camera," in *Proceedings of the IEEE 5th International Conference on Intelligent Computer Communication and Processing*, pp. 423–428, IEEE, Cluj-Napoca, Romania, August 2009.

[23] B. K. Yeong, "Pothole detection system using 2D LiDAR and camera," in *Proceedings of the Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 744–746, IEEE, Milan, Italy, July 2017.

[24] X. Yu and E. Salari, "Pavement pothole detection and severity measurement using laser imaging," in *Proceedings of the IEEE International Conference on Electro/Information Technology*, pp. 1–5, IEEE, Mankato, MN, USA, May 2011.

[25] L. Caltagirone, "Fast LIDAR-based road detection using fully convolutional neural networks, intelligent vehicles symposium," in *Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (iv)*, Los Angeles, CA, USA, June 2017.

[26] K. Schulmeister and G. Vees, "Principles for consistent application of the IEC laser product safety standard based on legal requirements," in *Proceedings of the International Laser Safety Conference*, vol. 2005, no. 1, March 2005.

[27] W. J. Marshall, "Update of national standard for the safe use of lasers outdoors," *Journal of Laser Applications*, vol. 11, no. 5, pp. 234–236, 1999.

[28] A. Krizhevsky, I. Sutskever, and E. Geoffrey, "Hinton, imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.

[29] J. Deng, "Imagenet: a large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, IEEE, Miami, FL, USA, June 2009.

[30] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[31] Y. Pu, Z. Gan, R. Henao et al., "Variational autoencoder for deep learning of images, labels and captions," pp. 2352–2360, 2016, https://arxiv.org/abs/1609.08976.

[32] A. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, 2011.

[33] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: a comprehensive review," *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[34] Y. LeCun, "Handwritten digit recognition with a back-propagation network," *Advances in Neural Information Processing Systems*, vol. 2, pp. 396–404, 1990.

[35] W. Liu, "Large-margin soft max loss for convolutional neural networks," *ICML*, vol. 2, no. 3, p. 7, 2016.

[36] C. Szegedy, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, Boston, MA, USA, June 2015.

[37] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[38] J. Shijie, "Research on data augmentation for image classification based on convolution neural networks," in *Proceedings*

of the 2017 Chinese automation congress (CAC), pp. 4165–4170, IEEE, Jinan, China, October 2017.

[39] X. Glorot, B. Antoine, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323, Lauderdale, FL, USA, April 2011.

[40] H. F. Yu, F. L. Huang, and C. J. Lin, "Dual coordinate descent methods for logistic regression and maximum entropy models," *Machine Learning*, vol. 85, no. 1-2, pp. 41–75, 2011.

[41] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[42] Z. Xuegong, "Introduction to statistical learning theory and support vector machines," *Acta Automatica Sinica*, vol. 26, no. 1, pp. 32–42, 2000.

[43] R. Hecht-Nielsen, *Theory of the Backpropagation Neural Network, Neural Networks for Perception*, Academic Press, Cambridge, MA, USA, 1992.

[44] X. Chen, L. Qi, Y. Yang, Q. Luo et al., "Video-based detection infrastructure enhancement for automated ship recognition and behavior analysis," *Journal of Advanced Transportation*, vol. 2020, Article ID 7194342, 12 pages, 2020.

[45] X. Chen, Z. Li, Y. Yang, L. Qi, and R. Ke, "High-resolution vehicle trajectory extraction and denoising from aerial videos," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 5, pp. 3190–3202, 2021.

Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 5598390, 14 pages
https://doi.org/10.1155/2021/5598390

Research Article

CNN-Enabled Visibility Enhancement Framework for Vessel Detection under Haze Environment

Yuxu Lu,[1,2] **Yu Guo**,[1,2] **and Maohan Liang**[1,2]

[1]*Hubei Key Laboratory of Inland Shipping Technology, School of Navigation, Wuhan University of Technology, Wuhan, China*
[2]*National Engineering Research Center for Water Transport Safety, Wuhan, China*

Correspondence should be addressed to Maohan Liang; mhliang@whut.edu.cn

Received 16 January 2021; Revised 5 March 2021; Accepted 24 April 2021; Published 20 May 2021

Academic Editor: Yi-Sheng Lv

Maritime images captured under haze environment often have a terrible visual effect, making it easy to overlook important information. To avoid the failure of vessel detection caused by fog, it is necessary to preprocess the collected hazy images for recovering vital information. In this paper, a novel CNN-enabled visibility dehazing framework is proposed, consisting of two subnetworks, that is, Coarse Feature Extraction Module (C-FEM) and Fine Feature Fusion Module (F-FFM). Specifically, C-FEM is a multiscale haze feature extraction network, which can learn information from three scales. Correspondingly, F-FFM is an improved encoder-decoder network to fuse multiscale information obtained by C-FEM and enhance the visual effect of the final output. Meanwhile, a hybrid loss function is designed for monitoring the multiscale output of C-FEM and the final result of F-FFM simultaneously. It is worth mentioning that massive maritime images are considered the training dataset to further adapt the vessel detection task under haze environment. Comprehensive experiments on synthetic and realistic images have verified the superior effectiveness and robustness of our CNN-enabled visibility dehazing framework compared to several state-of-the-art methods. Our method preprocesses images before vessel detection to demonstrate our framework has the capacity of promoting maritime video surveillance.

1. Introduction

1.1. Background and Related Work. It is well known that the maritime surveillance system is an indispensable part of vessel traffic services [1]. As an efficient, convenient, and intuitive monitoring method, Closed Circuit Television (CCTV) is thus widely applied to critical regions, for example, ports and waterways. As shown in Figure 1, significant information in the images, however, is easily buried under the haze. Therefore, it is difficult for maritime regulatory authority to effectively extract detailed information (e.g., monitoring targets and water traffic conditions) from degraded images, which seriously affects maritime supervision efficiency. Besides, the low-quality images collected under haze environment have also brought severe challenges to intelligent surveillance methods based on vessel detection, recognition, and tracking [2–5]. To improve the maritime safety surveillance capability under haze environment, it is necessary to restore images under CCTV monitoring. In current literature, dehazing methods can be categorized into image enhancement-based methods, physical model-based methods, and deep learning-based methods.

1.1.1. Image Enhancement-Based Methods. Early research mainly enhanced the contrast of hazy images to highlight the scene characteristics of the interest region. Histogram Equalization (HE) [6] is a classic enhancement method devoted to enhancing the contrast by stretching the dynamic range of image pixel values. In current literature, HE-based methods can be divided into two categories, that is, global and local histogram equalization. Since the global histogram equalization can enhance the entire image by single mapping, it has the characteristics of simple principle and fast calculation. However, these methods often ignore the local information, resulting in the haze-free images having poor performance. To solve this problem, Stark et al. [7] proposed

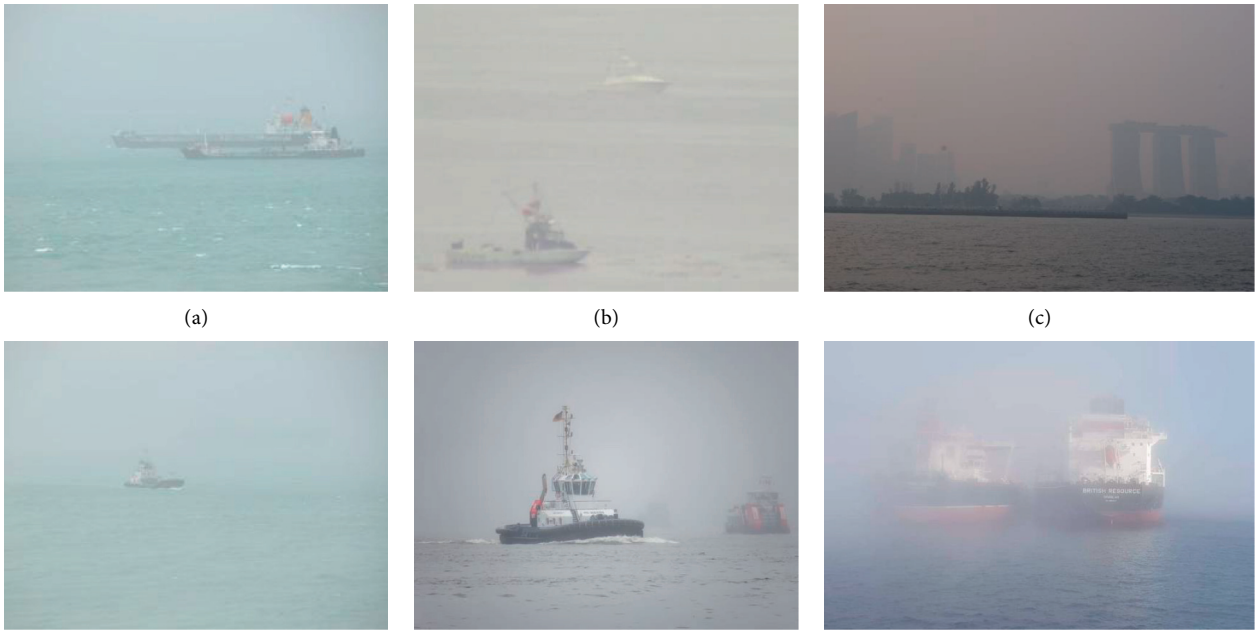


Figure 1: Maritime images captured under haze environment.

an adaptive local histogram equalization method. Subsequently, Kim et al. [8] proposed a nonoverlapping subblock histogram equalization method to reduce the blocky effect and computational complexity. Retinex theory-based image dehazing method is devoted to separating the illumination and reflection from the hazy image and enhancing the image by reducing the illumination impact. Jobson et al. [9] first used the Gaussian filter to obtain a smooth illumination according to the Retinex theory and thus proposed a single-scale Retinex (SSR). To avoid color distortion, Rahman et al. [10] proposed a multiscale Retinex algorithm with color restoration (MSRCR) by introducing a color compensation factor. To sum up, the image generated by these methods has higher contrast and color fidelity, but the halo often appears on the edge of the interest object.

1.1.2. Physical Model-Based Methods. These methods are proposed based on a certain physical model that describes the process of image degradation under haze weather. Because these methods use mathematical methods to describe the haze formation process based on light scattering, the final restored target is clear and natural. Physical model-based methods include the following categories, that is, depth-based method and prior-based method. The depth-based methods mainly obtain depth information through a specific method and then get stable model parameters. Finally, the potentially clear image can be obtained by the atmospheric scattering model. For instance, Oakley et al. [11] first used radar and other types of equipment to measure the shooting scene depth. Hautiere et al. [12] proposed an image dehazing algorithm based on the 3D geographic model for vehicle vision systems. Although these methods have an excellent dehazing effect, they heavily rely on distance measuring equipment. Therefore, Liu et al. [13] proposed a dehazing method to estimate the depth map through a second-order variational framework. In contrast, the prior-based method mainly analyzes haze formation and relies on specific prior information to achieve image dehazing. Dark channel prior (DCP) [14] and its improvements [15–17] have an excellent performance in the image dehazing task. Through numerous statistics on outdoor haze-free images, He et al. proposed DCP based on the assumption that most local color blocks contain some pixels with very low intensity in at least one color channel. Zhu et al. proposed a novel linear color attenuation prior [18], based on the difference between the brightness and the saturation of pixels within the hazy image. Subsequently, a nonlocal prior dehazing method [19] is employed to obtain the nonlocal transmission map from the haze-line property. To reduce halo and unnatural artifacts, a low-complexity color ellipsoid prior [20] is designed to accurately and swiftly estimate the transmission map. In current literature, several variational model-based transmission estimation methods [15, 21, 22] are also proposed. Although prior-based methods have verified excellent dehazing performance, they may cause a loss in color fidelity under certain circumstances and fail to obtain pleasing visual effects on maritime images.

1.1.3. Deep Learning-Based Method. Affected by the excellent results of machine learning technology in computer vision, the CNN-enabled dehazing method has gradually become a new research direction. For instance, Tang et al. [23] proposed a learning-based approach to systematically investigate different haze-relevant features and identify the best feature combination for image dehazing. Zhu et al. [18] designed the color attenuation prior model to estimate the

scene depth. Meanwhile, a supervised learning method is used to obtain the scene depth and the atmospheric scattering model parameters. However, this method fails to estimate the scene depth in the white environment. To further improve the deep learning-based dehazing methods performance, Cai et al. [24] first constructed a convolutional neural network (DehazeNet) to learn the mapping relationship between hazy images and transmission. DehazeNet employed artificially synthesized hazy images of different concentrations as the dataset. The trained DehazeNet can directly estimate the corresponding transmission and restore the potentially clear image according to the traditional atmospheric light scattering model. Ren et al. [25] proposed a Multiscale Convolutional Neural Network (MSCNN). MSCNN used the New York University indoor image depth database [26] to synthesize different hazy images as the dataset, making the hazy image more realistic. Subsequently, Zhao et al. [27] used outdoor scenes synthetic images and proposed a fully convolutional neural network model to estimate the transmission. Compared with DehazeNet and MSCNN, this network has better visual performance. However, it produces more parameters and calculations. To simplify the calculation, Li et al. [28] designed an end-to-end light-weight convolutional neural network (AOD-Net) that effectively balances calculation speed and visual effects. Inspired by image denoising, Du et al. [29] proposed a Deep Residual Learning (DRL) network to reconstruct the potential image. Besides, Chen et al. [30] proposed an end-to-end gated context aggregation network to directly restore the final haze-free image. It is worth noting that if the training datasets do not contain the geometric features presented in the haze-free target, it is usually difficult to produce satisfactory image quality. Therefore, it is necessary to design a CNN-enabled visibility enhancement framework for vessel detection under haze environment to further improve maritime video surveillance efficiency.

1.2. Contributions. This paper presents a CNN-enabled framework for practically solving vessel detection problem under haze environment. The main contribution of our method differs from others in the following aspects:

(i) A CNN-enabled visibility dehazing framework is proposed to improve the visibility of maritime images. Specifically, this framework includes a *Coarse Feature Extraction Module* (C-FEM) for capturing multiscale features and *Fine Feature Fusion Module* (F-FFM) for information fusion and enhancement.

(ii) To improve the generalization of the proposed network, we design a novel hybrid loss function to supervise the multiscale outputs of C-FEM and the final output of F-FFM simultaneously.

(iii) Image dehazing and vessel detection experiments under haze conditions are conducted to verify our superior performance compared to several state-of-the-art methods.

1.3. Construction. The remainder of this paper is divided into the following sections. Section 2 mainly describes the problem formulation related to the imaging model. In Section 3, a CNN-enabled visibility enhancement framework is proposed to improve the visual effect of hazy images. Implementation details and experiments are implemented in Section 4. Finally, we conclude our main contributions in Section 5.

2. Problem Formulation

2.1. Atmospheric Scattering Model. Video images collected by maritime video surveillance system under haze conditions often have poor visual quality. As shown in Figure 2, Narasimhan et al. [31] proposed the atmospheric scattering model to divide the light irradiance into the incident light attenuation part $J(x)e^{-\beta d(x)}$ and the atmospheric light imaging part $A_\infty(1-e^{-\beta d(x)})$. The incident light attenuation model considers that the reflected light by the vessel surface is scattered and attenuated by particulate impurities in the air, reducing the intensity of light reaching the imaging system. Note that as the propagation distance increases, the reflected light intensity decays exponentially. On the contrary, the atmospheric light imaging model believes that light intensity scattered by natural light enters the imaging system to participate in imaging. As the propagation distance increases, the scattered light intensity will gradually increase. Finally, the images collected by the imaging system under haze environments exhibit degradation phenomena such as low contrast, blurred images, and color distortion under the combined action of these two models. Mathematically, the atmospheric light scattering model can be expressed as

$$I(x) = J(x)e^{-\beta d(x)} + A_\infty\left(1 - e^{-\beta d(x)}\right), \tag{1}$$

where J and I, respectively, denote the hazy image and haze-free image, A_∞ and β represent the atmospheric light value and scattering coefficient, x is the image pixel index, and d is the distance between the scene point and the imaging system, that is, field depth. When we set $t(x) = e^{-\beta d(x)}$, equation (1) can thus be rewritten as follows:

$$I(x) = J(x)t(x) + A_\infty(1 - t(x)), \tag{2}$$

with t being the transmission. According to equation (2), the restoration haze-free image can be easily obtained by

$$J(x) = \frac{I(x) - A_\infty}{t(x)} + A_\infty. \tag{3}$$

2.2. Transformed Formula. According to equation (3), we can obtain a satisfactory haze-free image J by accurately estimating A_∞ and t. However, it is intractable to estimate two parameters simultaneously. For the sake of better performance of the end-to-end network, Li et al. [28] proposed the transformed atmospheric scattering model, which can be given by

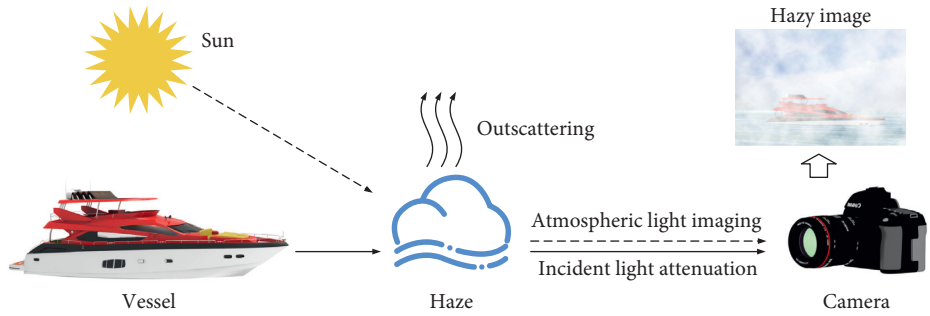


Figure 2: The principle of the atmospheric scattering model.

$$J(x) = \mathscr{K}(x)I(x) - \mathscr{K}(x) + 1, \tag{4}$$

where $\mathscr{K}$ is a particular parameter to integrate A_∞ and t; that is, $\mathscr{K}(x) = ((1/t(x))(I(x) - A_\infty) + A_\infty - 1/I(x) - 1)$. It is worth noting that hazy maritime images usually contain background (i.e., sky and water regions). Many statistical features-based methods, for example, DCP and maximum local contrast, often fail to obtain ideal transmission maps. Deep learning-based methods do not rely on these statistical features and can learn the mapping of hazy and haze-free images. Therefore, we will propose the CNN-enabled visibility enhancement network to effectively improve the quality of hazy maritime images and improve vessel detection accuracy.

3. CNN-Enabled Visibility Enhancement Framework

In this section, a CNN-enabled visibility enhancement framework is proposed to process hazy maritime images shown in Figure 3. This framework consists of two subnetworks, that is, *Coarse Feature Extraction Module* (C-FEM) and *Fine Feature Fusion Module* (F-FFM). In this work, C-FEM is introduced to learn multiscale hazy features. Meanwhile, F-FFM, an improved encoder-decoder network, is proposed to fuse and enhance the hazy image and the multiscale output obtained by C-FEM. Once our method gets the sharp image, it can easily detect the vessel containing the image by any target detection method.

$$I * w_{(t)} = \sum_{p+q=t} I(p)w(q), \tag{5}$$

$$I *_d w_{(t)} = \sum_{p+lq=t} I(p)w(q), \tag{6}$$

3.1. C-FEM. C-FEM is a module for initial extracting the features of the hazy image. In particular, C-FEM can perform mapping learning on three scales (i.e., 1, 1/4, and 1/16) to obtain coarse feature information with different resolutions simultaneously. Figure 4 shows the network architecture of C-FEM under one resolution, which is only composed of six convolutions. In this work, dilated convolution is embedded to increase the reception field of C-FEM. According to our research, dilated convolution can reduce the loss of spatial features without reducing the receptive field. However, the use of dilated convolution may increase the risk of spatially continuous information loss, destroying image feature information (especially edges). To alleviate the interference caused by dilated convolution, we combine standard convolution (Conv) [32] and dilated convolution (DConv) with improving the detailed information extraction ability. DConv can effectively solve this difficult problem with different receptive fields by adjusting the dilation rate value. Formally, standard convolution and dilated convolution are, respectively, defined as follows: where I is the discrete signal, w is convolution kernel, subscript $(\cdot)$ is the position of a discrete signal, $d \in Z^+$ is the dilation factor, and $*_d$ is dilated convolutions with a factor d. The only difference between standard convolution and dilated convolution is the influence of the dilation factor d on the multiplication position of $I(p)$ and $w(q)$. Dilated convolution benefits from $w(q)$ are no longer limited to a fixed receptive field, and the dilation factor d can be adjusted to have a larger receptive field. In this work, the method of fusing Conv and DConv can reduce the loss of spatial information caused by the excessive dilated rate and fully consider long- and short-distance information to present a better visual effect. Furthermore, Instance Normalization (IN) [33] and Rectified Linear Unit (ReLU) [34] are deployed after each Conv layer. Meanwhile, the feature map channels of the first five convolution outputs are set to 32.

$$\mathscr{L}^{\mathrm{MAE}}(\widehat{J}, J) = |\widehat{J} - J|. \tag{10}$$

According to our research, most deep learning-based dehazing methods rely on more complex network models to obtain better visual effects. When the model is relatively simple, it is usually hard to learn the fog feature, causing information damage to potential images. In contrast, the imaging model $J(x) = \mathscr{K}(x)I(x) - \mathscr{K}(x) + 1$ reduces the algorithm complexity, making it easier for the network to extract information. The introduction of this model makes it possible for a simple network model to extract potential multiscale features from the original image. It is worth noting that the output of C-FEM only introduced to provide a prior is not used as the final result. Simultaneously, it has a faster calculation speed and can satisfy the needs of real-time processing.

3.2. F-FFM. Coarse feature maps of three resolutions (i.e., 1, 1/4, and 1/16) have been obtained by C-FEM, which

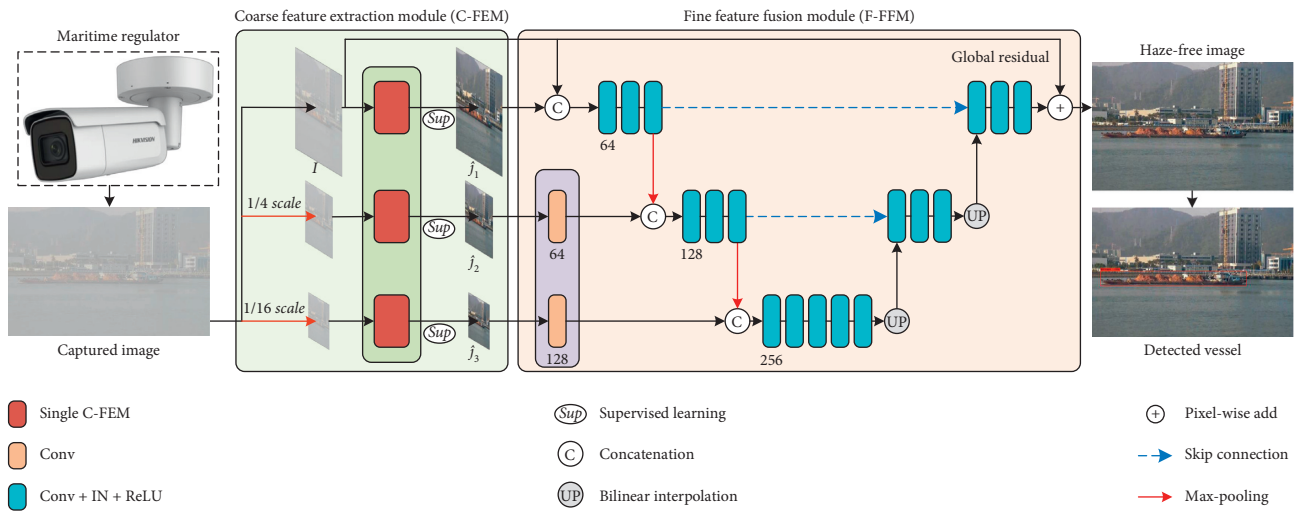


Figure 3: The flowchart of our method. From top to bottom in C-FEM: subimages reduced to 1, 1/4, and 1/16 of the original image, respectively.

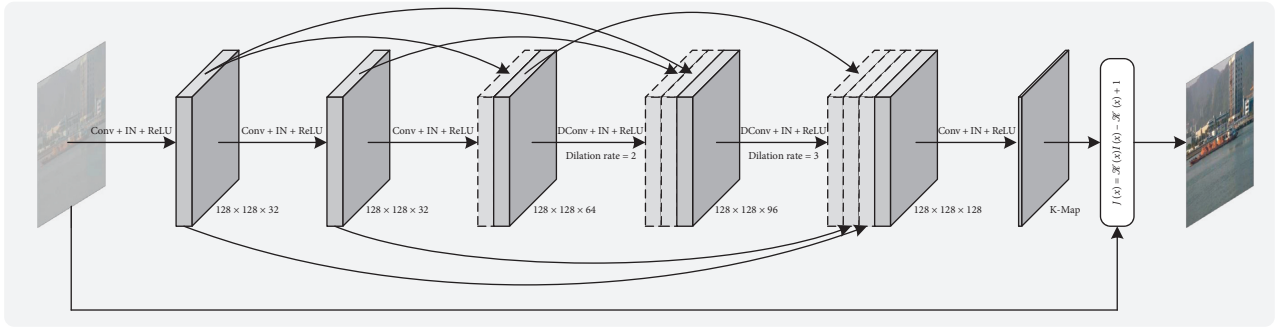


Figure 4: Network architecture of single C-FEM.

contains most of the potentially clear image information. The feature map information obtained by a standard encoder-decoder CNN is usually found irregularly. When the prior information obtained by C-FEM is introduced to the encoder, we believe that F-FFM can obtain better parameters and accelerate the convergence speed. When the feature maps are fused at different scales, the deep network can further extract edge detail information. Table 1 shows that the architecture of Fine Feature Fusion Modul (F-FFM) is a special encoder-decoder structure. Specifically, F-FFM only performs two downsampling operations and merges with the corresponding output of C-FEM. Both the encoder and the decoder consist of the same module, that is, a 3×3 convolution filter (Conv) [32], Instance Normalization (IN) [33], and Rectified Linear Unit (ReLU) [34]. Maximum pooling and bilinear interpolation are exploited to perform down- and upsampling operations on the feature map, respectively. Different from traditional encoder-decoder structures, our F-FFM encoder integrates the output of C-FEM. This strategy can guide F-FEM to learn the mapping of hazy images and haze-free targets. To better preserve the boundary details of the input, we adopt a global skip connection strategy to further ensure the details of the output image. In other words, the output of the last convolution and the input image is directly added as the output of F-FFM, and we find that it can significantly improve the dehazing effect through comparative experiments.

3.3. Loss Function. To robustly learn the multiscale mapping relationship between hazy image and haze-free image, a specific loss function $\mathscr{L}^{C-\mathrm{FEM}}$ is proposed. As shown in Figure 3, C-FEM has three scale outputs (i.e., $\widehat{J}_1$, $\widehat{J}_2$, and $\widehat{J}_3$). These three images sequentially have 1, 1/4, and 1/16 of the original image size. Subsequently, the maximum pooling operation is used to obtain clear images with three scales named J_1, J_2, and J_3, which, respectively, correspond to the scale of $\widehat{J}_1$, $\widehat{J}_2$, and $\widehat{J}_3$. In this work, Mean Square Error (MSE) loss function is employed to constrain each scale output of C-FEM; that is,

$$\mathscr{L}^{\mathrm{C-FEM}} = \lambda_1 \mathscr{L}^{\mathrm{MSE}}\left(\widehat{J}_1, J_1\right) + \lambda_2 \mathscr{L}^{\mathrm{MSE}}\left(\widehat{J}_2, J_2\right) + \lambda_3 \mathscr{L}^{\mathrm{MSE}}\left(\widehat{J}_3, J_3\right), \tag{7}$$

where $\mathscr{L}^{\mathrm{MSE}}(\widehat{J}_*, J_*) = (\widehat{J}_* - J_*)^2$, λ_1, λ_2, and λ_3 are trade-off parameters of corresponding loss functions. To further preserve the high-frequency details of the potential haze-free image while eliminating boundary artifacts, a hybrid loss function $\mathscr{L}^{F-\mathrm{FFM}}$ is introduced to limit the ground truth J and the predicted restored image $\widehat{J}$; that is,

Table 1: Network architecture of F-FFM.

Input	Layers	Type	Channels	Filter	Stride	Output	Size
$I + \widehat{J}_1$	1	Convolutional	64	3×3	1	$\downarrow_1$	128×128
$\downarrow_1$	3	Convolutional	64	3×3	1	$\downarrow_2$	128×128
$\downarrow_2$		Max pooling	64	3×3	2	$\downarrow_3$	
$\downarrow_3 + \widehat{J}_2$	1	Convolutional	128	3×3	1	$\downarrow_4$	64×64
$\downarrow_4$	3	Convolutional	128	3×3	1	$\downarrow_5$	64×64
$\downarrow_5$		Max pooling	128	3×3	2	$\downarrow_6$	
$\downarrow_6 + \widehat{J}_3$	1	Convolutional	256	3×3	1	$\downarrow_7$	32×32
$\downarrow_7$	5	Convolutional	256	3×3	1	$\downarrow_8$	32×32
$\downarrow_8$		Bilinear interpolation				$\downarrow_9$	
$\downarrow_9 + \downarrow_4$		Skip connection	128	1×1	1	$\downarrow_{10}$	64×64
$\downarrow_{10}$	1	Convolutional	128	3×3	1	$\downarrow_{11}$	64×64
$\downarrow_{11}$	3	Convolutional	128	3×3	1	$\downarrow_{12}$	64×64
$\downarrow_{12}$		Bilinear interpolation				$\downarrow_{13}$	
$\downarrow_{13} + \downarrow_1$		Skip connection	64	1×1	1	$\downarrow_{14}$	128×128
$\downarrow_{14}$	1	Convolutional	64	3×3	1	$\downarrow_{15}$	128×128
$\downarrow_{15}$	3	Convolutional	64	3×3	1	$\downarrow_{16}$	128×128
$\downarrow_{16} + I$		Global residual	3	3×3	1	$\widehat{J}$	128×128

$$\begin{aligned}\mathscr{L}^{F-\mathrm{FFM}} &= \lambda_4 \mathscr{L}^{\mathrm{MS-SSIM}}(\widehat{J}, J) + \lambda_5 \mathscr{L}^{\mathrm{MAE}}(\widehat{J}, J) \\ &+ \lambda_6 \mathscr{L}^{\mathrm{Edge}}(\widehat{J}, J) + \lambda_7 \mathscr{L}^{\mathrm{TV}}(\widehat{J}),\end{aligned} \tag{8}$$

with λ_4, λ_5, λ_6, and λ_7 being the penalty weights. Multiscale structural similarity (MS-SSIM) [35] is firstly employed to constrain the structure, brightness, and contrast of the image. The MS-SSIM loss function can be defined as follows:

$$\mathscr{L}^{\mathrm{MS-SSIM}}(\widehat{J}, J) = 1 - \mathrm{MSSSIM}(\widehat{J}, J), \tag{9}$$

with MSSSIM being the calculation operation of the multiscale structural similarity index between two images. The hazy image inevitably has a low contrast phenomenon in local regions, resulting in color distortion. To solve this problem, the Mean Absolute Error (MAE) loss function $\mathscr{L}^{MAE}$ is introduced as a part of $\mathscr{L}^{F-FFM}$, which can reduce the color distortion problem to a certain extent. In particular, $\mathscr{L}^{MAE}$ is defined as

The high-frequency detail information is easily destroyed in the process of image dehazing. To further improve the fidelity and authenticity of details, we propose an additional edge loss function [36] to limit the high-frequency components, for example, edge and texture. $\mathscr{L}^{\mathrm{Edge}}$ can be written as

$$\mathscr{L}^{\mathrm{Edge}}(\widehat{J}, J) = \sqrt{(\mathrm{Lap}(\widehat{J}) - \mathrm{Lap}(J))^2 + \varepsilon^2}, \tag{11}$$

where $\mathrm{Lap}(J)$ and $\mathrm{Lap}(\widehat{J})$ represent edge maps extracted from J and $\widehat{J}$ via the Laplacian operator, respectively. The penalty coefficient ε is empirically set to 10^{-3}. In addition, the Total Variation (TV) loss function [37] is exploited to suppress the pixel-jump problem, which can be given bywhere ∇_h and ∇_v represent the operators of the horizontal and vertical gradients, respectively. We refer interested readers to [35–37] for more details on calculations of MS-SSIM, edge loss, and TV. To sum up, the total loss function can be written as follows:

$$\mathscr{L}^{\mathrm{Total}} = \gamma_1 \mathscr{L}^{\mathrm{C-FEM}} + \gamma_2 \mathscr{L}^{F-\mathrm{FFM}}, \tag{13}$$

where γ_1 and γ_2 are the penalty coefficient of $\mathscr{L}^{C-FEM}$ and $\mathscr{L}^{F-FFM}$. By comparative experiment, we manually selected the optimal weight of all loss functions; that is, $\lambda_1 = 1, \lambda_2 = 1$, $\lambda_3 = 1$, $\lambda_4 = 0.8$, $\lambda_5 = 0.2$, $\lambda_6 = 0.05$, $\lambda_7 = 0.05$, $\gamma_1 = 1$, and $\gamma_2 = 1$.

4. Experimental Results and Analysis

This section will describe all the implementation details of network training, including dataset construction and network parameter settings. We will compare our method with several state-of-the-art dehazing methods on both synthetic and realistic hazy maritime images. To prove that our method can improve detection accuracy, our proposed framework will be employed in vessel detection tasks under haze environment.

4.1. Comparison Methods and Evaluation Indicators. Our method will be compared with four handcrafted prior-based methods and three deep learning-based methods. For the sake of fair comparison, the parameters of other competing dehazing methods are provided by the authors' code.

(1) DCP: *Dark Channel Prior-Based Method* [14]. Through numerous statistics on outdoor haze-free images, DCP is proposed based on the assumption that most local color blocks contain some pixels with very low intensity in at least one color channel. According to this statistic prior and the haze imaging model, a high-quality haze-free image can be directly obtained.

(2) GRM: *Gradient Residual Minimization-Based Method* [16]. This method first proposes the depth-edge-aware smoothing algorithm to refine the transmission map generated by local priors. Meanwhile, Gradient Residual Minimization (GRM) is

introduced during the image recovery process. By comparison, the GRM-based method can jointly recover the haze-free image and explicitly minimize possible visual artifacts in it.

(3) HL: *Haze-Lines-Based Method* [38]. This method finds that the pixel values of a hazy image can be modeled as lines intersecting at the air light. Based on this prior condition, a novel haze-lines-based method is proposed to restore the hazy image better. It is worth noting that the complexity of HL is linear in the number of pixels, having higher computational efficiency.

(4) F-LDCP: *Fusion of Luminance and Dark Channel Prior-Based Method* [39]. To make the sky region more natural in long-shot images, a Fusion of Luminance and Dark Channel Prior (F-LDCP) method is proposed. The transmission maps estimated by the brightness model and the DCP model are fused through a soft segmentation.

(5) MSCNN: *Multiscale Convolutional Neural Networks* [25]. To learn the practical features of a hazy image, a multiscale deep network (MSCNN) is designed to address the image dehazing problem. MSCNN can be divided into the coarse-scale network and fine-scale network. The coarse-scale network can learn a holistic estimation of the scene transmission, and the fine-scale network is used to optimize the obtained transmission. Finally, the haze-free image can be obtained by the atmospheric scattering model.

(6) AOD-Net: *All-in-One Dehazing Network* [28]. AOD-Net, a light-weight CNN, is designed according to the reformulated atmospheric scattering model. This network replaces the atmospheric light value and transmission with one parameter. It is worth mentioning that AOD-Net has been embedded in other deeper models (e.g., Faster R-CNN) to improve the advanced tasks of hazy images.

(7) GCA-Net: *Gated Context Aggregation Network* [30]. GCA-Net is an end-to-end Gated Context Aggregation Network. In particular, the latest smoothed dilation technology is designed to eliminate gridding artifacts caused by the extensive-used dilated convolution with negligible additional parameters.

$$\mathscr{L}^{\mathrm{TV}}(\widehat{J}, J) = \left(\nabla_h \widehat{J}\right)^2 + \left(\nabla_v \widehat{J}\right)^2, \tag{12}$$

In synthetic and realistic experiments, we will compare these methods with our proposed method. In addition, three full-reference indicators, that is, Peak-Signal-to-Noise Ratio (PSNR) [40], SSIM [41], and Feature Similarity (FSIM) [42], are introduced to evaluate the dehazing performance in the synthetic experiment. Meanwhile, one popular no-reference image quality assessment method, that is, Natural Image Quality Evaluator (NIQE) [43], is also exploited to perform dehazing quality evaluation in the real experiment. Theoretically, higher values of PSNR, SSIM, FSIM, and lower values of NIQE indicate better visual performance.

4.2. Experimental Datasets and Settings. To guarantee high-quality dehazing results, we tend to select 7000 haze-free maritime images as the dataset and randomly cropped these images into 34000 patches of size 256×256. In this work, our network is trained for 80 epochs. The learning rate of the first 40 epochs is 10^{-3} and the learning rate of the last 40 epochs is 10^{-4} to increase the convergence rate. In each epoch, the hazy synthetic versions are obtained by equation (2), that is, atmospheric scattering model. In particular, the transmission t and atmospheric light value A_∞ are random constants ranging between (0.2, 0.6) and (0.8, 0.9). All numerical experiments and model training are conducted in Python 3.7 and Matlab2019a environment running on a PC with Intel(R) Core (TM) i7-9750H CPUa 2.60 GHz and a Nvidia GeForce GTX 2080Ti GPU. It takes about 10 hours to train our network with the Pytorch package [44]. The Python source code is available at https://github.com/LouisYuxuLu/JAT_Dehazing.

4.3. Experiments on Synthetic Maritime Datasets. This subsection is devoted to comparing our proposed method with seven popular dehazing methods, that is, DCP [14], GRM [16], HL [38], F-LDCP [39], MSCNN [25], AOD-Net [28], and GCA-Net [30]. In this work, six clear maritime images are exploited to conduct synthetic experiments shown in Figure 5. In particular, we tend to produce 36 degraded images by setting $t \in \{0.2, 0.4, 0.6\}$ and $A_\infty \in \{0.8, 0.9\}$ according to equation (2). To quantitatively evaluate the dehazing performance, three full-reference metrics (i.e., PSNR, SSIM, and FSIM) are employed in this experiment.

For the sake of better visual comparisons, the dehazing versions of hazy images with different degrees obtained by various methods are shown in Figure 6. It can be clearly observed that DCP and HL often make the color unnatural. Meanwhile, due to the incomplete dehazing, the results obtained by DCP easily suffer from the interference of boundary artifacts around the object. Although GRM can get satisfactory visual effects, it requires complex calculations and has the risk of excessive smoothness. F-LDCP can excellently solve the blocking artifacts and halo problems in the sky regions, but the color fidelity in the water regions needs improvement. MSCNN and AOD-Net can handle the low-concentration hazy image. However, the restored versions of the high-concentration hazy images (i.e., hazy images with $t = 0.2$) usually have a poor visual effect. GCA-Net fails in the synthetic experiment, resulting in a nonuniform distribution of fog remaining in the results. By comparison, our method can not only make the restored image visually more natural but also ensure the color reproduction of the sky and water regions.

To further confirm the superiority of our method, the quantitative results of PSNR, SSIM, and FSIM are shown in Figure 7 and Table 2. PSNR, SSIM, and FSIM values are

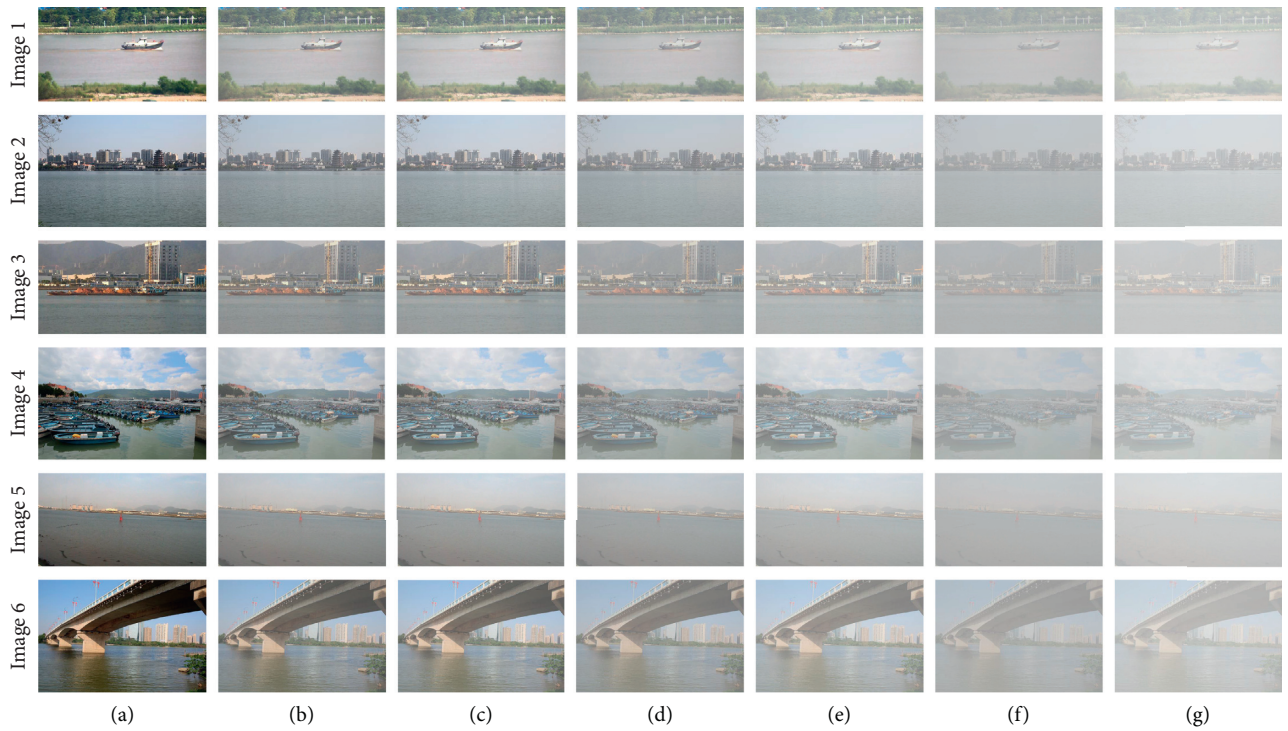


Figure 5: Six different original sharp maritime images selected to generate synthetic hazy images. From left to right: (a) original images; hazy images with (b) $A_\infty = 0.8/t = 0.6$, (c) $A_\infty = 0.9/t = 0.6$, (d) $A_\infty = 0.8/t = 0.4$, (e) $A_\infty = 0.9/t = 0.4$, (f) $A_\infty = 0.8/t = 0.2$, and (g) $A_\infty = 0.9/t = 0.2$, respectively.

illustrated using box-plot in Figure 7. It can be seen that our method has higher index values in most cases. Particularly for high-concentration hazy images, our method can stably obtain high-quality restored versions. Besides, Table 2 shows three metrics value comparisons of various image enhancement methods on 36 hazy images. In particular, we display the best result of each metric in bold. Due to the highest values of PSNR, SSIM, and FSIM, our method has the best dehazing performance. Meanwhile, the standard deviation calculated by the SSIM and FSIM is the smallest, which verifies that our method has excellent robustness.

4.4. Experiments on Realistic Maritime Datasets. This subsection will verify the reliability of several methods in realistic hazy maritime images due to the distinctness between synthetic and realistic versions. Meanwhile, NIQE is introduced to describe the naturalness of visual effects quantitatively, and our proposed method will be compared with seven dehazing methods, that is, DCP [14], GRM [16], HL [38], F-LDCP [39], MSCNN [25], AOD-Net [28], and GCA-Net [30]. Figure 8 shows several dehazing results to reflect the imaging performance more intuitively.

From the visual comparisons, DCP and HL have serious color distortion problems and blocking effects in the sky regions. Recovery results obtained by GRM have the risk of low contrast, especially in the recovery task of Image 9. F-LDCP and AOD-Net fail to correct the color of the image. GCA-Net not only has the problem of overexposure in the sky region but also has nonuniform fog remaining in the image. Although MSCNN has better visual effects than other methods, our method has pleasing color and can remove fog more fully. Our superior performance can be further confirmed by the quantitative results NIQE shown in Table 3.

4.5. Experiments on Vessel Detection under Haze Environment. In the maritime imaging system, the harsh imaging environment severely restricts the regular operation of the visible light imaging sensor, reduces vessel detection accuracy, and leads to incorrect identification. To prove this phenomenon, we, respectively, used YOLOv4 [45] and Faster-RCNN [46] to detect vessels in haze and haze-free images. As shown in Figure 9, it is easily found that the haze image has low contrast and massive useful information is obscured, which leads to problems, for example, identification errors or missing identification during the target detection process. After dehazing, the vessel target is effectively captured and recognized, and the recognition accuracy is significantly increased. Therefore, dehazing the degraded hazy image by our method can improve vessel detection performance. The computer and the related workers can make correct decisions in time.

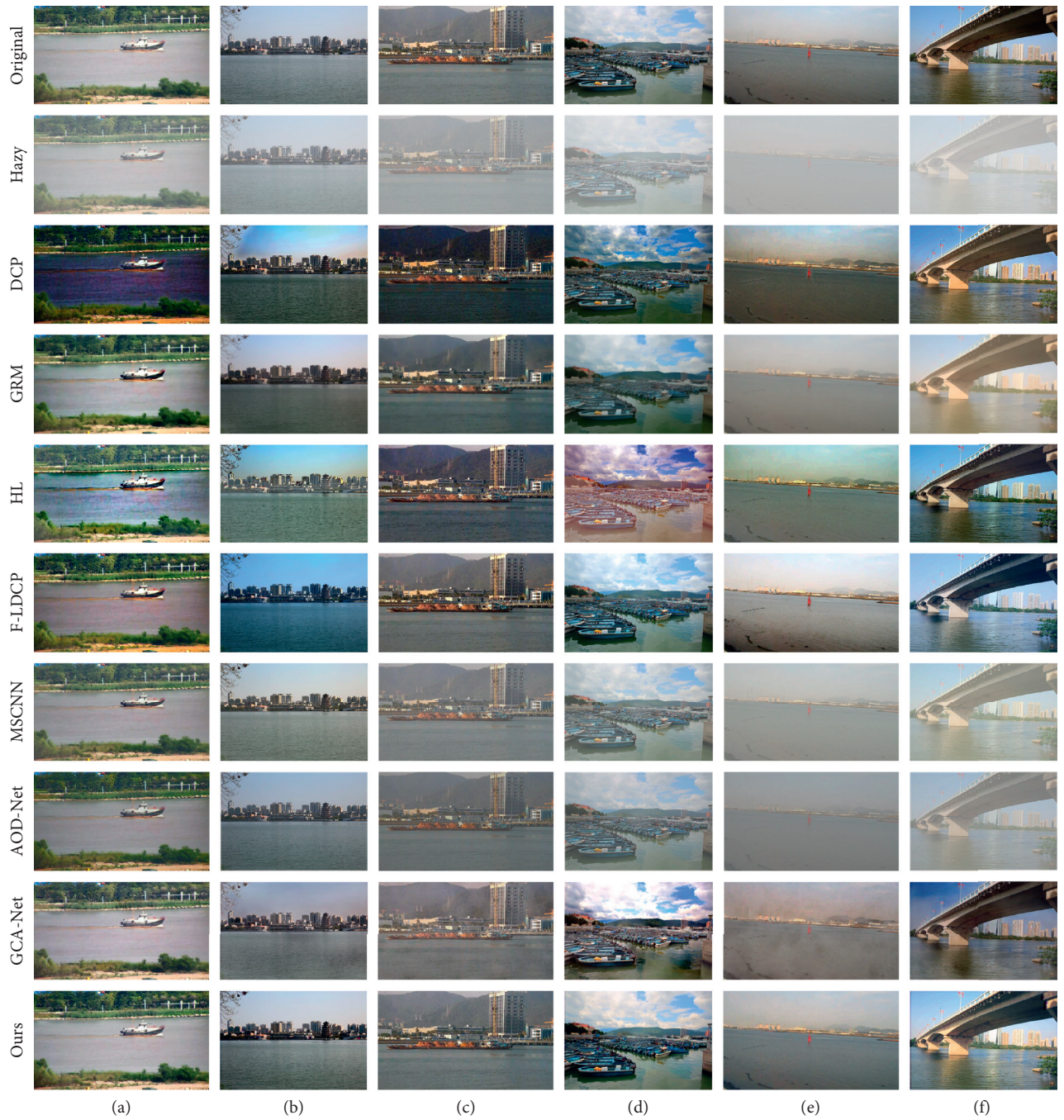
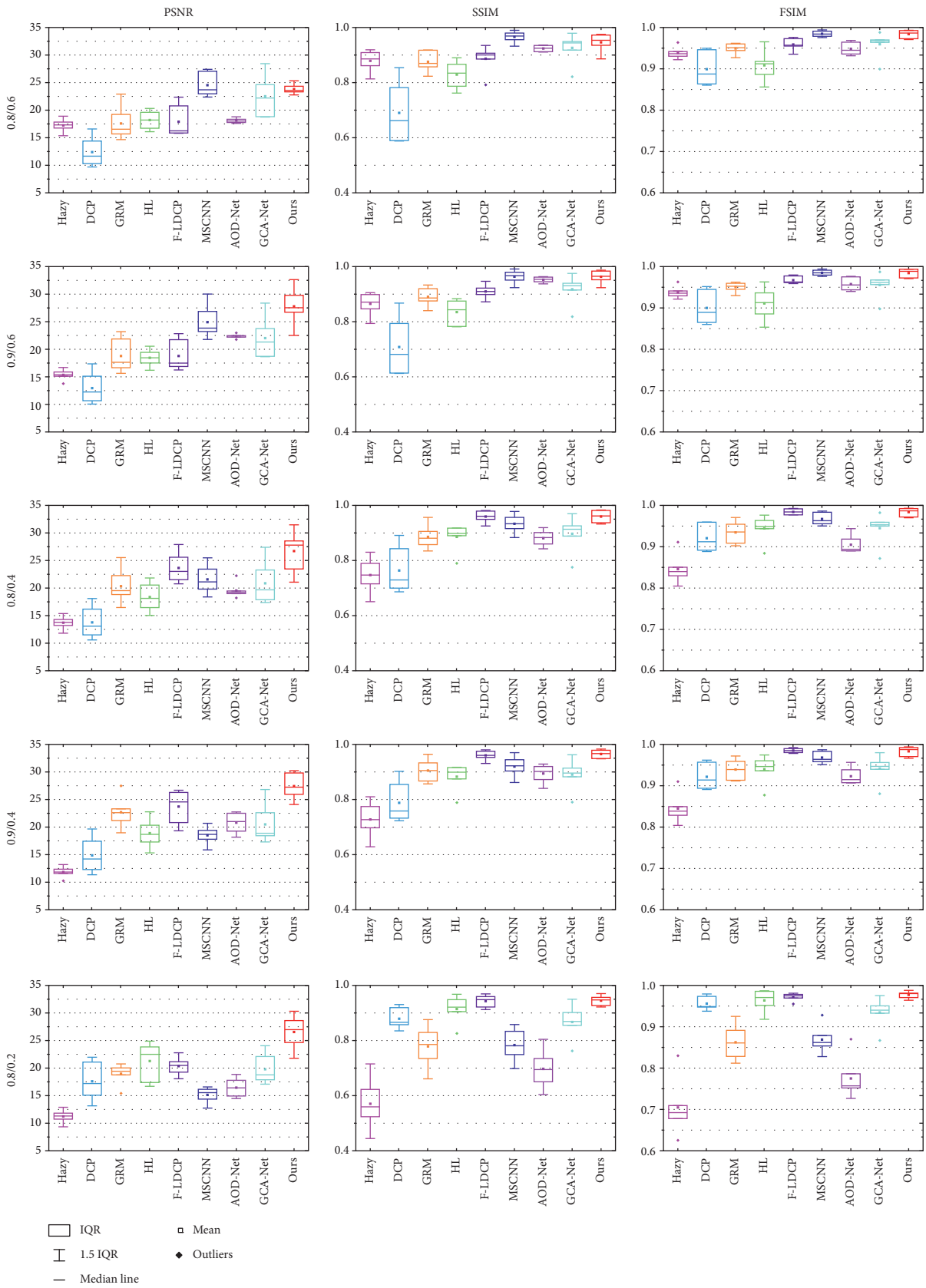
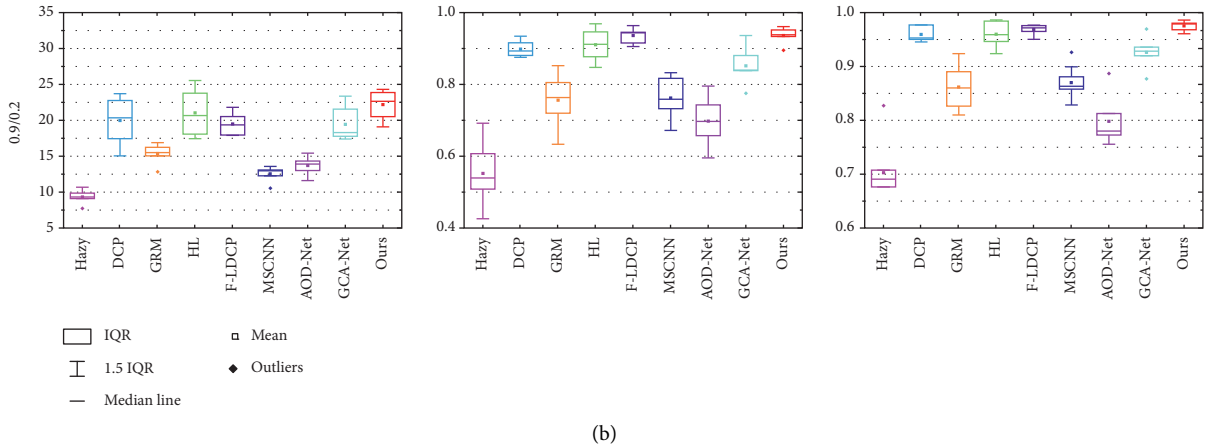


Figure 6: Comparisons of synthetic experiments on six typical hazy images in Figure 5. Each column represents the hazy image synthesized by different combinations of A_∞ and t, the original image corresponding to the hazy image, and the dehazing results obtained by all methods. From left to right in the second line: (a) hazy Image 1 with $A_\infty = 0.8/t = 0.6$, (b) hazy Image 2 with $A_\infty = 0.9/t = 0.6$, (c) hazy Image 3 with $A_\infty = 0.8/t = 0.4$, (d) hazy Image 4 with $A_\infty = 0.9/t = 0.4$, (e) hazy Image 5 with $A_\infty = 0.8/t = 0.2$, and (f) hazy Image 6 with $A_\infty = 0.9/t = 0.2$, respectively.



(a)

Figure 7: Continued.



(b)

Figure 7: The box-plot of PSNR, SSIM, and FSIM values of various methods on all hazy images shown in Figure 5. From top to bottom: the synthetic hazy experiment with $A_\infty = 0.8/t = 0.6$, $A_\infty = 0.9/t = 0.6$, $A_\infty = 0.8/t = 0.4$, $A_\infty = 0.9/t = 0.4$, $A_\infty = 0.8/t = 0.2$, and $A_\infty = 0.9/t = 0.2$, respectively. Note that IQR represents interquartile range.

Table 2: PSNR, SSIM, and FSIM comparisons (mean ± std) of various image enhancement methods on all test images shown in Figure 5. The best results are highlighted in bold.

Methods	PSNR	SSIM	FSIM
Hazy	13.17 ± 2.90	0.727 ± 0.138	0.830 ± 0.105
DCP	15.23 ± 2.93	0.789 ± 0.086	0.933 ± 0.022
GRM	18.96 ± 3.21	0.845 ± 0.070	0.912 ± 0.046
HL	19.46 ± 1.71	0.874 ± 0.042	0.937 ± 0.027
F-LDCP	20.58 ± 3.00	0.933 ± 0.032	0.972 ± 0.011
MSCNN	19.71 ± 5.19	0.890 ± 0.091	0.940 ± 0.055
AOD-net	18.70 ± 3.30	0.848 ± 0.113	0.889 ± 0.079
GCA-net	21.26 ± 1.38	0.899 ± 0.027	0.947 ± 0.012
Ours	25.80 ± 3.01	0.954 ± 0.014	0.982 ± 0.003

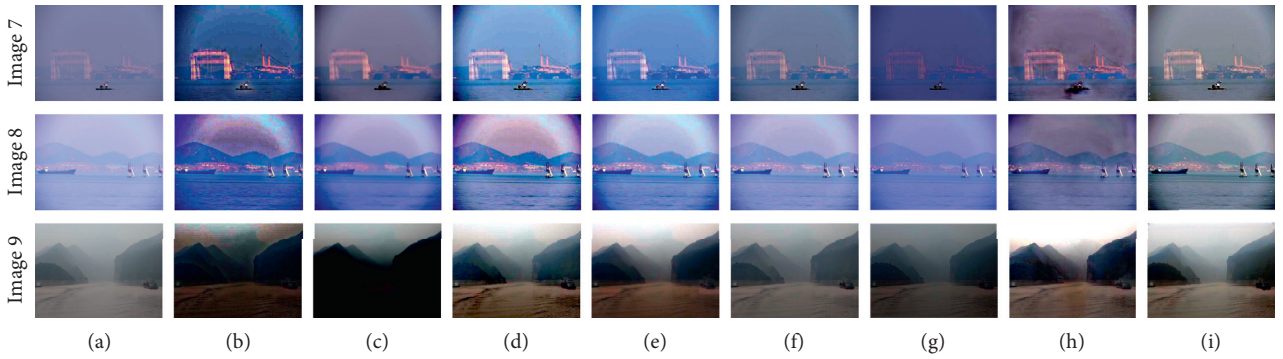


Figure 8: Comparisons of realistic experiments on three hazy maritime images. (a) Hazy, (b) DCP, (c) GRM, (d) HL, (e) F-LDCP, (f) MSCNN, (g) AOD-Net, (h) GCA-Net, and (i) ours.

Table 3: NIQE comparison of various image enhancement methods on all test images shown in Figure 6. The best results are highlighted in bold.

	DCP	GRM	HL	F-LDCP	MSCNN	AOD-Net	GCA-Net	Ours
Image 7	5.448	5.780	5.296	5.932	6.224	7.220	5.080	5.057
Image 8	5.525	10.297	4.961	5.868	5.826	5.716	5.694	5.122
Image 9	4.856	5.281	5.204	5.330	5.255	5.409	4.626	5.103
Average	5.276 ± 0.299	7.119 ± 2.299	5.154 ± 0.141	5.710 ± 0.270	5.768 ± 0.398	6.115 ± 0.792	5.133 ± 0.438	5.094 ± 0.027

Figure 9: Vessel detection experiments under haze environment. (a) Hazy image, (b) hazy image (YOLOv4), (c) hazy image (Faster-RCNN), (d) dehazed image, (e) dehazed image (YOLOv4), and (f) dehazed image (Faster-RCNN).

5. Conclusion

In this paper, a novel CNN-enabled visibility dehazing framework was proposed, which could significantly improve the visual effect of images captured by the maritime camera under haze environment. In particular, this framework is composed of two subnetwork named Coarse Feature Extraction Module (C-FEM) and Fine Feature Fusion Module (F-FFM). C-FEM is an initial multiscale feature extraction network containing three simple six-layer convolutional networks, that is, Single C-FEM. C-FEM can obtain coarse feature maps from 1, 1/4, and 1/16 of the original image pixel size. F-FFM is a special encoder-decoder structure used to fuse and enhance the multiscale information obtained by C-FEM and original hazy image. To further improve the network performance, a corresponding loss function is proposed to simultaneously supervise the multiscale output of C-FEM and the final result of F-FFM. Furthermore, our dataset contains massive maritime images to complete the vessel detection task under haze environment successfully. Both qualitative and quantitative experiments have illustrated the effectiveness of our proposed framework.

Data Availability

The image data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Yuxu Lu and Yu Guo are co-first authors.

Acknowledgments

This work was supported by the National Key R&D Program of China (No. 2018YFC0309602).

References

[1] R. W. Liu, J. Nie, S. Garg, Z. Xiong, Y. Zhang, and M. S. Hossain, "Data-driven trajectory quality improvement for promoting intelligent vessel traffic services in 6G-enabled maritime IoT systems," *IEEE Internet of Things Journal*, vol. 8, pp. 1–12, 2021.

[2] X. Chen, L. Qi, Y. Yang et al., "Video-based detection infrastructure enhancement for automated ship recognition and behavior analysis," *Journal of Advanced Transportation*, vol. 2020, Article ID 7194342, 12 pages, 2020.

[3] X. Chen, X. Xu, Y. Yang, H. Wu, J. Tang, and J. Zhao, "Augmented ship tracking under occlusion conditions from maritime surveillance videos," *IEEE Access*, vol. 8, pp. 42884–42897, 2020.

[4] X. Chen, S. Wang, C. Shi, H. Wu, J. Zhao, and J. Fu, "Robust ship tracking via multi-view learning and sparse representation," *Journal of Navigation*, vol. 72, no. 1, pp. 176–192, 2019.
[5] X. Chen, X. Xu, Y. Yang, Y. Huang, J. Chen, and Y. Yan, "Visual ship tracking via a hybrid kernelized correlation filter and anomaly cleansing framework," *Applied Ocean Research*, vol. 106, Article ID 102455, 2020.
[6] E. D. Pisano, S. Zong, B. M. Hemminger et al., "Contrast limited adaptive histogram equalization image processing to improve the detection of simulated spiculations in dense mammograms," *Journal of Digital Imaging*, vol. 11, no. 4, pp. 193–200, 1998.
[7] J. A. Stark, "Adaptive image contrast enhancement using generalizations of histogram equalization," *IEEE Transactions on Image Processing*, vol. 9, no. 5, pp. 889–896, 2000.
[8] T. K. Kim, J. K. Paik, and B. S. Kang, "Contrast enhancement system using spatially adaptive histogram equalization with temporal filtering," *IEEE Transactions on Consumer Electronics*, vol. 44, no. 1, pp. 82–87, 1998.
[9] D. J. Jobson, Z. Rahman, and G. A. Woodell, "Properties and performance of a center/surround retinex," *IEEE Transactions on Image Processing*, vol. 6, no. 3, pp. 451–462, 1997.
[10] Z. Rahman, D. J. Jobson, and G. A. Woodell, "Multi-scale retinex for color image enhancement," in *Proceedings of the IEEE International Conference on Image Processing*, vol. 3, pp. 1003–1006, Lausanne, Switzerland, September 1996.
[11] J. P. Oakley and B. L. Satherley, "Improving image quality in poor visibility conditions using a physical model for contrast degradation," *IEEE Transactions on Image Processing*, vol. 7, no. 2, pp. 167–179, 1998.
[12] N. Hautiere, J. P. Tarel, and D. Aubert, "Towards fog-free in-vehicle vision systems through contrast restoration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, Minneapolis, MN, USA, June 2007.
[13] R. W. Liu, S. Xiong, and H. Wu, "A second-order variational framework for joint depth map estimation and image dehazing," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1433–1437, Calgary, Canada, September 2018.
[14] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 12, pp. 2341–2353, 2011.
[15] Q. Shu, C. Wu, R. W. Liu, K. T. Chui, and S. Xiong, "Two-phase transmission map estimation for robust image dehazing," in *Proceedings of the International Conference on Neural Information Processing*, pp. 529–541, Siem Reap, Cambodia, December 2018.
[16] C. Chen, M. N. Do, and J. Wang, "Robust image and video dehazing with visual artifact suppression via gradient residual minimization," in *Proceedings of the European Conference on Computer Vision*, pp. 576–591, Amsterdam, The Netherlands, October 2016.
[17] Q. Liu, X. Gao, L. He, and W. Lu, "Single image dehazing with depth-aware non-local total variation regularization," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 5178–5191, 2018.
[18] Q. Zhu, J. Mai, and L. Shao, "A fast single image haze removal algorithm using color attenuation prior," *IEEE Transactions on Image Processing: A Publication of the IEEE Signal Processing Society*, vol. 24, no. 11, pp. 3522–3533, 2015.
[19] D. Berman and S. Avidan, "Non-local image dehazing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1674–1682, Las Vegas, NV, USA, June 2016.
[20] T. M. Bui and W. Kim, "Single image dehazing using color ellipsoid prior," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 999–1009, 2018.
[21] Q. Shu, C. Wu, Z. Xiao, and R. W. Liu, "Variational regularized transmission refinement for image dehazing," in *Proceedings of the IEEE International Conference on Image Processing*, pp. 2781–2785, Taipei, Taiwan, August 2019.
[22] Q. Shu, C. Wu, Q. Zhong, and R. W. Liu, "Alternating minimization algorithm for hybrid regularized variational image dehazing," *Optik*, vol. 185, pp. 943–956, 2019.
[23] K. Tang, J. Yang, and J. Wang, "Investigating haze-relevant features in a learning framework for image dehazing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2995–3000, Columbus, GA, USA, September 2014.
[24] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao, "Dehazenet: an end-to-end system for single image haze removal," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5187–5198, 2016.
[25] W. Ren, S. Liu, H. Zhang, J. Pan, X. Cao, and M.-H. Yang, "Single image dehazing via multi-scale convolutional neural networks," in *Proceedings of the European Conference on Computer Vision*, pp. 154–169, Cham, Amsterdam, October 2016.
[26] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proceedings of the European Conference on Computer Vision*, pp. 746–760, Berlin, Germany, October 2012.
[27] X. Zhao, K. Wang, Y. Li, and J. Li, "Deep fully convolutional regression networks for single image haze removal," in *Proceedings of the IEEE Visual Communications and Image Processing*, pp. 1–4, St. Petersburg, FL, USA, December 2017.
[28] B. Li, X. Peng, Z. Wang, J. Xu, and D. Feng, "AOD-net: all-in-one dehazing network," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4770–4778, Venice, Italy, October 2017.
[29] Y. Du and X. Li, "Recursive deep residual learning for single image dehazing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 730–737, Salt Lake City, UT, USA, June 2018.
[30] D. Chen, M. He, Q. Fan et al., "Gated context aggregation network for image dehazing and deraining," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pp. 1375–1383, Waikoloa Village, HI, USA, January 2019.
[31] S. G. Narasimhan and S. K. Nayar, "Chromatic framework for vision in bad weather," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 598–605, Hilton Head Island, SC, USA, June 2000.
[32] Y. Pang, M. Sun, X. Jiang, and X. Li, "Convolution in convolution for network in network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 5, pp. 1587–1597, 2018.
[33] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1510–1519, Venice, Italy, October 2017.
[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
[35] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multi-scale structural similarity for image quality assessment," in

Proceedings of the Asilomar Conference on Signals, Systems and Computers, pp. 1398–1402, Pacific Grove, CA, USA, November 2003.

[36] K. Jiang, Z. Wang, P. Yi et al., "Multi-scale progressive fusion network for single image deraining," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8346–8355, Seattle, WA, USA, June 2020.

[37] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D Nonlinear Phenomena*, vol. 60, no. 1–4, pp. 259–268, 1992.

[38] D. Berman, T. Treibitz, and S. Avidan, "Air-light estimation using haze-lines," in *Proceedings of the IEEE International Conference on Computer Photography*, pp. 1–9, Stanford, CA, USA, May 2017.

[39] Y. Zhu, G. Tang, X. Zhang, J. Jiang, and Q. Tian, "Haze removal method for natural restoration of images with sky," *Neurocomputing*, vol. 275, no. 31, pp. 499–510, 2017.

[40] Z. Wang and A. C. Bovik, "Mean squared error: love it or leave it?," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, 2009.

[41] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[42] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "FSIM: a feature similarity index for image quality assessment," *IEEE Transactions on Image Processing: A Publication of the IEEE Signal Processing Society*, vol. 20, no. 8, pp. 2378–2386, 2011.

[43] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a "completely blind" image quality analyzer," *IEEE Signal Processing Letters*, vol. 20, no. 3, pp. 209–212, 2013.

[44] P. Adam, G. Sam, and C. Soumith, "Automatic differentiation in PyTorch," in *Proceedings of the Conference and Workshop on Neural Information Processing Systems*, pp. 1–4, Long Beach, CA, USA, December 2017.

[45] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: optimal speed and accuracy of object detection," 2020, https://arxiv.org/abs/2004.10934.

[46] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 6658763, 18 pages
https://doi.org/10.1155/2021/6658763

Research Article

Deep Learning-Enabled Variational Optimization Method for Image Dehazing in Maritime Intelligent Transportation Systems

Xianjun Hu,[1] **Jing Wang**,[2] **Chunlei Zhang**,[1] **and Yishuo Tong**[1]

[1]*College of Electronic Engineering, Naval University of Engineering, Wuhan 430033, China*
[2]*China Aerospace Science and Industry Corporation, Beijing 100048, China*

Correspondence should be addressed to Xianjun Hu; xianjunhoo@163.com

Received 10 November 2020; Revised 30 March 2021; Accepted 18 April 2021; Published 3 May 2021

Academic Editor: Wen Liu

Image dehazing has become a fundamental problem of common concern in computer vision-driven maritime intelligent transportation systems (ITS). The purpose of image dehazing is to reconstruct the latent haze-free image from its observed hazy version. It is well known that the accurate estimation of transmission map plays a vital role in image dehazing. In this work, the coarse transmission map is firstly estimated using a robust fusion-based strategy. A unified optimization framework is then proposed to estimate the refined transmission map and latent sharp image simultaneously. The resulting constrained minimization model is solved using a two-step optimization algorithm. To further enhance dehazing performance, the solutions of subproblems obtained in this optimization algorithm are equivalent to deep learning-based image denoising. Due to the powerful representation ability, the proposed method can accurately and robustly estimate the transmission map and latent sharp image. Numerous experiments on both synthetic and realistic datasets have been performed to compare our method with several state-of-the-art dehazing methods. Dehazing results have demonstrated the proposed method's superior imaging performance in terms of both quantitative and qualitative evaluations. The enhanced imaging quality is beneficial for practical applications in maritime ITS, for example, vessel detection, recognition, and tracking.

1. Introduction

Maritime video surveillance system has always been an indispensable part of maritime supervision. Affected by small droplets in the air, the image captured by the maritime surveillance system's imaging equipment always tends to be of low quality. There are phenomena such as poor visibility, edge degradation, color distortion, and texture distortion. This negative impact will directly affect the implementation of advanced vision tasks, such as ship detection and tracking [1, 2]. With vigorous computer vision development, many enhancement methods for maritime images have been proposed [3, 4]. Since haze is easily generated in the maritime environment and seriously affects the visual effect, it is also necessary to research the dehazing of maritime images. Starting from the correlation of haze images, Tang et al. [5] discovered some features, including Local Max Contrast, Local Max Saturation, and Hue Disparity, different from dark channel prior (DCP). In [5], features that significantly impact the dehazing effect are obtained by comparison and selection. The model is constructed using these combined features and trained using the synthetic dataset to optimize the parameters. However, it easily leads to noise amplification and distortion when dealing with those haze images with relatively high concentration. By proposing a new prior named color attenuation prior, Zhu et al. [6] created a linear model to estimate the scene depth under the new prior. It solved the atmospheric scattering model to obtain potential clear images. The algorithm's defect lies in that the estimation of the scene depth of hazy white images is biased, affecting the dehazing effect. Subsequently, many variations-based image dehazing methods have been proposed [7–10]. Although these methods can perform well in some situations, they cannot robustly process maritime images due to huge texture structure differences.

Based on Tang et al. [5] and Zhu et al. [6], Cai et al. [11] constructed a convolutional neural network (CNN) called DehazeNet for learning the mapping relations between hazy

images and their medium transmission maps. As the first successful usage of deep learning for image dehazing, DehazeNet is trained by building a training dataset with synthesized hazy images. The trained DehazeNet takes a hazy image as input and outputs its medium transmission map to acquire a haze-free result by a simple traditional method. Similar to DehazeNet, Ren et al. [12] developed a multiscale CNN (i.e., MSCNN) to learn the relations between hazy images and their transmission maps and synthesized an indoor dataset with different hazy images based on the NYU Depth dataset [13]. Zhao et al. [14] proposed a deep fully convolutional network for more accurate transmission estimation and developed a new outdoor synthetic training set. Compared with DehazeNet and MSCNN, the dehazing effect in [14] is improved, but it needs broad parameters, and the computational cost that results in the speed of dehaze is slow. Li et al. [15] reformulated the atmospheric scattering model and proposed a light-weight CNN, called AOD-NET. Unlike most previous indirect CNN-based works that first estimate medium transmission maps based on CNN and recover haze-free images by traditional physical methods, AOD-NET can directly dehazed images from their hazy ones. Motivated by image denoising, Du et al. [16] converted a dehazing problem into a denoising one and proposed a deep residue learning network to remove haze from hazy images. Chen et al. [17] proposed a gated context aggregation network for image dehazing and deraining and applied the smoothed dilated convolution to avoid the gridding artifacts. Most CNN-based methods leverage haze-free images to synthesize hazy datasets [17, 18]. However, some researchers thought that it could not represent the data distribution of real hazy images correctly, and some deficiencies existed in those models trained with synthesized datasets.

Recently, generative adversarial net (GAN), proposed by Goodfellow et al. [19], has been proven potent in image-to-image translation. Typically, GAN includes two subnetworks, that is, generator and discriminator, which are adversarially trained at the same time to acquire expected results. Besides, GAN cannot rely on any synthetic hazy image pairs. Yang et al. [20] proposed an end-to-end disentangled dehazing network trained by unpaired hazy images and clean images and generated haze-free images. Zhang et al. [21] proposed a new dehazing architecture, called densely connected pyramid dehazing network (DCPDN), that could jointly learn to estimate transmission map, atmospheric light, and dehazed images. Furthermore, a joint discriminator within DCPDN was designed to optimize dehazed images and transmission maps. Suarez et al. [22] proposed a stacked conditional GAN to dehaze each color channel of RGB image independently and applied multiple loss functions to optimize the network over a conditional probabilistic model. Generally, GAN-based algorithms need extensive data to train, but the training is toughed and requires higher equipment requirements. Although many dehazing methods based on the physical model and deep learning have been proposed, these methods do not consider the characteristics of maritime images. It cannot be well applied to maritime supervision tasks. Therefore, it is necessary to propose a method for enhancing the hazy maritime image.

In this work, our contribution can be described as follows:

We propose a dehazing method based on the atmospheric scattering model. Specifically, we use a fusion strategy to estimate the transmission. Then, a two-step optimization method based on deep learning is designed to optimize the transmission.

The solution to the subproblem obtained by our proposed two-step optimization algorithm is equivalent to image denoising based on deep learning.

Our method has the best performance in synthetic and real dehazing experiments compared with other methods.

The remainder of this paper is organized into the following several sections. Section 2 briefly introduces the problem formulation related to image dehazing. The optimal estimation of the transmission map is presented in Section 3. Section 4 proposes the CNN-enabled variational optimization method and its numerical optimization algorithm. Numerous experiments on synthetic and realistic datasets are performed in Section 5. Finally, we end this paper by summarizing the main contributions in Section 6.

2. Problem Formulation

2.1. DCP. Based on the statistical analysis of massive images, He et al. [23] discovered the dark channel phenomenon and proposed the DCP. DCP believes that some pixel values in the nonsky local region of any clear image are always low in a specific color channel, even approaching zero. Figure 1 shows various maritime images and corresponding dark channel images. It is evident that the dark channel value in most regions tends to zero. In this work, the mathematical expression of DCP can be written as

$$J^{\text{dark}}(x) = \min_{c\in\{r,g,b\}}\left(\min_{y\in\Omega(x)}\left(J^c(y)\right)\right) \longrightarrow 0, \tag{1}$$

where J and J^{dark}, respectively, indicate the outdoor haze-free image and the corresponding dark channel image, J^c represents the single-channel image corresponding to J in the color channel $c \in \{r, g, b\}$, and $\Omega(x)$ is the local region centred on the pixel point x.

2.2. Image Dehazing. To achieve image dehazing, we first describe the formation of hazy images. Narasimhan et al. [24] explained the hazy images' imaging process by establishing the mathematical model named atmospheric scattering model. This model can be exploited to describe the hazy image production process; that is,

$$I(x) = J(x)t(x) + A(1 - t(x)), \tag{2}$$

where I denotes the hazy image, t represents the transmission map, and A is the global atmospheric light value. Assuming that A is known, we can take minimization

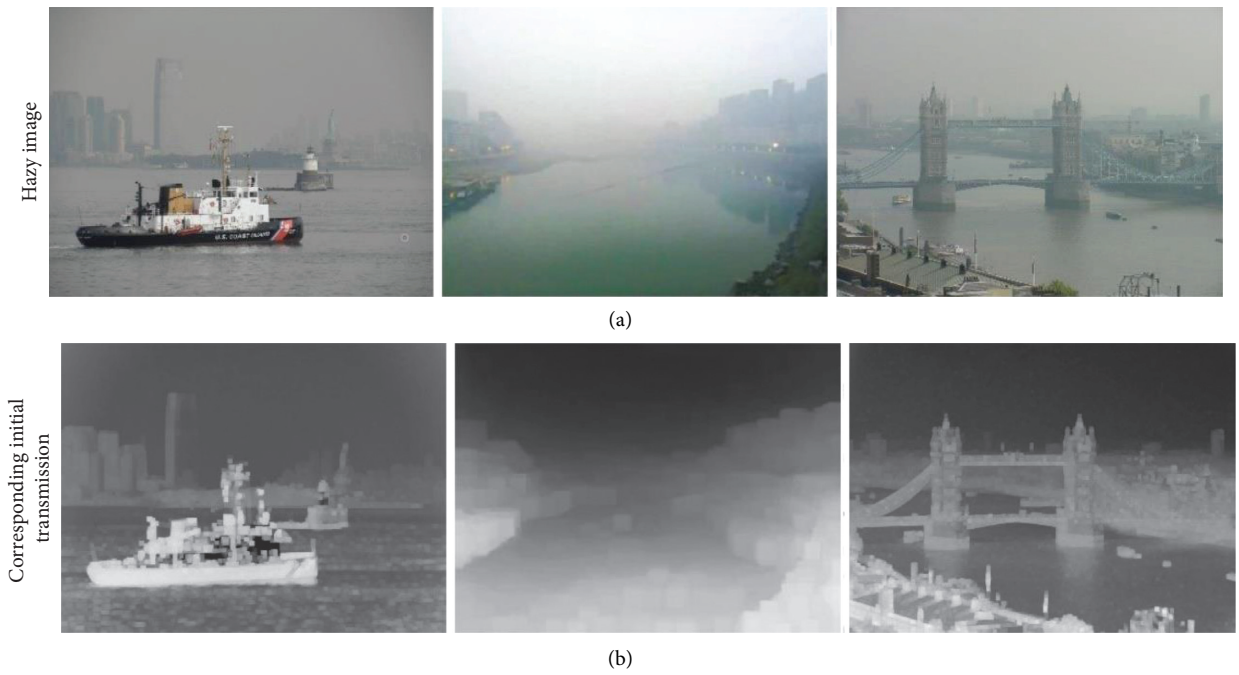


Figure 1: The hazy image on the water with sky region (a) and the corresponding initial transmission (b).

operation on both sides of equation (2) to obtain equation (3), which can be given by

$$\min_c\left(\min_{y\in\Omega(x)}\frac{I^c(y)}{A^c}\right)=\min_c\left(\min_{y\in\Omega(x)}\frac{J^c(y)}{A^c}\right)t(x)-(1-t(x)). \tag{3}$$

According to equation (1), we can approximate $\min_c(\min_{y\in\Omega(x)}J^c(y)/A^c)\approx 0$. Therefore, equation (3) can be rewritten as follows:

$$t(x)=1-\omega\min_c\left(\min_{y\in\Omega(x)}\frac{I^c(y)}{A^c}\right), \tag{4}$$

where ω is an adjustment parameter, indicating the degree of dehazing of the image. The parameter's introduction can preserve a certain haze in the sky region to make the dehazed image more natural. The value of ω is determined by the actual situation. In general, a better result can be obtained with $\omega = 0.95$.

In the atmospheric light value estimation, the traditional approach selects the pixel with the highest brightness in the sky region as the value of A. However, the pixels' inability with the highest brightness to be accurately distributed in the sky regions will result in a failed estimation of atmospheric light. Therefore, He et al. [23] selected the pixels with the 0.1% highest brightness in the dark channel image of the hazy image and used the maximum value of these pixels corresponding to the hazy image as the estimated value of atmospheric light.

It can be seen from equation (3) that the transmission map obtained by DCP has the same value in the local regions, and the transmission changes significantly when the brightness changes suddenly. Therefore, the image restored by this transmission map will have a blocking effect. To obtain a more refined transmission map, He et al. adopt the soft matting algorithm to optimize the initial transmission map. After obtaining the atmospheric light A and transmission map t, the potentially clear image can be restored according to the inverted atmospheric scattering model; that is,

$$J(x)=\frac{I(x)-A}{\max(t(x),t_0)}+A, \tag{5}$$

where t_0 represents the lower boundary of the transmission map. In our experiments, we set $t_0 = 0.1$ generally.

However, the above dehaze method based on transmission map estimation needs to ensure that the DCP theory is valid. When the DCP theory fails, the inaccurate transmission map estimated will lead to the restored image's poor visual effects. Theoretically, in the dark channel image corresponding to the clear maritime image, the sky and water regions' pixel value fails to approximate 0. Therefore, it is unreasonable to directly estimate the transmission map of the hazy maritime image with DCP.

3. Optimal Estimation of Transmission Map

Images captured on the water usually contain large sky regions that generally do not satisfy the DCP. The conventional algorithm introduced in Section 2 is easy to lead to inaccurate transmission estimation. To improve the dehazing effect of the proposed method in haze image on the water with sky region, we also use the soft segmentation method [15] to correct the initial transmission map in this section.

Firstly, we estimate the haze image's initial transmission map on the water based on the DCP. Secondly, we use the soft segmentation method to process the initial transmission map to obtain the transmission weight map. Thirdly, we use the brightness model proposed in [25] to estimate the sky

region's transmission map. Finally, we combine the transmission weight map to merge the transmission map of the nonsky region and the sky region to obtain the corrected initial transmission map. The revised initial transmission map in this section will help to calculate the joint optimization model.

3.1. Weight Function of Transmission Map. As for a hazy image on the water I, we can calculate the initial transmission map based on the DCP mentioned in Section 2. The initial transmission map $\tilde{t}_d(x)$ can be given by

$$\tilde{t}_d(x) = 1 - \omega \min_c \left(\min_{y \in \Omega(x)} \frac{I^c(y)}{A^c} \right), \tag{6}$$

where $\omega = 0.95$ and $\Omega(x)$ represents a 21×21 region centred on x. The atmospheric light A can be estimated by the DCP directly.

As shown in Figure 1, the sky region's values are obviously smaller than the values of other regions in the transmission map estimated by the DCP theory. Therefore, we can roughly distinguish the sky region and other hazy images based on the transmission value in the initial transmission map. The methods proposed in [25] can obtain the possibility that each pixel of a hazy image belongs to the sky region and other regions, that is, transmission weight map, based on the above transmission map. In the transmission weight map, the pixels close to 0 are considered sky region, while pixels close to 1 are considered other regions. To distinguish the sky region and other hazy image regions more accurately, we use a sigmoid function to stretch the transmission. Specifically, the transmission weight function of the initial transmission is obtained as follows:

$$w(x) = \frac{1}{1 + e^{-\theta_1 \left(\tilde{t}_d(x) - \theta_2 \right)}}, \tag{7}$$

where $\tilde{t}_d(x)$ is the initial transmission obtained using the DCP theory, θ_1 is the parameter to adjust the slope of the sigmoid curve, θ_2 is the centre of the horizontal coordinate set according to the $\tilde{t}_d(x)$ range, and θ_1 and θ_2 can be, respectively, given as follows:

$$\theta_1 = \frac{20}{\max\left(\tilde{t}_d(x)\right) - \min\left(\tilde{t}_d(x)\right)}, \tag{8}$$

$$\theta_2 = -10 - \theta_1 \min\left(\tilde{t}_d(x)\right). \tag{9}$$

This section uses the sigmoid function to stretch the initial transmission to get the transmission weight map, shown in Figure 2. The soft segmentation method can easily distinguish the sky region and other regions. Because the water surface condition is complicated, the transmission weight map of some hazy image on the water obtained by the soft segmentation method may have deviations, which will result in insufficient transmission after mergence. Therefore, the transmission after mergence is further optimized to obtain the optimal value.

3.2. Transmission Estimation with Sky Regions. The transmission of the sky region estimated by the DCP theory is not accurate because the white regions (such as the sky) do not conform to the DCP theory, which will result in the failure of dehazing the hazy image on the water with a large sky region. In order to estimate the transmission of the sky region more accurately, we estimate the transmission of the sky region based on the brightness model.

According to the hazy image degradation model described in Section 2, the relationship between the transmission of the scene and the depth information of the object is as follows:

$$t(x) = e^{-\beta(\lambda)d}, \tag{10}$$

where d represents the distance between the object and the imaging device. β is the dissipation coefficient of the medium.

The above formula shows that the scene transmission could be estimated if the depth information of the image can be obtained. In [25], Zhu et al. found that the brightness distribution in HSL color space in the hazy image is usually related to depth information through the statistics of a large number of hazy images, and the brightness of the sky region is much larger than other regions. Therefore, we can simulate the scene depth according to the brightness of the hazy image and then estimate the scene transmission as follows:

$$\tilde{t}_L(x) = e^{-\beta(\lambda)\widehat{L}(x)}, \tag{11}$$

where $\tilde{t}_L(x)$ is the transmission estimated from the brightness model, $\widehat{L}(x)$ is the corrected brightness, and β is the dissipation coefficient of the medium. Different wavelengths of light have different dissipation coefficients under the same medium. According to the Mie Scattering Model, the dissipation coefficients of red, green, and blue light are taken as 0.3324, 0.34333, and 0.3502, respectively.

To better simulate the real scene depth, the brightness is stretched as follows:

$$\widehat{L}(x) = \frac{\tau}{L^*} L(x), \tag{12}$$

where $L(x)$ is the brightness of the hazy image, L^* takes the value at the 95% quantile of brightness, and τ represents the depth range of the real scene. The value of τ can be selected according to the haze density in the hazy image. The greater the haze, the greater the value of τ, and vice versa. In this paper, $\tau = 3.4$.

By combining equations (11) and (12), we can see that the transmission obtained based on the brightness model is

$$\tilde{t}_L(x) = e^{-\beta(\lambda)\frac{\tau}{L^*}L(x)}. \tag{13}$$

3.3. Combination of Transmission Map. The brightness model can simulate the depth of the sky region well but cannot accurately simulate the depth of other regions, which will result in the inaccurate transmission of other regions estimated. On the contrary, the initial transmission estimated based on DCP theory can better estimate the transmission of other regions. Therefore, in this section, we merge the transmissions that are, respectively, estimated

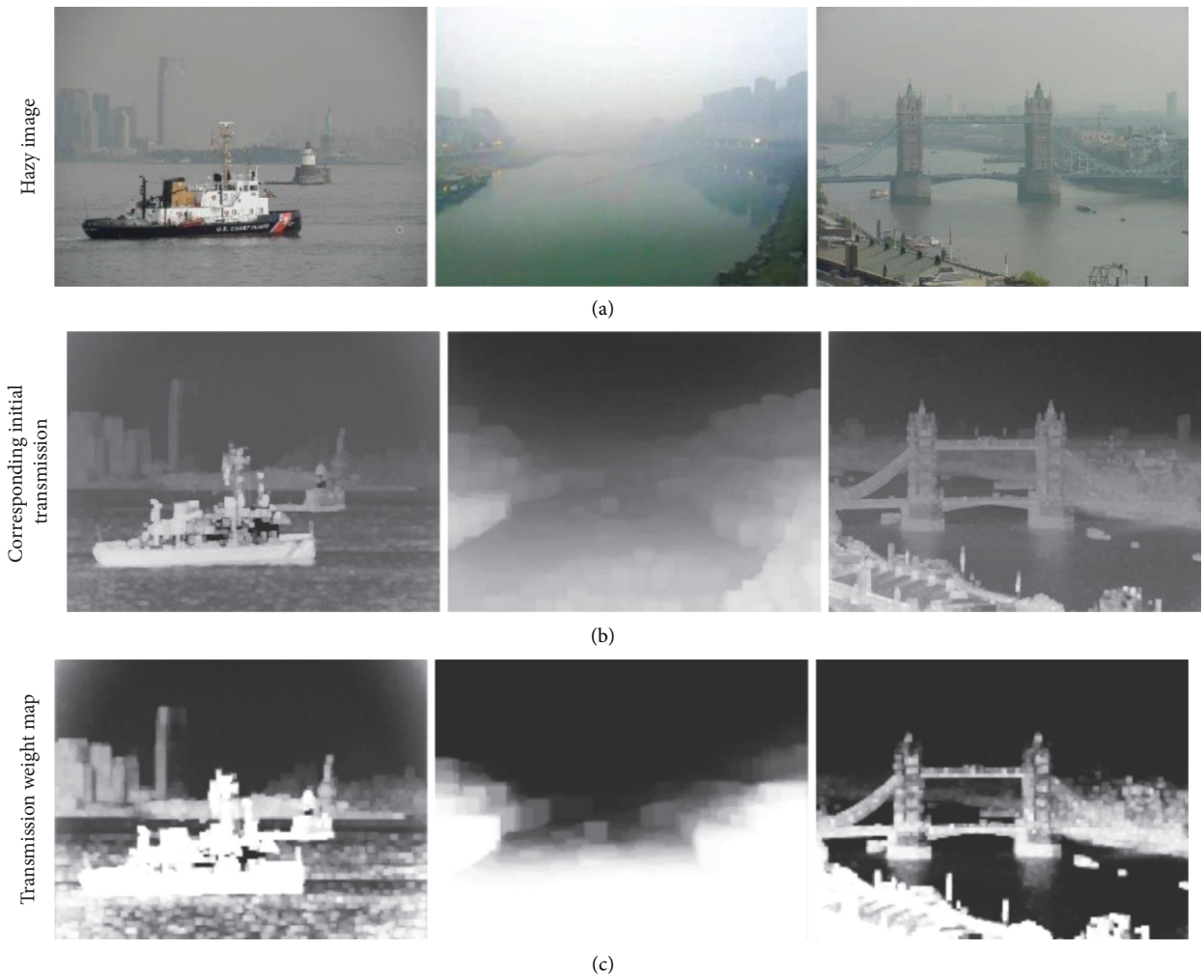


Figure 2: The result of the sky region segmentation. (a)-(c) From top to bottom: the hazy image on the water with a large sky region, the corresponding initial transmission, and the transmission weight map.

using the above methods through the transmission weight map to obtain a more accurate result. The corrected initial transmission t_0 is expressed as follows:

$$t_0 = w(x)\widetilde{t}_d + (1 - w(x))\widetilde{t}_L(x), \tag{14}$$

where $\widetilde{t}_d$ represents the initial transmission of other regions, $\widetilde{t}_L$ represents the initial transmission of the sky region, $w(x)$ represents the weight function of other regions' transmission, and $1 - w(x)$ represents the weight function of the sky region transmission.

4. CNN-Enabled Variational Optimization Method

4.1. The Unified Transmission Estimation and Image Dehazing Framework. In the previous section, the weight map of transmission is obtained by the soft segmentation and fuses the initial transmission by the weight map. Because the soft segmentation method only uses the sigmoid function to stretch the initial transmission to obtain the weight map, transmission obtained by fusion of this weight map is still inaccurate. According to the process of image dehazing, accurate estimation of transmission is a crucial step for image dehazing to obtain satisfactory results. To estimate the transmittance as accurately as possible and restore the potential clear image, we proposed a joint optimization model that simultaneously optimizes the transmittance and the potential clear image within a unified framework. The joint optimization model is given by

$$(\widehat{\mathbf{J}}, \widehat{t}) = \arg\min_{J,t} \mathbf{I} - \mathbf{J}t - \mathbf{A}(1-t)_2^2 + \alpha t - t_{02}^2 + \lambda_1 \varphi(\mathbf{J}) + \lambda_2 \psi(t), \tag{15}$$

where $\mathbf{J}$ is the haze-free image to be restored, $\mathbf{I}$ is the captured hazy image, A is the atmospheric light, t is the transmission of hazy image, and t_0 represents the corrected initial transmission. $\mathbf{I} - \mathbf{J}t - \mathbf{A}(1-t)_2^2$ and $t - t_{02}^2$ are data fidelity terms to constraint haze-free image and its transmission. $\varphi(\mathbf{J})$ and $\psi(t)$ are the regularization terms representing the prior information of J and t, respectively. α, λ_1, and λ_2 are positive parameters.

4.2. Numerical Optimization. Since the haze-free image $\mathbf{J}$ and the transmission t are independent, we tend to propose a two-step optimization method to decompose equation (15) into the following two subtasks.

4.2.1. Estimation of Haze-Free Image $\mathbf{J}$. If t^k is fixed, the optimization solution of haze-free images is as follows:

$$\mathbf{J}^{k+1} = \arg\min_{J}\left(\frac{\mathbf{I}-\mathbf{A}}{t^k}+\mathbf{A}\right)-\mathbf{J}_2^{k2}+\lambda_1\varphi\left(\mathbf{J}^k\right). \tag{16}$$

4.2.2. Estimation of Transmission $\mathbf{t}$. In a similar way, if $\mathbf{J^k}$ is fixed, transmission t optimization subtask is solved by

$$\begin{aligned} t^{k+1} &= \arg\min_{\mathbf{J},t}\ \mathbf{I}-\mathbf{J}t-\mathbf{A}(1-t)_2^2+\alpha t-t_{02}^2+\lambda_1\varphi(\mathbf{J})+\lambda_2\psi(t) \\ &= \arg\min_{t}\ \frac{\mathbf{I}-\mathbf{A}}{\mathbf{J^k}-\mathbf{A}}-t_2^2+\alpha t^k-t_{02}^2+\lambda_2\psi\left(t^k\right) \\ &= \arg\min_{t}(1+\alpha)\frac{\mathbf{I}-\mathbf{A}/\mathbf{J}^k-\mathbf{A}+\boldsymbol{\alpha} t_0}{1+\boldsymbol{\alpha}}-t_2^{k2}+\lambda_2\psi\left(t^k\right). \end{aligned} \tag{17}$$

The above two subtasks need to define appropriate regularization terms $\varphi(\mathbf{J})$ and $\psi(t)$ to be solved. In other words, the solution of the above subtasks is limited by the regularization terms $\varphi(\mathbf{J})$ and $\psi(t)$. Considering that deep learning has a strong prior learning ability, the deep learning method is used in this paper to solve these two subtasks. It can be found that the two subtasks have the same form. According to the Bayesian probability, the above two subtasks can be equivalent to Gaussian denoising tasks [26, 27].

Firstly, equations (16) and (17) are transformed into the following form:

$$\begin{aligned} \mathbf{J}^{\mathbf{k}+1} &= \arg\min_{\mathbf{J}}\left(\frac{\mathbf{I}-\mathbf{A}}{t^k}+\mathbf{A}\right)-\mathbf{J}_2^{\mathbf{k}2}+\lambda_1\varphi\left(\mathbf{J}^{\mathbf{k}}\right) \\ &= \arg\min_{\mathbf{J}}\ \frac{1}{2\left(\sqrt{\lambda_1/2}\right)^2}\left(\frac{\mathbf{I}-\mathbf{A}}{t^k}+\mathbf{A}\right)-\mathbf{J}_2^{\mathbf{k}2}+\varphi\left(\mathbf{J}^{\mathbf{k}}\right), \end{aligned} \tag{18}$$

$$\begin{aligned} t^{k+1} &= \arg\min_{t}(1+\alpha)\frac{\mathbf{I}-\mathbf{A}/\mathbf{J}^k-\mathbf{A}+\boldsymbol{\alpha} t_0}{1+\boldsymbol{\alpha}}-t_2^{k2}+\lambda_2\psi\left(t^k\right) \\ &= \arg\min_{t}\ \frac{1}{2\left(\sqrt{\lambda_2/2(1+\alpha)}\right)^2}\frac{\mathbf{I}-\mathbf{A}/\mathbf{J}^k-\mathbf{A}+\boldsymbol{\alpha} t_0}{1+\boldsymbol{\alpha}}-t_2^{k2}+\psi\left(t^k\right). \end{aligned} \tag{19}$$

According to the research in [22], the images $\mathbf{I}-\mathbf{A}/t^k+\mathbf{A}$ in equation (13) and $\mathbf{I}-\mathbf{A}/\mathbf{J}^k-\mathbf{A}+\alpha t_0/1+\alpha$ in equation (14) can both be considered as the denoising tasks whose noise levels are $\sqrt{\lambda_1/2}$ and $\sqrt{\lambda_2/2(1+\alpha)}$, respectively. Therefore, any image denoiser can be solved by equations (18) and (19).

$$\mathbf{J}^{\mathbf{k}+1} = \mathrm{Denoiser}_1\left(\left(\frac{\mathbf{I}-\mathbf{A}}{t^k}+\mathbf{A}\right),\sqrt{\frac{\lambda_1}{2}}\right), \tag{20}$$

$$t^{k+1} = \mathrm{Denoiser}_2\left(\frac{\mathbf{I}-\mathbf{A}/\mathbf{J}^k-\mathbf{A}+\boldsymbol{\alpha} t_0}{1+\boldsymbol{\alpha}},\sqrt{\frac{\lambda_2}{2(1+\alpha)}}\right), \tag{21}$$

where $\mathrm{Denoiser}_1$ and $\mathrm{Denoiser}_2$ are image denoisers.

4.3. CNN-Based Blind Denoising. In Section 4.2, a dehazing model is proposed to jointly optimize transmission and restore haze-free image, and the joint optimization model is converted into two subtasks. In this section, we mainly introduce a CNN-based blind denoising model to solve the above two subtasks.

4.3.1. Image Denoising Model. The image noise model can be expressed by

$$\mathbf{y} = \mathbf{x} + \mathbf{v}, \tag{22}$$

where $\mathbf{y}$ is observed noisy image, x is noise-free image to be restored, and $\mathbf{v}$ is white Gaussian noise (AWGN) with standard deviation σ.

Because image denoising is an ill-posed inverse problem, previous works generally adopt the prior knowledge or regularization terms to constrain variables to solve the problem. In the Bayesian framework, equation (22) can be

solved by solving the following maximum posterior problem:

$$\widehat{x} = \mathbf{arg}\max_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x}) + \log\, p(\mathbf{x}), \tag{23}$$

where $\log\, p(\mathbf{y}|\mathbf{x})$ is log-likelihood of observed noise image and $\log\, p(\mathbf{x})$ is prior information of $\mathbf{x}$.

According to equation (23), recovering a high-quality noise-free image from a noise image can be regarded as a problem of minimizing the energy function as follows:

$$\widehat{x} = \mathbf{arg}\min_{\mathbf{x}} \frac{1}{2}\mathbf{y} - \mathbf{x}^2 + \lambda\Phi(\mathbf{x}), \tag{24}$$

where $1/2\mathbf{y} - \mathbf{x}^2$ is data fidelity term, $\Phi(\mathbf{x})$ is regularization item related to image prior information, and λ is positive parameter.

4.3.2. CBDNet Structure. After decades of research, many image denoising methods have been proposed. These methods can be divided into two categories: model optimization-based methods and supervised learning-based methods. Model optimization-based methods mainly include total variation, Gaussian mixture model, and BM3D. Most of these algorithms are computationally complex and time-consuming. Supervised learning-based methods mainly contain denoising algorithms based on deep learning such as DnCNN and FFDNet. This denoising method has the advantages of fast speed, excellent performance, and strong robustness [28]. Because deep learning has strong prior learning capabilities and deep learning-based denoising methods have better denoising performance, we adopt the convolutional blind denoising network (CBDNet) proposed by Guo et al. [29] to solve equations (20) and (21). CBDNet benefits from its two subnetworks and provides a blind denoising solution for image denoising. Therefore, we use CBDNet as an optimization algorithm to eliminate haze of different concentrations. As shown in Figure 3, the CBDNet consists of two parts: the noise estimation subnetwork ($\mathrm{CNN_E}$) and the nonblind denoising subnetwork ($\mathrm{CNN_D}$). The noise estimation subnetwork firstly estimates the corresponding noise level image according to the noise image. The nonblind denoising subnetwork obtains the final denoising image according to the noise image and the estimated noise level image. The noise estimation subnetwork includes five fully convolutional layers. Each convolutional layer comprises 32 convolution kernels with size 3×3, followed by the ReLU nonlinear activation function. The nonblind denoising subnetwork is a 16-layer U-Net structure, and the input layer and output layer of the network are connected by skip connection to obtain final denoised images. Equation (20) is equivalent to denoising color images, while equation (21) is equivalent to grayscale images. Therefore, two CBDNet models denoted as $F_{\Theta_1}(\cdot)$ and $F_{\Theta_2}(\cdot)$ need to be trained with the color dataset and the grayscale dataset, respectively. Θ_1 and Θ_2 are the parameters of the networks, respectively. The two CBDNet models share the same structure, except that their input and output channel are different.

4.4. Restoration of Latent Haze-Free Images. In summary, the process of the joint optimization dehazing model is shown in Algorithm 1.

Although the haze image can be restored directly according to the proposed model, experiments show that the haze-free image restored with the optimized transmission has better visual effects in structure and texture. In this work, the coarse transmission map is firstly estimated using a robust fusion-based strategy. A unified optimization framework is then proposed to simultaneously estimate the refined transmission map and latent sharp image. The resulting constrained minimization model is solved using a two-step optimization algorithm. To further enhance dehazing performance, the solutions of subproblems obtained in this optimization algorithm are equivalent to deep learning-based image denoising. Therefore, according to the model, we restore the haze-free image after obtaining the optimized transmittance. In summary, the flow of this algorithm is shown in Figure 4.

5. Experimental Results and Analysis

5.1. Experimental Datasets and Settings

5.1.1. Training Data. As mentioned above, we need to train two CBDNet denoising networks (i.e., F_{Θ_1} and F_{Θ_2}) to remove the unwanted noise in the original image and transmission map, respectively.

Because the denoising network F_{Θ_1} mainly focuses on denoising color images, we choose the SeaShips dataset to build a training set for F_{Θ_1}. We randomly select 500 clear images that are high-quality and noise-free from the SeaShips dataset and synthesize noisy images with different noise levels according to equation (17). F_{Θ_2} aims to optimize the transmittance map, so the depth map in the NYU Depth dataset is used to make the corresponding training set. Similarly, we firstly select 500 depth maps from the NYU Depth dataset, use equation (17) to transform them into corresponding transmission maps, and then synthesize transmission maps with different noise levels according to equation (17). Then, these synthesized images and their source images are cropped into many image blocks whose sizes are 128×128.

5.1.2. Experimental Settings. We use the Adam Optimizer to optimize the network weight parameters in the network training stage. The batch size is set to 64, and the number of iterations (epoch) is set to 40. For the first 20 epochs, we set the learning rate to 10^{-3}, for the last 20 epochs, we set the learning rate to 10^{-4}. The network weight parameters are all initialized with a Gaussian distribution with a mean value of 0 and a variance of 0.01. The loss functions of the two denoising networks are shown in equations (20) and (21). The parameter settings in the

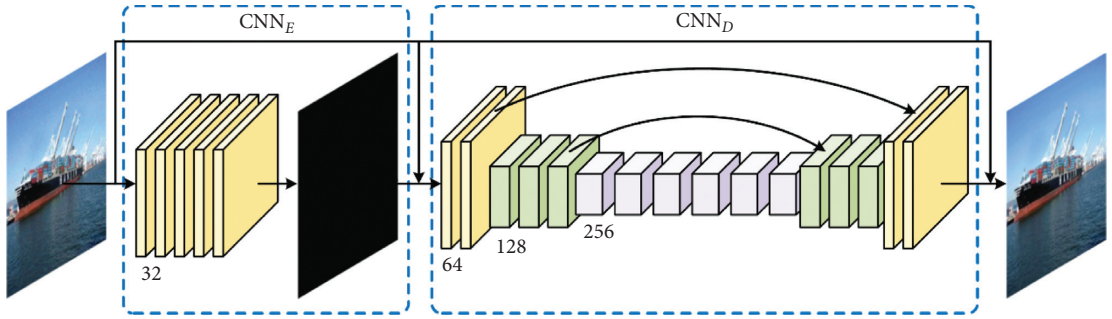


Figure 3: CBDNet structure.

(1) Input: haze image I, atmosphere light A, maximum number of iterations maxiter, adjustment parameter α.
(2) Init: $t^0 = t_0$, $\mathbf{J}^0 = \mathbf{I} - \mathbf{A}/t_0 + \mathbf{A}$.
(3) While $k <$ maxiter:
Assuming that t^k is a constant, $\mathbf{J}^{k+1} = F_{\Theta_1}(t^k)$,
Assuming that $\mathbf{J}^{k+1}$ is a constant, $t^{k+1} = F_{\Theta_2}(\mathbf{J}^k)$.
Output: optimized transmission $t^* = t^{k+1}$, haze-free image $\mathbf{J}^* = \mathbf{J}^{k+1}$

Algorithm 1: Deep learning-enabled variational optimization method.

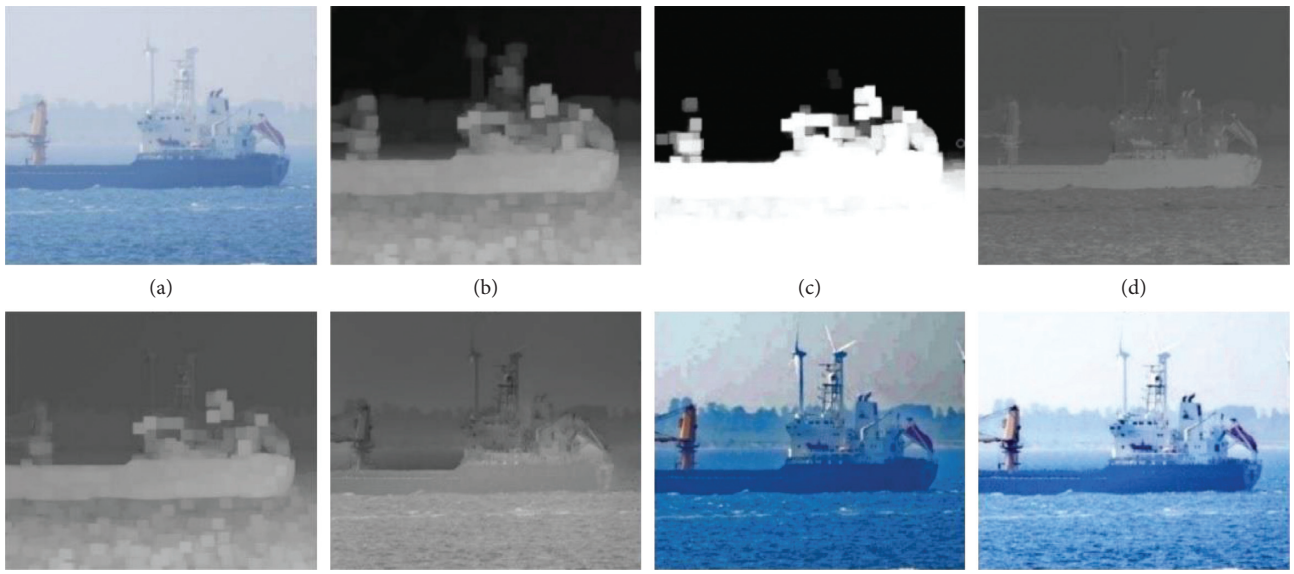


Figure 4: Our dehazing framework. From top-left to bottom-right: (a) input hazy image, (b) DCP-based transmission map $\tilde{t}_d$, (c) weight map of transmission, (d) luminance-based transmission map $\tilde{t}_L(x)$, (e) initial transmission $\tilde{t}_0$, (f) optimized transmission t^*, (g) dehazing result with $\tilde{t}_d$, and (h) dehazing result with t^*.

algorithm are as follows: the maximum number of iterations of the joint optimization model *maxiter* = 6, and the adjustment parameter mathrm $\alpha = 0.5$. The experiment is conducted in Python 3.7 environment with PyTorch package and an Ubuntu 18.04 system, Intel(R) Core(TM) i9-9900X processor, and NVIDIA GeForce RTX 2080Ti GPU. The training of a single model can be done in about one day.

The loss function of the denoising network F_{Θ_1} is as follows:

$$\text{loss} = \frac{1}{q}\sum_{i=1}^{q}\left\|F_{\Theta_1}(J_i) - J_i^*\right\|^2, \tag{25}$$

where $F_{\Theta_1}(J_i)$ is the potentially noise-free clear image, J_i is the input noisy image, J_i^* is the corresponding original noise-free image, and q is the number of training images.

Figure 5: Twelve different clear test images.

The loss function of the denoising network F_{Θ_2} is as follows:

$$\text{loss} = \frac{1}{q}\sum_{i=1}^{q}\left\|F_{\Theta_2}(t_i) - t_i^*\right\|^2, \tag{26}$$

where $F_{\Theta_2}(J_i)$ is output transmission map of F_{Θ_2}, t_i is the input transmission map, and t_i^* is the noise-free transmission map of t_i.

5.2. Experiments on Synthetic Datasets. In this section, comparison and analysis are made between the proposed method and three classic image dehazing methods on synthetic images. To verify the effect of the proposed algorithm on removing different haze concentrations, we have selected three transmissions (i.e., $t = 0.1, 0.3, 0.5$), and three atmospheric light values (i.e., $A = 0.7, 0.8, 0.9$). A total of 9 kinds of different degrees of haze are synthesized on twelve test images shown in Figure 5 to validate the performance of the proposed method. As shown in Tables 1 and 2, we have calculated the PSNR and SSIM objective evaluation indicators of our proposed method and the other three methods. It can easily be seen that our method has the best performance in most cases, the hazy concentration is more extensive, and there are more obvious indicator differences.

As shown in Figures 6–8, we have conducted subjective visual analysis experiments. It is obvious that the sky area of the DCP [23] recovery results has different degrees of color distortion and artificial vignetting. The restoration results in RIVD [30] and MSCNN [12] still have a certain degree of haze, and some color distortions appeared in the sky and water. In contrast, the overall effect of the restoration results of the algorithm in this paper is the best; while maintaining the best visual effect, it virtually eliminates the haze.

5.3. Experiments on Realistic Datasets. In this section, we choose some real hazy images to verify the superiority of our proposed method. Figure 9 shows the dehazing results of different dehazing methods on three maritime video surveillance images. It can be seen from the comparative experimental results in Figure 9 that DCP dehazing algorithm can effectively remove haze. However, at the same time, it also causes artifacts, blocking effects, and color distortion in the restoration results. RIVD and MSCNN dehazing algorithms have individual dehazing capabilities and can avoid artifacts, blocking effects, and so on. However, haze still exists in the restoration results. In contrast, the algorithm in this paper can effectively eliminate the haze, and the restoration results have richer colors, details, and other pieces of information.

The comparison experiment results in other on-water scenes containing hazy images in Figure 10 further prove the effectiveness of the proposed method. It can be seen from the experimental results in Figure 10 that the restoration results of DCP have noticeable artifacts, blocking effects, and color distortion. What is more, the restoration of the sky area is lacking. There is still an apparent haze in RIVD and MSCNN restoration results, resulting in unclear details of some objects. Evidently, the proposed method has an excellent dehazing effect on both sky and nonsky areas. The detailed information of the restoration result is more affluent, and the color is more natural.

It can be seen from the above visual experiments that the proposed method can better recover potentially clear images from hazy images in different water scenes. The restored

Table 1: SSIM comparisons (mean ± std) of various dehazed methods on all test images shown in Figure 5.

SSIM	$A = 0.7$			$A = 0.8$			$A = 0.9$		
	$t = 0.1$	0.3	0.5	0.1	0.3	0.5	0.1	0.3	0.5
Haze	0.593 ± 0.106	0.580 ± 0.109	0.563 ± 0.111	0.746 ± 0.067	0.731 ± 0.073	0.713 ± 0.078	0.866 ± 0.038	0.853 ± 0.044	0.838 ± 0.049
DCP	0.680 ± 0.080	0.690 ± 0.079	0.646 ± 0.111	0.840 ± 0.052	0.814 ± 0.063	0.751 ± 0.081	0.897 ± 0.035	0.869 ± 0.042	0.829 ± 0.051
RIVD	0.637 ± 0.101	0.625 ± 0.105	0.607 ± 0.108	0.819 ± 0.055	0.839 ± 0.060	0.843 ± 0.065	0.839 ± 0.109	0.884 ± 0.022	0.901 ± 0.021
MSCNN	0.704 ± 0.079	0.690 ± 0.085	0.669 ± 0.090	**0.913 ± 0.033**	**0.904 ± 0.042**	0.887 ± 0.051	**0.968 ± 0.018**	**0.967 ± 0.025**	**0.960 ± 0.032**
Ours	**0.796 ± 0.077**	**0.793 ± 0.083**	**0.779 ± 0.086**	0.883 ± 0.054	0.898 ± 0.052	**0.900 ± 0.054**	0.884 ± 0.044	0.899 ± 0.042	0.908 ± 0.041

Table 2: SSIM comparisons (mean ± std) of various dehazed methods on all test images shown in Figure 5.

PSNR	$A = 0.7$			$A = 0.8$			$A = 0.9$		
	$t = 0.1$	0.3	0.5	0.1	0.3	0.5	0.1	0.3	0.5
Haze	12.22 ± 2.08	10.59 ± 2.29	8.77 ± 2.08	14.40 ± 2.08	12.78 ± 2.29	10.96 ± 2.08	17.30 ± 2.09	15.72 ± 2.29	13.87 ± 2.08
DCP	11.92 ± 2.25	9.86 ± 2.30	7.53 ± 1.83	13.39 ± 2.42	10.94 ± 2.11	8.87 ± 1.79	15.23 ± 2.11	12.90 ± 1.80	10.96 ± 1.59
RIVD	14.82 ± 1.84	13.32 ± 2.52	11.09 ± 2.36	17.44 ± 2.25	19.92 ± 2.04	21.94 ± 3.86	16.08 ± 4.02	17.64 ± 3.03	19.10 ± 2.53
MSCNN	13.91 ± 1.97	12.26 ± 2.44	10.17 ± 2.23	18.70 ± 1.55	18.08 ± 2.87	15.68 ± 2.90	**21.28 ± 1.76**	**22.70 ± 2.15**	22.20 ± 3.91
Ours	**17.71 ± 2.02**	**16.93 ± 3.33**	**14.10 ± 3.20**	**19.63 ± 2.13**	**22.77 ± 2.79**	**24.27 ± 5.16**	18.86 ± 2.91	20.56 ± 2.30	**22.29 ± 2.07**

(a) (b) (c) (d) (e)

Figure 6: Continued.

(f)

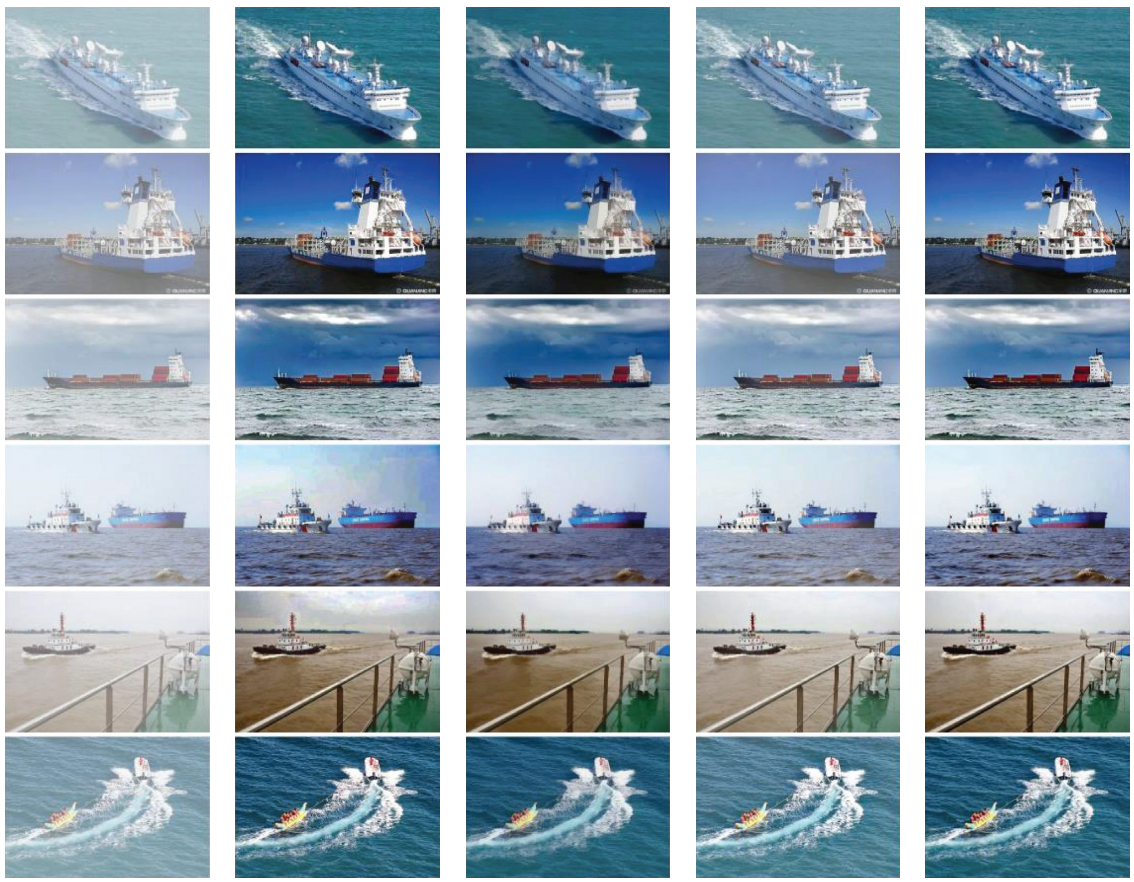
Figure 6: Visual comparisons of synthetic experiments on six images with transmittance value = 0.1 and atmospheric light value = 0.7. From left to right: (a) synthetic haze image and dehazed images generated by (b) DCP [23], (c) RIVD [30], (d) MSCNN [12], (e) ours, and (f) original clear images, respectively.

(a) (b) (c) (d) (e)

Figure 7: Continued.

(f)

Figure 7: Visual comparisons of synthetic experiments on six images with transmission value = 0.3 and atmospheric light value = 0.8. From left to right: (a) synthetic haze image and dehazed images generated by (b) DCP [23], (c) RIVD [30], (d) MSCNN [12], (e) ours, and (f) original clear images, respectively.



(a) (b) (c) (d) (e)

Figure 8: Continued.

(f)

Figure 8: Visual comparisons of synthetic experiments on six images with transmission value = 0.5 and atmospheric light value = 0.9. From left to right: (a) synthetic haze image and dehazed images generated by (b) DCP [23], (c) RIVD [30], (d) MSCNN [12], (e) ours, and (f) original clear images, respectively.

(a) (b) (c) (d) (e)

Figure 9: Visual comparisons of realistic experiments on three haze images. From left to right: (a) real haze image and dehazed images generated by (b) DCP [23], (c) RIVD [30], (d) MSCNN [12], and (e) ours, respectively.

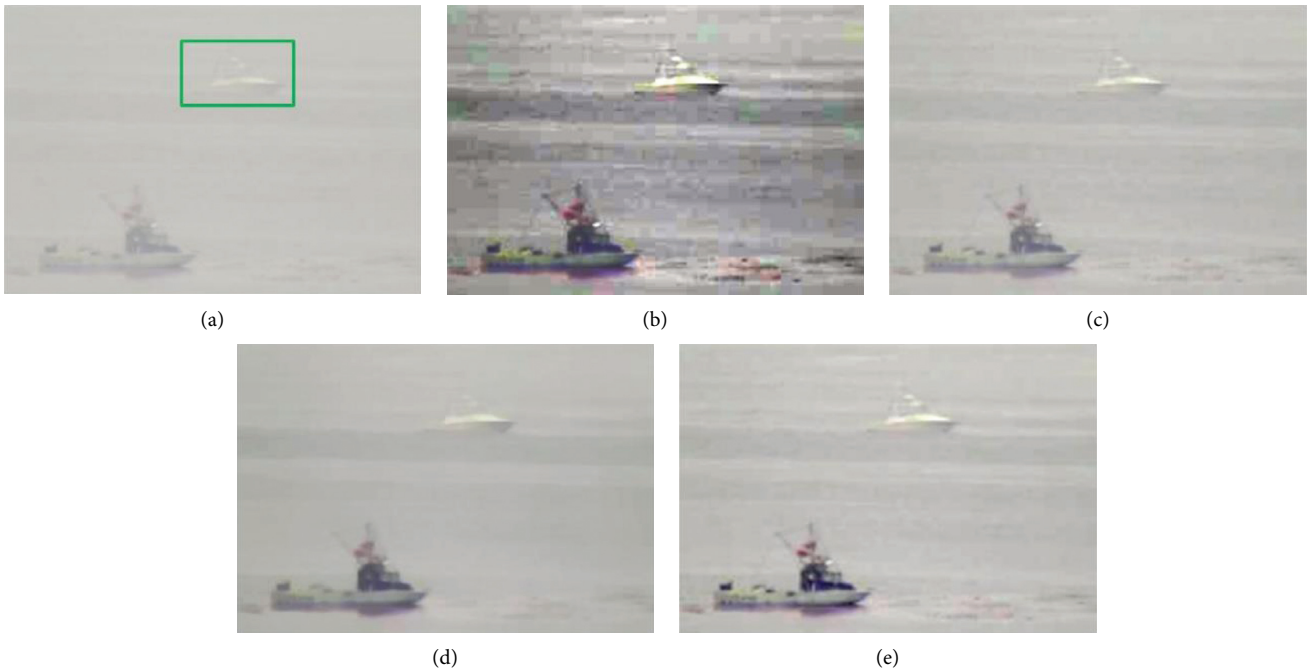


Figure 10: Visual comparisons of realistic dehazing experiments. The proposed method is able to significantly improve the image quality and preserve important structures. (a) Haze, (b) DCP, (c) RIVD, (d) MSCNN, (e) Ours.

images have richer detailed information, which shows the effectiveness and stability of the proposed method.

6. Conclusion

Image dehazing is an important preprocessing problem, which has great practical value in various applications in maritime ITS. In this work, a deep learning-enabled variational optimization method is proposed to reconstruct the latent haze-free image from the observed hazy version. Compared to several competing dehazing techniques, the proposed method is capable of generating superior image restoration results in terms of visual image quality and qualitative evaluation. The main benefit of our method is that it takes full advantage of the unified dehazing framework and the strong representation ability of deep learning. In practical applications, the effectiveness and robustness of vessel detection, recognition, and tracking could be significantly enhanced with the enhanced image quality.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was supported by the National Natural Science Foundation of China under Grant no. 41774021.

References

[1] X. Chen, X. Xu, Y. Yang, H. Wu, J. Tang, and J. Zhao, "Augmented ship tracking under occlusion conditions from maritime surveillance videos," *IEEE Access*, vol. 8, pp. 42884–42897, 2020.

[2] X. Chen, S. Wang, C. Shi, H. Wu, J. Zhao, and J. Fu, "Robust ship tracking via multi-view learning and sparse representation," *Journal of Navigation*, vol. 72, no. 1, pp. 176–192, 2019.

[3] Y. Guo, Y. Lu, R. W. Liu, M. Yang, and K. T. Chui, "Low-light image enhancement with regularized illumination optimization and deep noise suppression," *IEEE Access*, vol. 8, pp. 145297–145315, 2020.

[4] M. Yang, X. Nie, and R. W. Liu, "Coarse-to-fine luminance estimation for low-light image enhancement in maritime video surveillance," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pp. 299–304, Auckland, New Zealand, October 2019.

[5] K. Tang, J. Yang, and J. Wang, "Investigating haze-relevant features in a learning framework for image dehazing," in *in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2995–3000, Columbus, OH, USA, June 2014.

[6] Q. Zhu, J. Mai, and L. Shao, "A fast single image haze removal algorithm using color attenuation prior," *IEEE Transactions on Image Processing : A Publication of the IEEE Signal Processing Society*, vol. 24, no. 11, pp. 3522–3533, 2015.

[7] R. W. Liu, S. Xiong, and H. Wu, "A second-order variational framework for joint depth map estimation and image dehazing," in *Proceedings of IEEE International Conference on*

Acoustics, Speech and Signal Processing, pp. 1433–1437, IEEE, Calgary, Alberta, Canada, April 2018.

[8] Q. Shu, C. Wu, Z. Xiao, and R. W. Liu, "Variational regularized transmission refinement for image dehazing," in *Proceedings of IEEE International Conference on Image Processing*, pp. 2781–2785, IEEE, Taipei, Taiwan, 2019.

[9] Q. Shu, C. Wu, Q. Zhong, and R. W. Liu, "Alternating minimization algorithm for hybrid regularized variational image dehazing," *Optik*, vol. 185, pp. 943–956, 2019.

[10] Q. Shu, C. Wu, R. W. Liu, K. T. Chui, and S. Xiong, "Two-phase transmission map estimation for robust image dehazing," in *Proceedings of the International Conference on Neural Information Processing*, pp. 529–541, Springer, Siem Reap, Cambodia, December 2018.

[11] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao, "Dehazenet: an end-to-end system for single image haze removal," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5187–5198, 2016.

[12] W. Ren, S. Liu, H. Zhang, J. Pan, X. Cao, and M.-H. Yang, "Single image dehazing via multi-scale convolutional neural networks," in *Proceedings of the European Conference on Computer Vision*, pp. 154–169, Springer, Berlin, Germany, October 2016.

[13] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," *Computer Vision-ECCV 2012*, Springer, in *Proceedings of the European Conference on Computer Vision*, pp. 746–760, 2012.

[14] X. Zhao, K. Wang, Y. Li et al., "Deep fully convolutional regression networks for single image haze removal," in *Proceedings of the IEEE Visual Communications and Image Processing*, pp. 1–4, IEEE, St. Petersburg, FL, USA, December 2017.

[15] B. Li, X. Peng, Z. Wang et al., "Aod-net: all-in-one dehazing network," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4770–4778, Venice, Italy, October 2017.

[16] Y. Du and X. Li, "Recursive deep residual learning for single image dehazing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 730–737, IEEE, Salt Lake City, Utah, USA, June 2018.

[17] D. Chen, M. He, Q. Fan et al., "Gated context aggregation network for image dehazing and deraining," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pp. 1375–1383, IEEE, Waikoloa Village, HI, USA, January 2019.

[18] S. D. Das and S. Dutta, "Fast deep multi-patch hierarchical network for nonhomogeneous image dehazing," in *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops*, pp. 482-483, IEEE, Seattle, WA, USA, June 2020.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., *Generative Adversarial Nets. Advances in Neural Information Processing Systems*, pp. 2672–2680, Montréal, Canada, 2014.

[20] X. Yang, Z. Xu, and J. Luo, "Towards perceptual image dehazing by physics-based disentanglement and adversarial training," in *Proceedings of the Thirty-second AAAI Conference on Artificial Intelligence*, pp. 7485–7492, New Orleans, Louisiana, USA, February 2018.

[21] H. Zhang and V. M. Patel, "Densely connected pyramid dehazing network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3194–3203, IEEE, Salt Lake City, UT, USA, June 2018.

[22] P. L. Suarez, A. D. Sappa, B. X. Vintimilla et al., "Deep learning based single image dehazing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1169–1176, IEEE, Salt Lake City, UT, USA, June 2018.

[23] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 12, pp. 2341–2353, 2010.

[24] S. G. Narasimhan and S. K. Nayar, "Chromatic framework for vision in bad weather," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 598–605, IEEE, Hilton Head Island, SC, USA, June 2000.

[25] Y. Zhu, G. Tang, X. Zhang, J. Jiang, and Q. Tian, "Haze removal method for natural restoration of images with sky," *Neurocomputing*, vol. 275, pp. 499–510, 2018.

[26] K. Zhang, W. Zuo, S. Gu et al., "Learning deep CNN denoiser prior for image restoration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3929–3938, IEEE, Honolulu, HI, USA, July 2017.

[27] W. Lu, J. Duan, Z. Qiu, Z. Pan, R. W. Liu, and L. Bai, "Implementation of high-order variational models made easy for image processing," *Mathematical Methods in the Applied Sciences*, vol. 39, no. 14, pp. 4208–4233, 2016.

[28] B. Goyal, A. Dogra, S. Agrawal, B. S. Sohi, and A. Sharma, "Image denoising review: from classical to state-of-the-art approaches," *Information Fusion*, vol. 55, pp. 220–244, 2020.

[29] S. Guo, Z. Yan, K. Zhang et al., "Toward convolutional blind denoising of real photographs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1712–1722, IEEE, Long Beach, California, USA, June 2019.

[30] C. Chen, M. N. Do, and J. Wang, "Robust image and video dehazing with visual artifact suppression via gradient residual minimization," in *Proceedings of the European Conference on Computer Vision*, pp. 576–591, Springer, Amsterdam, The Netherlands, October 2016.

Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 5548725, 14 pages
https://doi.org/10.1155/2021/5548725

Research Article

Perceiving Excitation Characteristics from Interactions between Field Road and Vehicle via Vibration Sensing

Yuansheng Cheng,[1,2] Xiaoqin Li,[1,2] Xiaolan Man,[1,2] Feifan Fan,[1,2] and Zhixiong Li[3]

[1]*School of Mechanical Electrification Engineering, Tarim University, Alar 843300, Xinjiang, China*
[2]*Key Laboratory of Modern Agricultural Engineering, Tarim University, Alar 843300, Xinjiang, China*
[3]*Yonsei Frontier Lab, Yonsei University, 50 Yonsei-ro, Seodaemun-gu, Seoul 03722, Republic of Korea*

Correspondence should be addressed to Xiaoqin Li; 827625576@qq.com

Received 15 January 2021; Revised 21 February 2021; Accepted 17 March 2021; Published 5 April 2021

Academic Editor: Wen LIU

When agricultural vehicles operate in the field, the soft road excitation makes it difficult to measure the vehicle vibration. A camera-accelerator system can solve this issue by utilizing computer vision information; however, the relationship between the field road surface and the vehicle vibration response remains an unsolved problem. This study aims to investigate the correlation of the soft road excitation of different long-wave surfaces with the vehicle vibration response. Vibration equation between the vehicle and soft road surface system was established to produce an effective roughness model of the field soft road surface. In order to simulate the vehicle vibration state under different long-wave road surfaces, the soil rectangular pits with 21 kinds of different spans and depths were applied to the road surfaces, and a tractor vibration test system was built for vibration test. The frequency spectrum analysis was performed for the vibration response and the roughness signals of the road surfaces. The results showed that coefficient (R^2) of frequency correlation between the roughness excitation and the original unevenness at the excitation point at the rear end of the rectangular soil pit fell within 0.9641~0.9969. The main frequency band of the vibration response fell within 0~3 Hz, and the phenomenon of quadruple frequency existed. The correlation of roughness excitation with quadruple frequency fell within 0.992165~1. The primary excitation points were located at the rear end of the rectangular soil pit. In addition, it also indicated that when the vehicle was driven without autonomous power, the vehicle vibration frequency mainly depended on the excitation frequency of the field road surface and the frequency at the maximum vehicle vibration intensity was 2 or 3 times of that at the maximum field soft road excitation. These findings may provide a reference for optimal design of vibration reduction and control for agricultural vehicles.

1. Introduction

In recent years, agricultural mechanization in china has been improved to a new higher level, followed by an increase in the application of agricultural vehicles [1]. Vehicle vibration will be caused by soil excitation during its operation, which will cause damage of parts, reduce the service life of vehicles [2], and also cause damage to the driver's body [3]. Meanwhile, it will aggravate soil compaction, affect the growth of crop roots [4], and ultimately reduce the yield [5]. As the road surface will be compressed and deformed during the vehicle is driven in the field, a wheel envelope will be formed on the soft road surface if it is featured with short wavelength. This will increase the relative damping between wheel and ground to form buffer [6], making it difficult to cause vehicle vibration. Therefore, long-wave road surface (such as gully and ridge road surface) in field is an important factor causing vehicle vibration [7]. Thus, it is of practical significance to exploring the correlation between different excitation features of long-wave road surface and vehicle vibration response.

The features of road surface excitation mainly depend on road roughness. The excitation features of highway hard road surface can be evaluated by measuring the shape and size of the road surface [8]. The measurement methods generally include contact measurement method, noncontact

measurement method, and dynamic response measurement method [9]. For example, Bidgoli [10], Kheirati [11], and Putra [12] all measured the road surface with a noncontact measuring device and analyzed excitation features of the road surface, which provided a reference for the measurement of roughness of the hard road surface. Xue [13] and Shi [14] established vibration equation of vehicle-road surface system, analyzed vehicle vibration response signal under pothole road surface by using dynamic response measurement method, and calculated the size of pothole in the road, which provided great help for road maintenance. However, soil is a combination of solid, liquid, and gas phases [15]. When it is under pressure, it will be deformed. So, the shape of field soft road surface cannot be directly used as an excitation source for measurement. In the 1990s, research results of Zheng Research Group of Jilin University showed that interaction between soft road surface and wheels will make soft road surface deformed [16, 17], which makes soft road surface at different roughnesses before, at the moment of, and after compression by tires. Domestic scholars call displacement excitation of soft road surface to wheels in the form of a certain roughness as effective roughness. It is generally believed that effective roughness is the real excitation to vehicle [18–20]. Meanwhile, a set of soft road surface roughness test device was designed for measuring original roughness, effective roughness, and rutting roughness [18]. The empirical formula of effective roughness was obtained by combining the features of soil and wheels [20].

There are many researchers who take the effective roughness of soft soil pavement as the excitation input to explore the vibration characteristics of tractors. For example, Zhao [7] designed a set of vibration test system to measure the nonuniformity of paddy field bottom layer combined with rice transplanter. The results show that the nonuniformity of paddy field bottom layer is grade A pavement. The vibration characteristics of tractor are analyzed, which provide a new way for the design of paddy field agricultural machinery and the research of interaction mechanism between agricultural machinery and soil. Fan [21] selected a small sugarcane harvester combined with a vibration test system for vibration test with the random road surface in the field as the input, analyzed the vibration signals of the sugarcane harvester, and solved the excitation characteristics of the effective roughness of the road surface in the field. It is verified that the effective roughness of the soft road surface in the field has a great influence on the vibration of the harvesting cutter head of the sugarcane harvester. Zhu designed a test system of irregularity to measure the ups and downs of the hard bottom layer under paddy fields [22]. A tire-wet and soft paddy field soil system model is established to analyze the vibration characteristics of tractors. The results show that compared with the dirt road without vegetation coverage, the displacement transmission rate of front and rear wheels, pitch vibration, and vertical vibration of the fuselage all decrease [23]. Cutini [24] selected three tractors to install three-axis acceleration sensors at the seats and carried out vibration test combined with six kinds of field roads. The results show that the X, Y, and Z axes responses of the sensor are all in the low-frequency range, and the longitudinal vibration response is highly correlated with the vertical displacement. It shows that the longitudinal comfort depends to a large extent on the vertical motion of the tractor. The above scholars have all taken random soft road surface as input excitation to explore the characteristics of tractor vibration and provided basis for the design of tractor vibration absorption system and real-time detection of road surface excitation characteristics. However, the tractor has too many degrees of freedom, which would lead to mutual interference in the actual movement process and reduce the accuracy in the vibration test process. At the same time, the frequency domain distribution of random soft road excitation is wide, which makes it difficult to accurately express the vibration characteristics of tractors under specific wavelength excitation.

In this study, a test method of tractor vibration is used to explore the correlation between the excitation characteristics of field soft pavement with different wavelengths and tractor vibration by the abovementioned concept of effective roughness. We firstly use the vibration acceleration sensor to explore the vehicle responses and then use the subjective evaluation to find the correlation. Vibration tests are carried out on a specific pavement by setting up rectangular pit with various sizes in the soil tank laboratory, simulating soft soil pavements with different wavelengths, and a single-degree-of-freedom vibration test system is used to avoid the interference caused by the negative effect of unexpected degrees of freedom of the tractor. An effective roughness model is established based on the vibration equation of the tractor-ground system. The frequency spectrum and relevant statistical analysis of the effective roughness and vibration acceleration signals are carried out to obtain the excitation frequency band of the soft pavement, the vibration frequency band of tractor, and the excitation characteristics of the vibration intensity and effective roughness, in order to reveal the correlation between the ground excitation characteristics and tractor vibration characteristics. When driving on a soft road, the longer and shorter wavelengths can cause the space movement of the vehicle, which makes it hard to determine the effective wavelength range from the vibration sensors. Therefore, the binocular event camera is fixed on the vibration test system to perform vibration test on the constructed field road, combining with the three-dimensional spatial motion signal measured by the visual mileage calculation method to synthesize and analyze the one-dimensional vertical displacement signal of the vehicle. By combining the image and vibration measurements, it is possible to obtain the short-wave and long-wave ranges of the pavement and explain the vehicle-soft road interaction. This paper will focus on the correlation between the excitation of the soft pavement and vehicle vibration. The contribution of this work is to provide a basis for real-time detection of the excitation characteristics and determine the wavelength range of the soft pavement, which will provide a basis for effective vehicle vibration control.

2. Materials and Methods

2.1. Test Materials

2.1.1. Test Instruments and Equipment. The single-degree-of-freedom (SDOF) vibration test system is shown in Figure 1. The tire is 825R16 all-steel radial tire of Mountaintech with an air pressure of 0.5 MPa. The acquisition instrument model is WS-5291, the signal amplifier model is WS-2401, the signal acquisition software is Vib'SYS, and the acceleration sensor model is BZ1124, all of which are produced by Beijing Wavespectrum Science and Technology Co., Ltd. The sensitivity of the acceleration sensor in the Z-axis direction (vertical direction) is 6.234 pc/ms^{-2}, the frequency response range is 0.2~8 kHz, and the total mass m in the vertical direction of the wheel is 103.5 kg.

The test site is the soil trough laboratory of Tarim University (longitude: 81.297248, latitude: 40.544428), and the soil is aridisols [25]. The area of the soil tank is 5×30 m^2, and the soil firmness of the trimmed site is measured. The model of the soil firmness instrument is TJSD-750-II. The test site is evenly divided into 7 columns and 3 rows, with a total of 21 sampling points. Each point is measured three times. The depths of the three measurements are 10 cm, 20 cm, and 30 cm, respectively. The measured data are shown in Table 1.

Set the depth and span of the rectangular pit as variables. The depth is 10 cm, 20 cm, and 30 cm, respectively, which is used to change the amplitude of excitation. The spans are 10 cm, 20 cm, 30 cm, 40 cm, 50 cm, 60 cm, and 70 cm, respectively, which are used to change the excitation wavelength. The width is fixed at 30 cm, and the distance between the two pits is fixed at 60 cm. Ten rectangular pits with the same depth and span are one kind of pavement, which is divided into 21 kinds, as shown in Figure 2.

When a vehicle travels on a rectangular pit road surface, assuming that the rear end of the rectangular pit is the excitation point, the wavelength λ of road surface excitation is defined as the sum of the spacing between the two pits and the span of the rectangular pit. Therefore, the equation [7, 27] for solving the excitation frequency of the actual rectangular pit pavement is

$$f = n \times v, \tag{1}$$

where f is the spatial frequency, which is the reciprocal of the wavelength λ, and v is the vehicle speed.

2.2. Test Methods

2.2.1. Vibration Test Method. In this paper, the tractor is used as the power to drive the vibration test bench. According to ISO5008, too slow speed of agricultural vehicles causes the phenomenon that it is difficult to cause excitation and too fast speed causes the nonlinear phenomenon [26]. Therefore, the vehicle speed is kept at 2 km/h during the test process to ensure that the resonance of the test bench can be caused [24]. The excitation frequency is changed by changing the original pavement wavelength, and the vibration intensity is analyzed by test bench vibration acceleration signal. The sampling frequency is set to 2000 Hz, and the flow diagram is shown in Figure 3.

In order to be able to evaluate the vibration intensity of 21 kinds of pavement test, vibrating test bench intensity of soft soil pavement, the weighted root mean square value statistical calculation of vibration acceleration signals is, respectively, selected. The equation for solving the root mean square value of acceleration a_w is shown in the following equation [24, 27]:

$$a_w = \left[\frac{1}{T}\int_0^T a_w^2(t)\mathrm{d}t\right]^{1/2}, \tag{2}$$

where $a_w(t)$ is the weighted acceleration time history function and T is the duration of the measurement.

2.2.2. Establishment of Effective Roughness Excitation Model. In this paper, the test bench response is taken as the output to reverse the effective roughness of soft soil pavement. It mainly includes three parts: (1) establishing the vibration model of the vehicle-soft pavement system, obtaining the effective roughness model, and finding out the parameters that affect the effective roughness, including the stiffness and damping of the system and the displacement, speed, and acceleration of the wheel in the vertical direction. (2) Obtaining equivalent stiffness and equivalent damping by hammering experiment and modal parameter identification. (3) Carrying out one-time integration and two-time integration on the acceleration signal to obtain the vertical speed and displacement signals of the wheel.

(1) Effective Roughness Model. In the process of test, the wheels are set to be evenly and symmetrically excited by the ground, and the stiffness and damping of the soil itself would also affect the vibration characteristics of the vehicle. Therefore, the stiffness and damping of the vehicle-ground system are equivalently superimposed in the vertical direction [22, 27]. The simplified process of the vibration system is shown in Figure 4.

Figure 4(a) is the actual vibration system model. Figure 4(b) is the simplified vibration model. In the model, $K = k_1 + k_2$, $C = c_1 + c_2$. $q(t)$ is the effective ground roughness input excitation, and the vibration differential equation [13, 27] is established as shown in the following equation:

$$m\ddot{x}(t) + C[\dot{x}(t) - \dot{q}(t)] + K[x(t) - q(t)] = 0. \tag{3}$$

In the equation, $x(t)$ is the vertical vibration displacement of wheels; $\dot{x}(t)$ is the vertical vibration speed of wheels; $\ddot{x}(t)$ is vertical vibration acceleration of wheels; $q(t)$ is the effective ground roughness input displacement.

The vertical displacement can be solved by

$$q(t) = x(t) + \frac{m}{K}\ddot{x}(t) + \frac{C}{K}\dot{x}(t). \tag{4}$$

Here, the mass $m = 103.5$ kg and vehicle acceleration signal is $\ddot{x}(t)$. Measured by vibration test that, in order to be

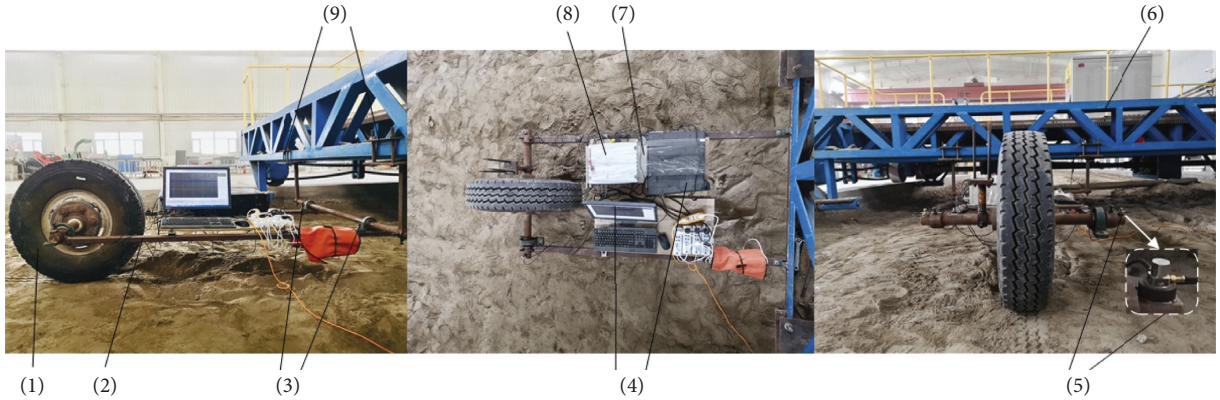
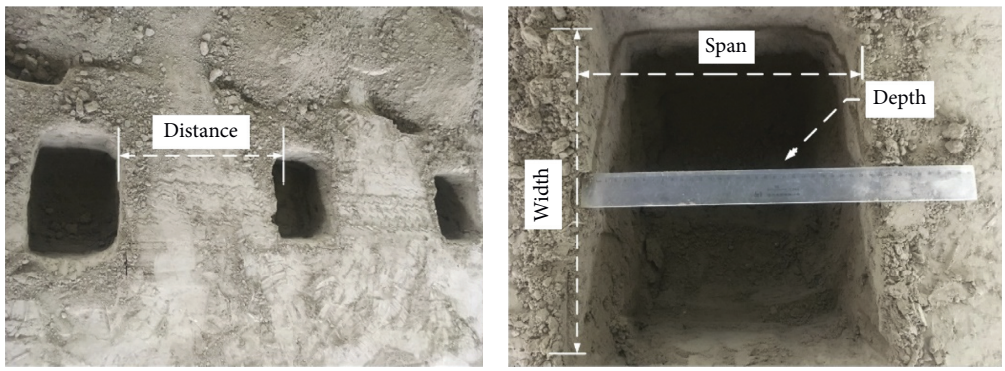


Figure 1: Test system. (1) Wheel assembly, (2) single wheel frame, (3) connection bearing, (4) computer, (5) acceleration sensor, (6) tractor, (7) signal amplifier, (8) data acquisition instrument, and (9) fixing device.

Table 1: Test statistics of solidarity (unit: kg/cm^2).

Depth (cm)	Columns	Rows						
		1	2	3	4	5	6	7
10	1	10.2	13.8	12.6	12.4	13.7	13.8	8.8
	2	15.8	12.6	14.7	13.8	14.8	15.9	16.2
	3	14.6	9.2	10.7	12.5	10.4	11.3	10.6
20	1	16.9	21.9	18.3	16.9	16.9	18.8	16.8
	2	19.0	20.1	21.0	21.1	21.2	22.8	19.3
	3	17.0	20.3	23.1	21.5	19.8	20.0	19.4
30	1	38.2	32.5	30.1	29.7	30.1	36.5	47.0
	2	41.7	50.7	38.8	32.0	23.1	33.9	39.2
	3	37	36	33.1	21.5	29.8	20.9	29.4



(a) (b)

Figure 2: Sample of the rectangular pit.

able to evaluate the road excitation intensity of 21 kinds of pavement test, the power spectral density method of pavement effective roughness is, respectively, selected. The equation for solving the spectral density of effective roughness rate is shown in equation (4). Since the function curve of effective roughness is already known, the power spectral density [5] can be calculated by using a computer, as follows:

$$P = \frac{1}{N} \sum_{k=0}^{N-1} [Q(k)]^2. \quad (5)$$

Here, P is the power spectral density of the effective roughness and $Q(k)$ is the spectral function of the effective roughness time domain signal $q(t)$ after Fourier transform, wherein, $k = 1, 2, 3 \ldots$.

(2) Test of Mechanical Parameters of Vehicle-Ground System. At present, the method of experimental modal analysis is relatively mature. Generally, it includes single-point excitation, multipoint excitation, and single-point partition excitation [28]. In this paper, the hammering method is used

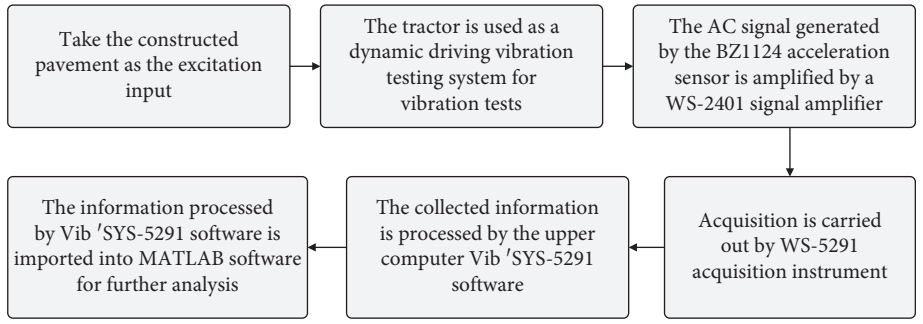


Figure 3: Vibration test flow.

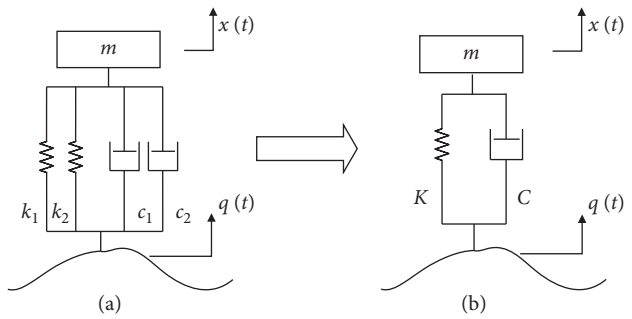


Figure 4: Simplified schematic diagram of the vehicle-ground system model. k_1: soil vertical stiffness; k_2: tire vertical stiffness; c_1: soil vertical damping; c_2: tire vertical damping; $x(t)$: wheel up and down displacement; $q(t)$: effective ground displacement input; K: system vertical equivalent stiffness sum; C: system vertical equivalent damping sum.

for testing, and the signal generated by hammering is taken as the excitation force signal and the signal measured by the acceleration sensor is taken as the response signal. The ratio of Laplace transform of the acceleration response signal to Laplace transform of the excitation signal is taken as system response, i.e., transfer function [29]. The transfer function signal curve is used to identify the parameters and obtain the mechanical parameters of the system. In this paper, a single-point hammer test method is selected. The force sensor model is LC-50 piezoelectric sensor (produced by Beijing Wavespectrum Science and Technology Co., Ltd.). The sampling frequency is 5000 Hz, and the sampling time is 0.2 seconds, as shown in Figure 5.

After modal analysis, the natural frequency ω_n, damping ratio ξ, equivalent stiffness K, and equivalent damping C of the system can be obtained.

The relationship between each parameter is shown in equations 6 and (7) [27, 29]:

$$\omega_n^2 = \frac{K}{m}, \tag{6}$$

$$\xi = \frac{C}{2m\omega_n}. \tag{7}$$

In order to obtain the wheel vertical displacement signal $x(t)$ and the wheel vertical speed signal $\dot{x}(t)$, it is necessary to carry out one-time integration and two-time integration on the measured wheel vertical acceleration signal $\ddot{x}(t)$. For signal integration, there are mainly two methods: time domain integration and frequency domain integration. As time domain integration will cause the generation of DC component and trend term of signal, it will lead to integration error, while frequency domain integration can convert integration operation into division operation, which can better avoid the generation of error [29, 30]. As a result, the frequency domain primary integration of the acceleration signal is performed to obtain the frequency spectrum of the velocity signal as follows:

$$V(K) = \frac{A(K)}{j\omega} \sum_{n=0}^{N} \frac{1}{j2\pi k\Delta f} H(K) a_n e^{-j2\pi kn/N}. \tag{8}$$

The frequency spectrum of the displacement signal obtained by quadratic integration is

$$S(K) = \frac{A(K)}{\omega_k^2} \sum_{n=0}^{N} \frac{1}{(2\pi k\Delta f)^2} H(K) a_n e^{-j2\pi kn/N}. \tag{9}$$

Here,

$$H(K) = \begin{cases} 1, & f_d < k\Delta f < f_u, \\ 0, & \text{otherwise}, \end{cases} \tag{10}$$

where Δf is frequency resolution; f_d is the lower limit of the cutoff frequency; f_u is the upper limit of cutoff frequency; N is the number of sampling points of the data; and ω_k is the frequency corresponding to the Fourier component.

3. Vibration Test Analysis

The acceleration signal in the vertical direction of the wheel is collected through the vibration test system, and the collected voltage signal value is imported into the *m* file written by MATLAB and converted into the acceleration signal value. The vibration response data of 21 kinds of pavements are divided into 7 groups by span, and each group contains pavement data of 3 different depths for comparison and display, as shown in Figure 6. It can be seen that each group of data can see 10 peaks, proving that each rectangular pit constructed can cause vehicle vibration.

MATLAB is used to write a program to calculate the power spectral density of the signal, as shown in Figure 7. It can be seen that the frequency band range in the power

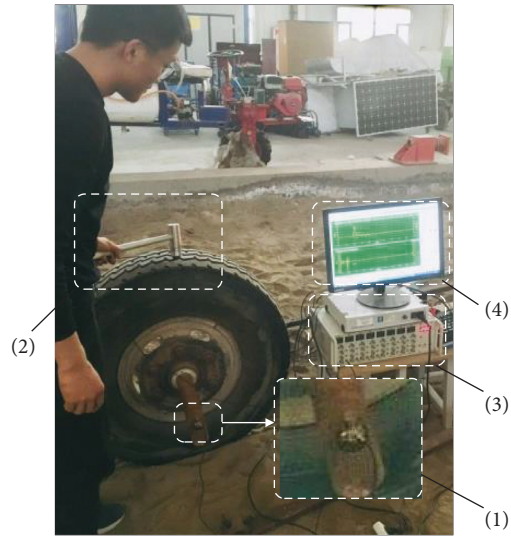


Figure 5: Schematic diagram of hammering test. (1) Acceleration sensor; (2) force hammer sensor; (3) signal amplification and acquisition equipment; (4) computer display.

spectral density is 0~13 Hz, and the most active part of the vibration power value is 0~3 Hz. It is close to the test frequency band of tractor vibration in references [21, 24, 31–33], and the frequency band range does not change significantly with the change of rectangular pit size. This shows that, under the excitation of soft soil pavement, the vehicle vibration is mainly in the low-frequency range, and the SDOF vibration test system can reflect the overall vibration characteristics of the tractor. The vibration response signal is analyzed within 0~3 Hz, and four peak points appear. Taking the data with a span of 10 cm and three different depths as an example, the data rules are consistent.

3.1. Effective Roughness Model Parameters. The transfer function of the vehicle-ground system is obtained through hammering test, as shown in Figure 8. Both the real part and the imaginary part of the transfer function curve have 5 peak points. The first-order natural frequency [28, 29] is obtained by selecting the peak frequency with the real part being 0 and the imaginary part being the highest and carrying out modal parameter identification. A total of 7 hammering tests were carried out (test has the same law for 7 times), and the 7 test data of the first-order natural frequency were averaged. The results are shown in Table 2.

The first-order natural frequency is selected as the dominant frequency of the system, and the ω_n value is 211.8998 Hz. Modal parameters of the first-order natural frequency are identified to obtain damping ratio $\xi = 1.16\%$. Combining equations 6 and (7), the vertical equivalent stiffness K is 4647307.86 N m^{-1} and the equivalent damping coefficient C is 508.8138 N s m^{-1}.

3.2. Vertical Wheel Displacement and Speed Signals. The frequency domain integration program is written by MATLAB, and the vertical vibration acceleration signal of the axle is integrated first and second in the frequency domain. The upper and lower limit cutoff frequencies are 100 Hz and 0.3 Hz [34], respectively, and the vertical speed signal $\dot{x}(t)$ and displacement signal $x(t)$ of the axle are obtained, as shown in Figure 7 (the data are data with a depth of 10 cm and a span of 10 cm). Combining equations 8 and (9), it can be seen that frequency domain integration is equivalent to division operation in frequency domain, so the amplitude would be amplified when the frequency is in the range of 0~1 Hz, and the amplitude would decrease when it is greater than 1 Hz, and the amplitude would increase and decrease as the frequency value increases. As a result, the signal shows a smooth trend in the time domain with the increase of the number of frequency domain integrations, as shown in Figure 9(a). As the number of frequency domain integrations increases in the frequency domain, the amplitude of the signal gradually increases in the lower frequency range and gradually decreases in the higher frequency range, as shown in Figure 9(b).

3.3. Effective Roughness Signal. The vertical speed signal $\dot{x}(t)$ and displacement signal $x(t)$ of the axle, as well as the stiffness K and damping C are obtained, and the time domain signal of the effective roughness is obtained by solving equation (4). Since the effective roughness of soft soil pavement is the superposition of the vertical displacement and velocity of test bench and the product of acceleration and various mechanical parameters, as well as the value of vertical stiffness K is much larger than the value of vertical damping C and vertical mass m, the product of speed and acceleration and various mechanical parameters is smaller, and the sum of tire deformation is smaller, due to relatively small stiffness of soil compared to tire. So, the effective roughness value basically depends on the vertical displacement of the wheel. As shown in Figure 10, the span is 10 cm. The data of rectangular pit pavement with depths of 10 cm, 20 cm, and 30 cm show that the amplitude of effective roughness changes irregularly with the increase of depth. The rest of the data are consistent with the above. Fourier transform is performed on the effective roughness time domain signal to obtain a frequency domain diagram. As shown in Figure 10(b), the excitation frequency band range is from 0 to 3 Hz, and that frequency values corresponding to the two peak points of each group of data are different. The resulting difference has a linear multiple relationship with the frequency corresponding to the first peak point. The vertical displacement of the peak point decreases with the increase of frequency, which is similar to the nonsinusoidal periodic signal [35]. The rest of the data are consistent with the above.

4. Discussion

4.1. Correlation Analysis of Effective Roughness, Original Road Roughness, and Main Frequency Band of Vibration Response. This experiment includes 21 kinds of rectangular pit pavements with 7 different wavelengths. Because the constructed rectangular pit pavement is similar to a nonsinusoidal periodic signal, there will be frequency doubling. Since there are 4 peak

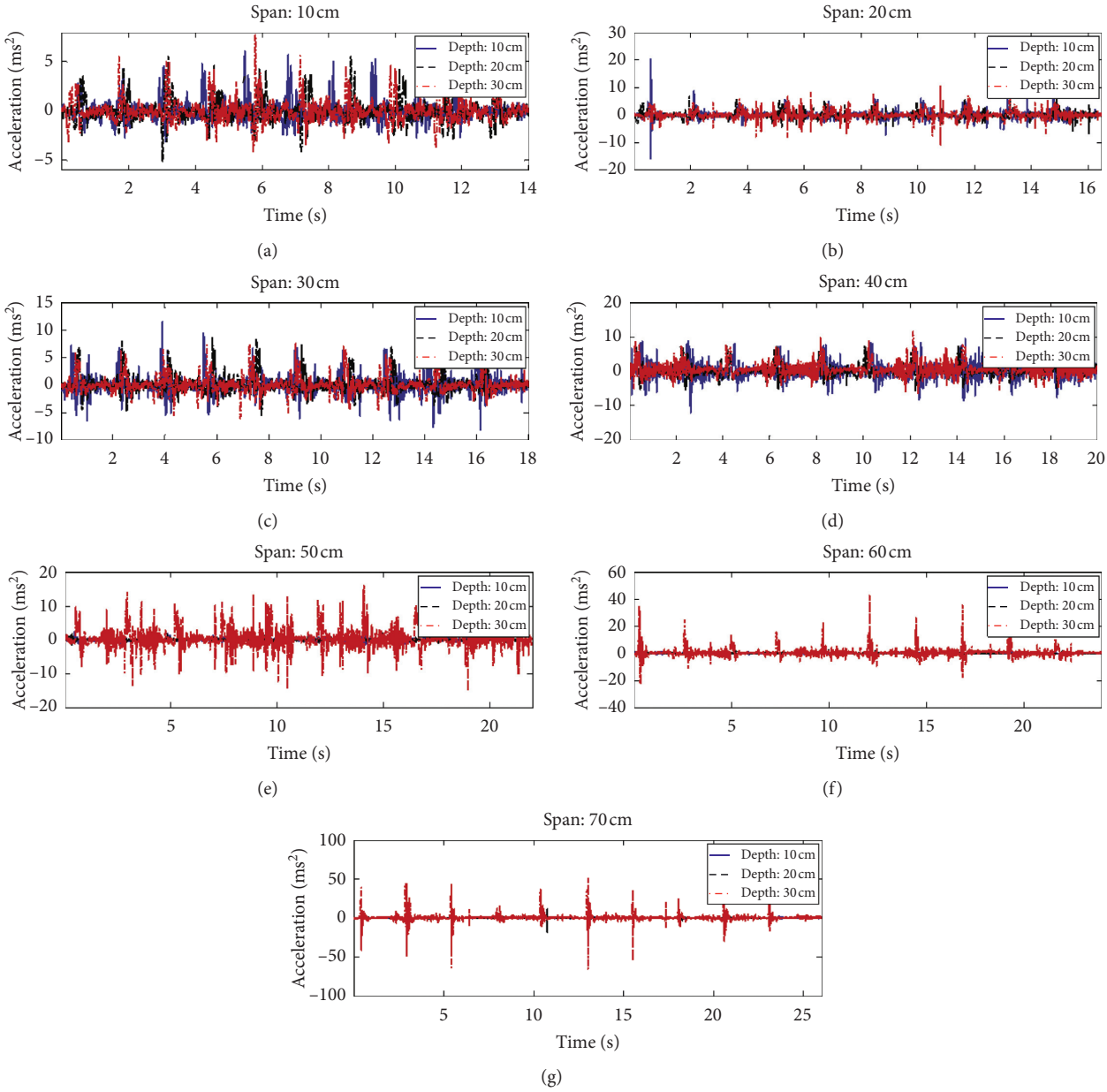


Figure 6: Time-domain diagram of the vibration acceleration signal.

points in the spectrum diagram of the effective roughness signal, the four frequency doubles of the original road roughness excitation signal are counted, and Table 3 is drawn in combination with the spatial frequency n and time frequency f corresponding to different wavelengths, as follows.

Linear regression analysis is carried out on the peak point frequency of the effective roughness model signal, the excitation point frequency of the original pavement roughness, and the four peak point frequencies of the main frequency band of the vibration signal, respectively. The results show that the coefficients of the linear regression equation between the effective roughness model signal and the excitation point frequency of the original pavement roughness are close to 1, and the R^2 value ranges from 0.9641 to 0.9969. The coefficients of the frequency linear regression equation of the four peak points of the main frequency band of the effective roughness model signal and the vibration acceleration signal are all around 1, and the R^2 value range is 0.992165~1, as shown in Table 4. It is shown that the effective irregularity excitation frequency is highly correlated with the original irregularity excitation point frequency of rectangular pit pavement. The feasibility of the effective roughness solution method is verified. It is proved that the excitation point of the vehicle when driving on the soft road surface is the rear end of the rectangular pit. At the same time, it is shown that the main frequency band of vehicle vibration is highly correlated with the excitation signal of the road surface and basically depends on the excitation characteristics of the ground.

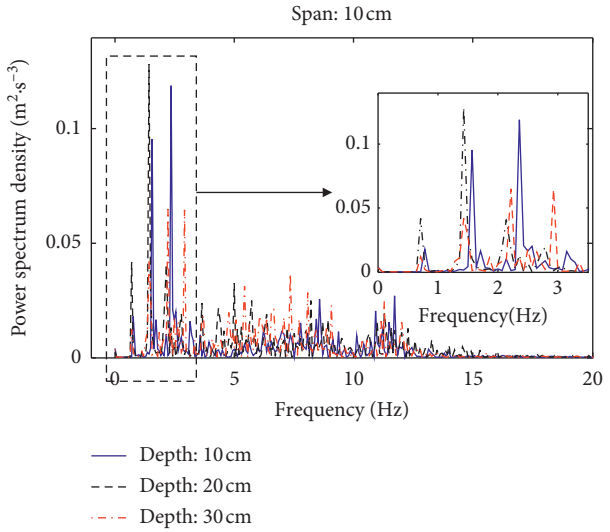


Figure 7: Frequency domain diagram of the vibration acceleration signal.

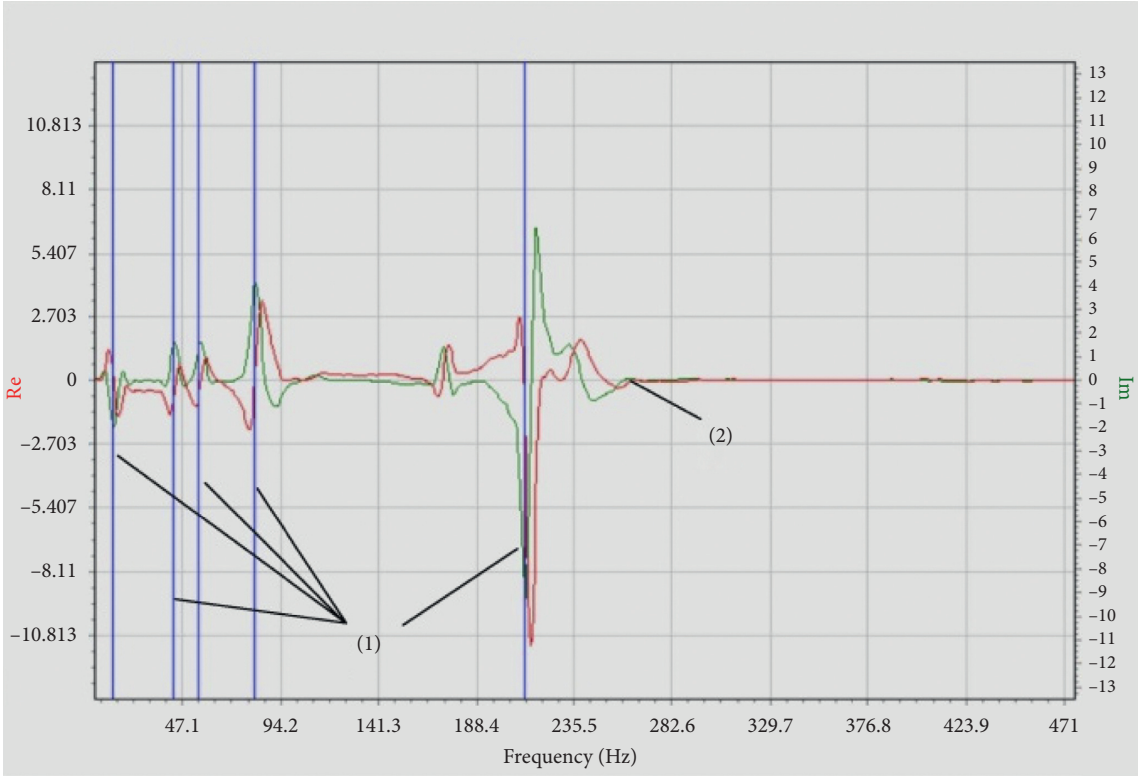


Figure 8: Hammer test: (1) 5-order mode shape; (2) transfer function curve.

Table 2: Statistics of first-order natural frequency of the vehicle-ground system (unit: Hz).

Number of hammers	First order natural frequency	Average
1	211.128	
2	211.792	
3	212.235	
4	211.885	211.8998
5	212.127	
6	211.787	
7	211.838	

Linear fitting is carried out on the frequency value of the original road roughness excitation, and the actual measured peak frequencies of the effective roughness are compared with the four peak frequencies of the main frequency band of the vibration signal, as shown in Figure 11. It can be seen that the difference between the three is small, and with the change of rectangular pit depth, the excitation frequency and vibration frequency values do not change significantly. With the increase of the rectangular pit span, the excitation frequency and vibration frequency values decrease. This shows

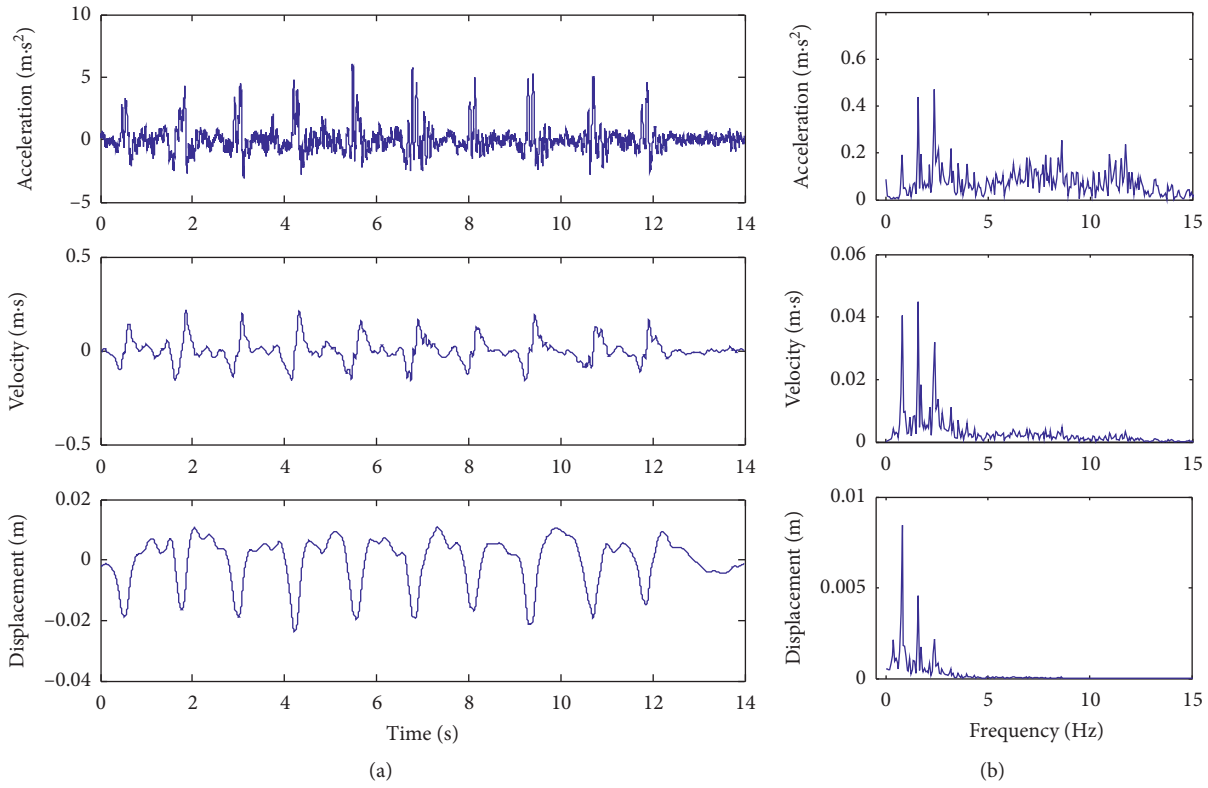


Figure 9: Primary and secondary frequency domain integration and time and frequency correspondence diagram of vibration acceleration signal. (a) Time domain diagram. (b) Frequency domain diagram.

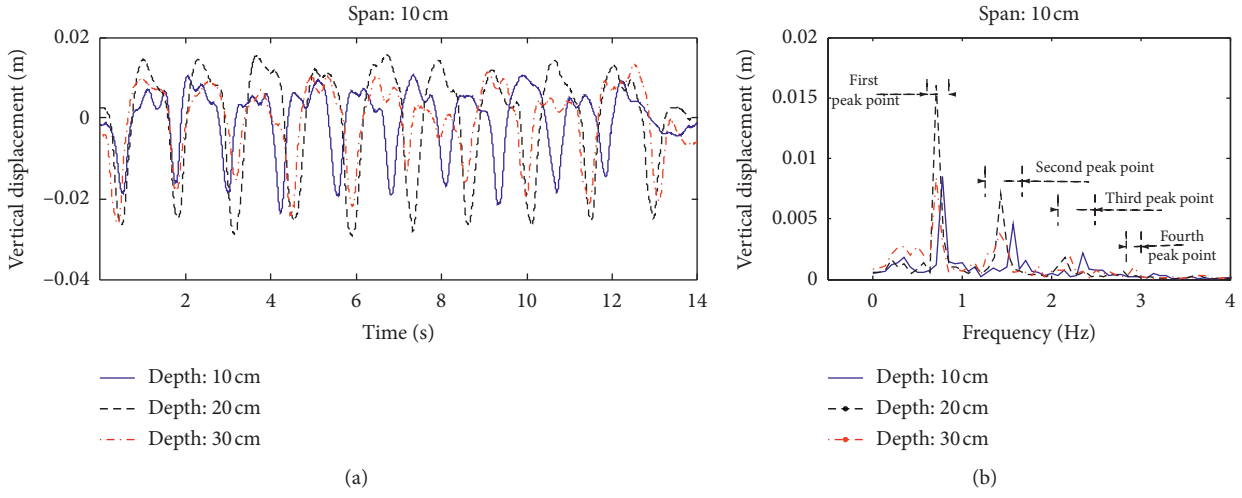


Figure 10: Time-frequency correspondence diagram of effective roughness. (a) Time domain diagram. (b) Frequency domain diagram.

that the change of rectangular pit span is the main reason to change the road wavelength, indirectly changing the excitation frequency of effective uneven and vibration frequency value of vehicles. At the same time, the frequency of effective roughness excitation points is slightly lower than that of the original pavement excitation points. There are mainly two reasons: (1) systematic errors will occur when the application of measuring tools and shovels cannot be completely unified when the road surface is constructed manually. (2) In the process of test, tires and soil will deform and increase the wavelength of the original road surface, resulting in the effective roughness excitation frequency (consistent with the vibration frequency) being smaller than the original road surface roughness excitation frequency value.

4.2. Correlation Analysis of Vibration Acceleration Intensity and Effective Roughness Intensity. MATLAB combination equation (2) is used to solve and make statistics on the root mean square value of acceleration of 21 kinds of road surface

Table 3: Statistics of calculation results of excitation point wavelength, spatial frequency, and quadruple frequency of original pavement roughness signal.

Span (unit: cm)	Wavelength, λ (unit: m)	Spatial frequency, n (unit: m^{-1})	Quadruple frequency time frequency (unit: Hz)			
			Fundamental	Double	Triple	Quadruple
10	0.7	1.429	0.794	1.588	2.382	3.176
20	0.8	1.250	0.694	1.388	2.082	2.776
30	0.9	1.111	0.617	1.234	1.851	2.464
40	1.0	1.000	0.556	1.112	1.668	2.224
50	1.1	0.909	0.505	1.010	1.515	2.020
60	1.2	0.833	0.463	0.928	1.391	1.854
70	1.3	0.769	0.427	0.854	1.281	1.708

Table 4: Regression analysis of effective roughness peak point frequency and original roughness excitation point and vibration signal main frequency band peak point frequency.

Depth (cm)	Peak point	Frequency regression analysis of effective roughness and original roughness			Regression analysis of effective roughness and main frequency band of vibration signal		
		Regression equation	Standard error	R^2	Regression equation	Standard error	R^2
10	First peak point	$y = 1.0009x - 0.0482$	0.027790	0.964080	$y = 0.9730x + 0.0174$	0.012979	0.992165
	Second peak point	$y = 1.0619x - 0.1499$	0.034771	0.987183	$y = 0.9772x + 0.0309$	0.007792	0.999356
	Third peak point	$y = 1.0736x - 0.2628$	0.057738	0.983743	$y = 0.9639x + 0.0704$	0.025687	0.996959
	Fourth peak point	$y = 0.8164x + 0.1873$	0.095525	0.976495	$y = 1.0134x - 0.0394$	0.034624	0.996912
20	First peak point	$y = 0.8768x + 0.0110$	0.007053	0.996882	$y = x$	0	1
	Second peak point	$y = 0.8730x + 0.0407$	0.032586	0.983410	$y = 0.9867x + 0.0124$	0.003107	0.999849
	Third peak point	$y = 0.9124x - 0.0119$	0.035318	0.992006	$y = 0.9872x + 0.0177$	0.004976	0.999841
	Fourth peak point	$y = 0.6540x + 0.5266$	0.047584	0.991348	$y = 0.9846x + 0.0364$	0.02293	0.997991
30	First peak point	$y = 0.8768x + 0.0110$	0.007053	0.996882	$y = x$	0	1
	Second peak point	$y = 0.9117x - 0.0033$	0.036182	0.981283	$y = x$	0	1
	Third peak point	$y = 0.9431x - 0.0654$	0.035183	0.992572	$y = x$	0	1
	Fourth peak point	$y = 0.7153x + 0.3941$	0.035697	0.995736	$y = x$	0	1

data, as shown in Figure 11(a). Combined with equation (5), the effective irregularity power spectral density value is solved and counted, as shown in Figure 11(b). It can be seen from the test data of 7 spans with a depth of 30 cm and the depth of 10~20 cm and the span of 10~40 cm that, with the increase of the span, the weighted root mean square value of vibration acceleration and the power spectral density value of effective roughness of road surface increase. This is due to the increase of rectangular pit size, which leads to the increase of wheel vertical displacement and effective roughness power value. At the same time, the vertical displacement of the wheels increases, which results in an increase in the shear force given to the wheels by the soil [6]. As a result, an increase in the vertical moving load on the vibration test bench occurs when driving on the road [34], so the instantaneous excitation suffered by the wheels when passing through the rear end of the rectangular pit increases, thus enhancing the vibration intensity of the vehicle. As shown in Figure 12(b), with this increase of the depth of the rectangular pit, the amount of subsidence at the front and rear end of the rectangular pit increases, so that the effective roughness power spectral density value increases accordingly. However, with the increase of the depth of the rectangular pit, the weighted root mean square value of the vibration acceleration signal does not change significantly, as shown in Figure 12(a). It is mainly because the effective roughness is basically equal to the vertical displacement of the wheel, and it shows that the acceleration signal is mainly noise at 4~13 Hz since the amplitude of the acceleration signal and the effective roughness signal are mainly concentrated in 0~3 Hz, as well as the coupling between the vibration acceleration signal frequency and the effective roughness excitation frequency. Therefore, there are errors in the acceleration signal measured by the vibration testing system, resulting in irregular changes in the root mean square value of the acceleration signal with the increase of the depth of the road rectangular pit. At the same time, two integrals are carried out in the process of solving the effective roughness, which will reduce the noise part and improve the prediction accuracy of the effective roughness of soft pavement. Therefore, with the increase of the depth of rectangular pit, the power spectral density value of the effective roughness increases obviously. When the depth is 10~20 cm and the span is 50 cm, 60 cm, and 70 cm, the vibration amplitude is small because the wheel collapses too fast when passing through the front end of the rectangular pit. In the meantime, the wheel will contact the bottom of the rectangular pit, the collapsed soft soil will fall on the bottom of the rectangular pit as a buffer, and the rear end of the rectangular pit cannot directly give excitation, resulting in the process which cannot generate large excitation. It shows that, in the complete excitation of the rectangular pit (in this paper, the test data of 7 spans with a depth of 30 cm and the test data with a depth of 10~20 cm

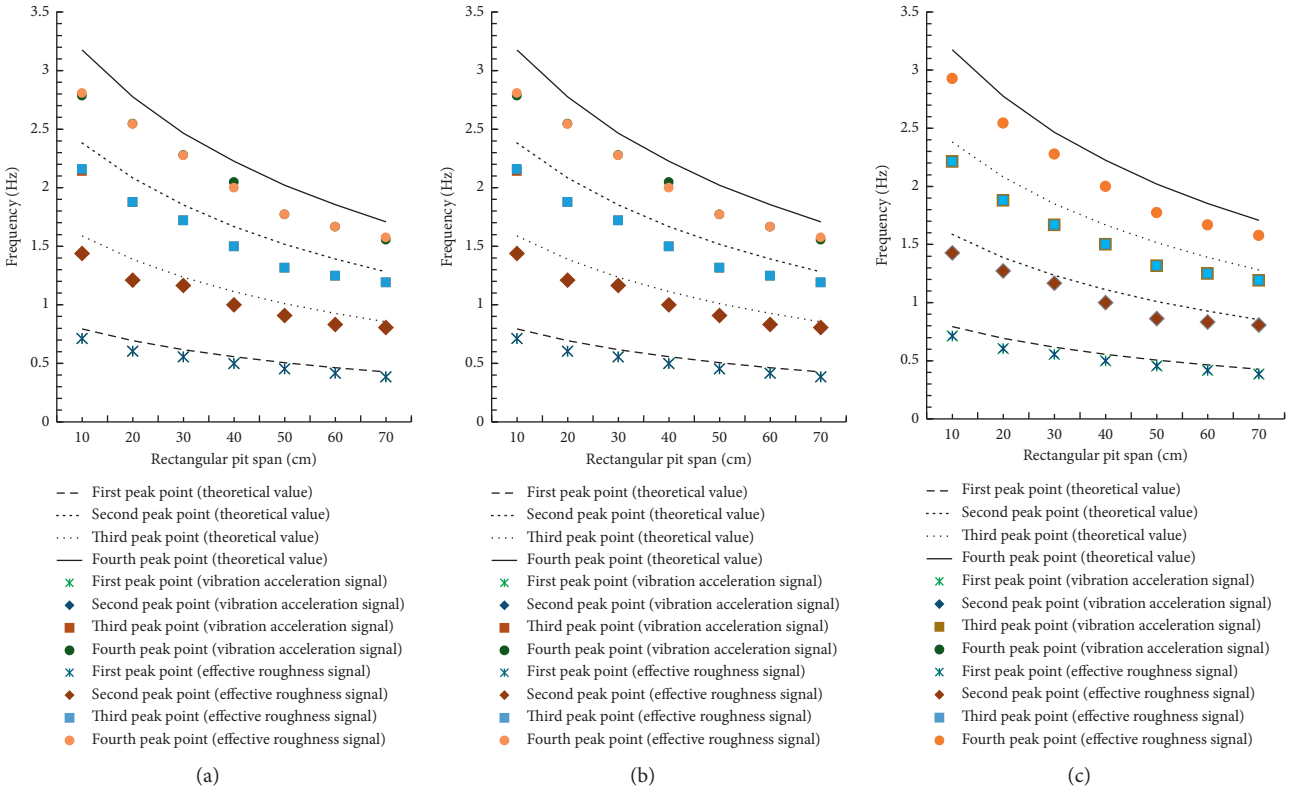


Figure 11: Comparison of the frequency value of original road roughness excitation point, frequency value of effective roughness peak point, and frequency value of vibration peak point. (a) Depth: 10 cm. (b) Depth: 20 cm. (c) Depth: 30 cm.

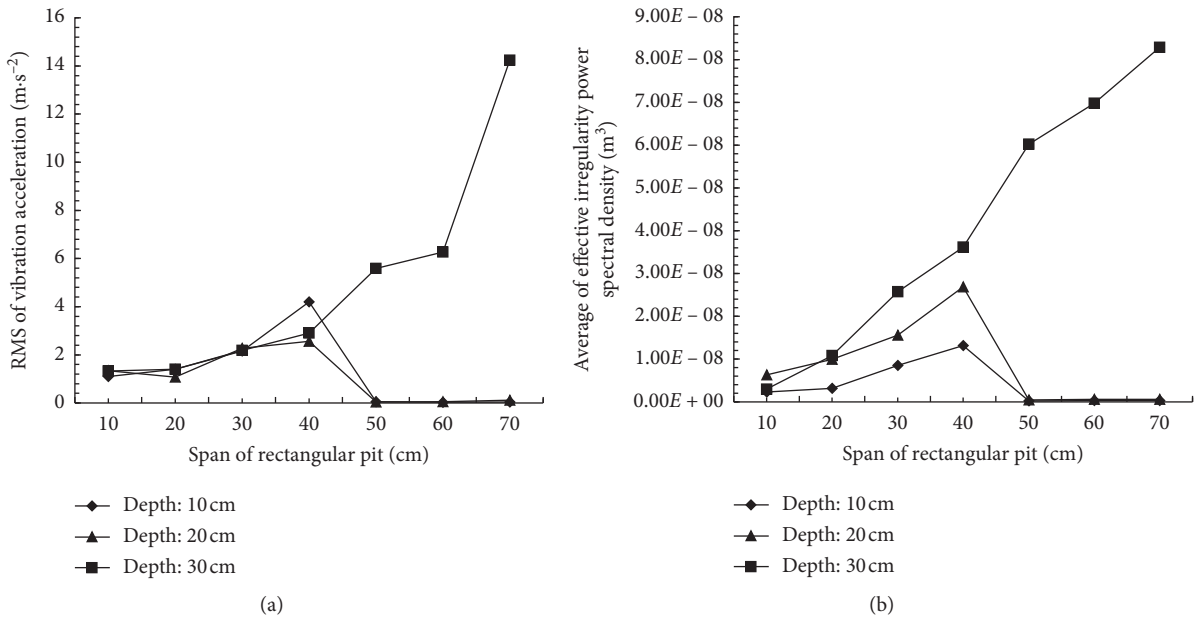


Figure 12: Vibration response and road excitation law. (a) RMS of vibration acceleration, (b) Average of effective irregularity power spectral density.

and a span of 10~40 cm are called complete excitation of rectangular pit, and the rest data are called incomplete excitation of rectangular pit), the increase of rectangular pit span is the main reason for the increase of vehicle vibration power.

We continue to carry out statistical analysis on the vertical displacement corresponding to the four peak points of the power value of the four peak points of the first band of the vibration signal and the four peak points of the effective roughness. Figure 13 is a rectangular pit road surface with a

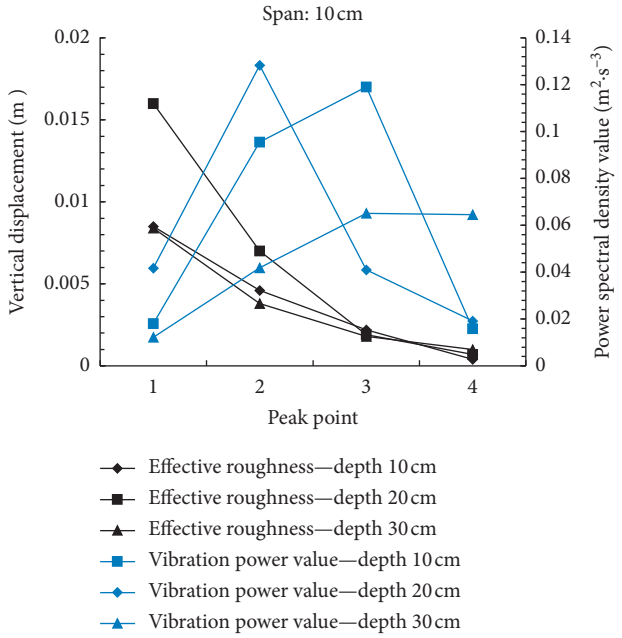


Figure 13: Vertical displacement of vibration power value and effective roughness corresponding to peak point.

span of 10 cm and a depth of 10 cm, 20 cm, and 30 cm. It can be seen that, with the increase of the peak point, the vibration power value increases first and then decreases. The highest peak is the second peak point and the third peak point. The vertical displacement of the effective roughness excitation point decreases in turn. The maximum value is at the first peak point, which conforms to the excitation law of the constructed pavement, and the other data laws are consistent. The main reason is that the effective irregularity excitation characteristics mainly depend on the vertical displacement of the wheel, and the first peak point is amplified in turn and the other peak points are reduced in turn in the two integration processes. It shows that the double frequency and triple frequency of road excitation are the main frequencies that cause the maximum vibration power of vehicles. Moreover, in the frequency domain, the output signal is the product of the frequency response function of the system and the input signal [29], indicating that the frequency response function of the system is the most active in the excitation of frequency doubling 2 and frequency doubling 3. At the same time, the vibration generated by the ground excitation of the whole tractor is in the low frequency band (0~3 Hz) [21, 24, 31–33]. When the acceleration signal is integrated in the frequency domain, the signal in the 0~1 Hz frequency band would be amplified, and the signal in the 1~3 Hz frequency band would be reduced, which results that the soil soft road excitation corresponding to the highest peak of the vehicle vibration response is not the fundamental frequency. At the same time, it provides a reference for the prediction of soft soil pavement excitation and the design of tractor vibration reduction.

5. Conclusions

In this paper, the constructed rectangular pit pavement is used to simulate the field soft pavement with different wavelengths, and the test is carried out in combination with the vibration test system. The effective roughness model is established through the vibration equation of the vehicle-ground system, and the effective roughness of soft pavement is obtained, which is used as the excitation characteristic of pavement. In order to explore the correlation between the excitation characteristics of different long-wave roads in the field and the vibration response of vehicles, the following conclusions are drawn.

(1) In the full excitation of the rectangular pit, the increase of the span of the rectangular pit leads to the increase of the effective irregularity excitation wavelength and the decrease of the excitation frequency given to the wheel. At the same time, the increase of rectangular pit span leads to the increase of vertical moving load during the driving process of the vehicle, resulting in the increase of vehicle vibration intensity. In the incomplete excitation of the rectangular pit, the collapse of the front end of the rectangular pit will form a buffer zone. The wheel cannot directly contact the rear end of the rectangular pit due to the contact with the bottom of the rectangular pit, which is difficult to cause large excitation.

(2) The correlation coefficient (R^2) between the frequency of the peak point of effective roughness and the frequency of the excitation point at the back end of the rectangular pit is 0.9641~0.9969, and the frequency range is 0~3 Hz. This shows the effectiveness of the method for solving the effective roughness of soft soil pavement and determines the rear end of rectangular pit as the main excitation point. The correlation coefficient (R^2) between the main frequency band frequency of the vibration test bench response acceleration signal and the effective roughness excitation frequency ranges from 0.992165 to 1. It indicates that the vibration frequency generated by agricultural vehicles traveling on soft pavement is mainly determined by the excitation frequency of soft pavement, which provides a basis for predicting the excitation characteristics of soft pavement.

(3) Due to the periodic characteristics of the arrangement of rectangular pit excitation points, the vibration acceleration signal and the effective roughness excitation signal have the generation of double frequency band. The highest point of vehicle vibration acceleration power value is concentrated at the frequency doubling and frequency doubling points, and the fundamental frequency amplitude of effective roughness excitation on soft road surface in the field is the highest. The results show that the maximum value of vehicle vibration acceleration response and the maximum value of field soft pavement excitation have a relationship of 2 or 3 times in frequency distribution, which provides reference for the prediction of pavement excitation frequency and the design of vehicle shock absorption.

In this paper, the vehicle vibration response is used to evaluate the correlation between the excitation characteristics of the soft road surface and the vehicle vibration, but it is difficult to determine the effective wavelength range. Future plan will solve this issue by using the binocular visual odometer algorithm to obtain the spatial motion of the vehicle. A computer vision system will be developed to reduce the agriculture vehicles in the soft-filed roads.

Data Availability

All data that were used to produce the results in this paper are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was mainly supported by the National Natural Science Foundation of China (51565052) and part of the funding comes from the open project of Key Laboratory of Modern Agricultural Engineering (TDNG20170301) of Tarim University in Xinjiang Uygur Autonomous Region, China.

References

[1] M. Jiang, X. Hu, C. Joseph, Z. Lin, and R. Fei, "Does the popularization of agricultural mechanization improve energy-environment performance in China's agricultural sector?," *Journal of Cleaner Production*, vol. 276, Article ID 124210, 2020.

[2] R. C. Ugras, O. K. Alkan, S. Orhan, M. Kutlu, and A. Mugan, "Real time high cycle fatigue estimation algorithm and load history monitoring for vehicles by the use of frequency domain methods," *Mechanical Systems and Signal Processing*, vol. 118, pp. 290–304, 2019.

[3] S. Milosavljevic, F. Bergman, B. Rehn, and B. Allan, "Carman. All-terrain vehicle use in agriculture: exposure to whole body vibration and mechanical shock," *Applied Ergonomics*, vol. 41, pp. 530–535, 2010.

[4] B. Mariotti, Y. Hoshika, M. Cambi et al., "Vehicle-induced compaction of forest soil affects plant morphological and physiological attributes: a meta-analysis," *Forest Ecology and Management*, vol. 462, Article ID 118004, 2020.

[5] P. R. Hargreaves, K. L. Baker, A. Graceson, S. Bonnett, B. C. Ball, and J. M. Cloy, "Soil compaction effects on grassland silage yields and soil structure under different levels of compaction over three years," *European Journal of Agronomy*, vol. 109, Article ID 125916, 2019.

[6] W. K. Carsten Hoever, "A model for investigating the influence of road surface texture and tyre tread pattern on rolling resistance," *Journal of Sound and Vibration*, vol. 351, pp. 161–176, 2015.

[7] R. Zhao, L. Hu, X. Luo et al., "A novel approach for describing and classifying the unevenness of the bottom layer of paddy fields," *Computers and Electronics in Agriculture*, vol. 162, pp. 552–560, 2019.

[8] Y. Qin, Z. Wang, C. Xiang, E. Hashemi, A. Khajepour, and Y. Huang, "Speed independent road classification strategy based on vehicle response: theory and experimental validation," *Mechanical Systems and Signal Processing*, vol. 117, pp. 653–666, 2019.

[9] Y. Qin, C. Wei, X. Tang, N. Zhang, M. Dong, and C. Hu, "A novel nonlinear road profile classification approach for controllable suspension system: simulation and experimental validation," *Mechanical Systems and Signal Processing*, vol. 125, pp. 79–98, 2018.

[10] M. A. Bidgoli, A. Golroo, H. S. Nadjar, A. G. Rashidabad, and M. R. Ganji, "Road roughness measurement using a cost-effective sensor-based monitoring system," *Automation in Construction*, vol. 104, pp. 140–152, 2019.

[11] T. E. Putra, Husaini, and M. N. Machmud, "Predicting the fatigue life of an automotive coil spring considering road surface roughness," *Engineering Failure Analysis*, vol. 116, Article ID 104722, 2020.

[12] A. Kheirati and A. Golroo, "Low-cost infrared-based pavement roughness data acquisition for low volume roads," *Automation in Construction*, vol. 119, Article ID 103363, 2020.

[13] G. Xue, H. Zhu, Z. Hu et al., "Pothole in the dark: perceiving pothole profiles with participatory urban vehicles," *IEEE Transactions on Mobile Computing*, vol. 16, no. 5, 2016.

[14] W. Shi, M. Li, J. Guo, and K. Zhai, "Evaluation of road service performance based on human perception of vibration while driving vehicle," *Journal of Advanced Transportation*, vol. 2020, Article ID 8825355, 8 pages, 2020.

[15] L. Guo, T. Li, G. Chen, P. Yu, X. Peng, and D. Yang, "A method for microscopic unsaturated soil-water interaction analysis based on DDA," *Computers and Geotechnics*, vol. 108, pp. 143–151, 2019.

[16] L. Zheng, L. Mei, Y. Liu, Z. Zhang, and Y. Zhang, "A measuring equipment of terrain roughness and its dynamic characteristic analysis," *Transactions of the Chinese Society for Agricultural Machinery*, vol. 2, pp. 68–74, 1991.

[17] Y. Yin, L. Zheng, F. Gong, Y. Zhang, and J. Yin, "The principle of rough terrain excitation acting on vehicle wheels," *Transactions of the Chinese Society for Agricultural Machinery*, vol. 3, pp. 72–77, 1991.

[18] L. Zheng, Y. Cheng, L. Mei, Z. Zhang, and J. Yin, "Statistics analysis of the effective spectrum of the soft terrain surface," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 4, p. 2027, 1991.

[19] L. Zheng, M. Liu, and Y. Zhang, "Formative process analysis of the effective spectrum of the soft terrain surface," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 2, pp. 29–34, 1994.

[20] M. Liu, Y. Cheng, D. Wang, L. Zheng, Y. Zhang, and Z. Wang, "The theoretical trans form method of from original spectrum to effective spectrum of terrain," *Transactions of the Chinese Society for Agricultural Machinery*, vol. 4, pp. 11–17, 1993.

[21] Z. Fan, S. Li, F. Ma, L. Wan, L. Xiao, and B. Xue, "Simulation analysis and experiment on influence of road spectrum of cane field on cutting quality," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 28, no. 1, pp. 37–41, 2012.

[22] S. Zhu, J. Ma, J. Yuan, G. Xu, Y. Zhou, and X. Deng, "Vibration characteristics of tractor in condition of paddy operation," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 32, no. 11, pp. 31–38, 2016.

[23] Y. Yilidaer, S. Zhu, C. Minh, X. Nie, and G. Xu, "Study on vibration characteristics of tractor travel at the condition of marshy paddy soil," *Journal of Nanjing Agricultural University*, vol. 39, no. 6, pp. 1062–1068, 2016.

[24] M. Cutini, C. Costa, and C. Bisaglia, "Development of a simplified method for evaluating agricultural tractor's operator whole body vibration," *Journal of Terramechanics*, vol. 63, pp. 23–32, 2016.

[25] C. Hu, X. Wang, S. Wang et al., "Impact of agricultural residual plastic film on the growth and yield of drip-irrigated cotton in arid region of Xinjiang, China," *International Journal of Agricultural and Biological Engineering*, vol. 13, no. 1, , 2020, https://www.ijabe.org.

[26] ISO 5008, *Agricultural Wheeled Tractors and Field Machinery–Measurement of Whole-Body Vibration of the Operator*, 2002.

[27] Y. Zhisheng, *Vehicle Theory*, Machinery Industry Press, Beijing, China, 3rd edition, 2004.

[28] J. Li, Z. Zhou, and L. Yan, "Test analysis of relationship between natural frequency and compaction degree for roadbed soil," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 28, no. 14, pp. 71–76, 2012.

[29] J. Wang and X. Hu, *Application of Matlab in Vibration Signal Analysis*, China Water Resources and Hydropower Press, Intellectual Property Publishing House, Beijing, China, 2006.

[30] G. Wen, L. Yang, Y. Liao, and Q. He, "Faulty rotor system vibration acceleration signal integration method based on precise information reconstruction," *Journal of Mechanical Engineering*, vol. 49, no. 8, pp. 1–9, 2013.

[31] R. Deboli, A. Calvo, and C. Preti, "Whole-body vibration: measurement of horizontal and vertical transmissibility of an agricultural tractor seat," *International Journal of Industrial Ergonomics*, vol. 58, Article ID 69e78, 2017.

[32] S. Aisyah Adam and N. A. Abdul Jalil, "Vertical suspension seat transmissibility and SEAT values for seated person exposed to whole-body vibration in agricultural tractor preliminary study," *Procedia Engineering*, vol. 170, pp. 435–442, 2017.

[33] E. Zheng, X. Zhong, R. Zhu et al., "Investigation into the vibration characteristics of agricultural wheeled tractor-implement system with hydro-pneumatic suspension on the front axle," *Biosystems Engineering*, vol. 186, Article ID 14e33, 2019.

[34] H. Zhu, Y. Zhou, and Y. Hu, "Displacement reconstruction from measured accelerations and accuracy,control of integration based on a low-frequency attenuation algorithm," *Soil Dynamics and Earthquake Engineering*, vol. 133, Article ID 106122, 2020.

[35] G. Liu, S.-x. Chen, and W.-d. Luo, "Synchronous sampling measuring method of the single-cycle RMS of motor non-sinusoidal power supply voltage," *Electric Machines and Control*, vol. 18, no. 1, pp. 112–116, 2014.

[36] Z. Lyu, J. Qian, Z. Shi, and Q. Gao, "Dynamic responses of layered poroelastic ground under moving traffic loads considering effects of pavement roughness," *Soil Dynamics and Earthquake Engineering*, vol. 130, Article ID 105996, 2020.

Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 8877138, 15 pages
https://doi.org/10.1155/2021/8877138

WILEY | Hindawi

Research Article

Dynamic Path Flow Estimation Using Automatic Vehicle Identification and Probe Vehicle Trajectory Data: A 3D Convolutional Neural Network Model

Can Chen, Yumin Cao, Keshuang Tang, and Keping Li

Key Laboratory of Road and Traffic Engineering in the Ministry of Education, Tongji University, Shanghai 201804, China

Correspondence should be addressed to Keshuang Tang; tang@tongji.edu.cn

Received 1 October 2020; Revised 23 December 2020; Accepted 22 January 2021; Published 8 February 2021

Academic Editor: Wen LIU; wenliu@whut.edu.cn

Dynamic path flows, referring to the number of vehicles that choose each path in a network over time, are generally estimated with the partial observations as the input. The automatic vehicle identification (AVI) system and probe vehicle trajectories are now popular and can provide rich and complementary trip information, but the data fusion was rarely explored. Therefore, in this paper, the dynamic path flow estimation is based on these two data sources and transformed into a feature learning problem. To fuse the two data sources belonging to different detection ways at the data level, the virtual AVI points, analogous to the real AVI points (turning movements at nodes with AVI detectors), are defined and selected to statically observe the dynamic movement of the probe vehicles. The corresponding selection principles and a programming model considering the distribution of real AVI points are first established. The selected virtual AVI points are used to construct the input tensor, and the turning movement-based observations from both the data sources can be extracted and fused. Then, a three-dimensional (3D) convolutional neural network (CNN) model is designed to exploit the hidden patterns from the tensor and establish the high-dimensional correlations with path flows. As the path flow labels commonly with noises, the bootstrapping method is adopted for model training and the corresponding relabeling principle is defined to purify the noisy labels. The entire model is extensively tested based on a realistic road network, and the results show that the designed CNN model with the presented data fusion method can perform well in training time and estimation accuracy. The robustness of a model to noisy labels is also improved through the bootstrapping method. The dynamic path flows estimated by the trained model can be applied to travel information provision, proactive route guidance, and signal control with high real-time requirements.

1. Introduction

Unlike the static path flows, which represent the average path flows during a long period, the dynamic path flows represent the real-time path flows in a relatively small time interval and the corresponding estimation problem becomes more challenging. The dynamic path flow data have a wide range of applications, such as the analysis of user travel patterns, large-scale traffic network simulation, and traffic planning and management. The path flow and OD matrix estimates are sometimes similar and interdependent. The OD flows can be assigned to obtain path flows and the sum of several path flows can be used to obtain one specific OD flow. Due to the high cost and less efficiency of manual survey for directly observing path flows, a typical way has been widely used to indirectly estimate them from observed link flows.

For this way, numerous mathematical programming models have been established to solve the OD demands that are most consistent with observed link flows under certain assumptions, but the results remain unsatisfactory. The major reason is that the information provided by the observed links is limited. It is common for a network that the number of OD pairs is much larger than the number of observed links [1, 2], which is often referred to the under-specified problem. More information such as prior OD matrices is needed and the bi-level programming framework is mostly applied among existing studies [3, 4]. At the upper

level, optimal solutions are searched according to certain objectives that minimize the distance between the possible estimates and observed values, including generalized least squares, entropy maximization, etc. At the lower level where dynamic traffic assignment (DTA) is usually conducted, estimated OD flows are mapped to observed link flows. Unfortunately, without observed trip information as constraints, the assumptions related to path choice and the driving behavior of a traveler using the DTA model may be inconsistent with realistic conditions [5].

Compared with link traffic counts, the AVI data and probe vehicle trajectories can provide more information about trips. The current AVI system is generally composed by radio frequency identification device-based detector, Bluetooth detector, and video-based detector. The unique vehicle's ID and passing time can be recorded with the vehicle passing the detector. By matching vehicles' IDs, the traffic counts and travel time of partial paths can be further obtained. In the current studies concerning path and OD flow estimations, the AVI information is generally incorporated in the following three ways: matched volume as prior observed OD data [6, 7] or observed counts of partial complete paths [2, 8]; the link choice [4] or lagged observation proportion based on travel time [6]; and recorded link volume [2, 8]. Compared with the AVI data, the probe vehicle trajectories are collected by GPS-enabled devices and the prior OD matrices and users' path choice behaviors could be further discovered based on the sampled and complete trajectories. Several researches have been conducted using either vehicle trajectories alone or as fused with the observed link flows [9, 10]. The information provided by the AVI system and the probe vehicles has different characteristics. Taking the video-based AVI system widely applied in China as an example, every vehicle passing the detector can ideally be detected, but the coverage rate and matched vehicles may still rapidly degenerate when the network size increases [8]. Recently, with the rapid development of online car-hailing market, more vehicle trajectories can be collected and the spatial coverage is considerably wide [7], but the comparatively low penetration rate remains a problem [11]. From the discussion, it is evident that the AVI and the probe vehicle systems have complementary characteristics in both spatial coverage and vehicle collection. Hence, fusing these two types of data is promising for dynamic path flow estimation but has rarely been explored.

Typically, dynamic path flow estimation is regarded as an optimization problem that searches for the most consistent path flow estimates and the solution is a high-dimensional time-dependent path flow matrix [12]. For mathematical programming models, the trip information obtained by fusing these two data is supposed to be described analytically. And the model accuracy may improve as rich information is added, yet the solving capacity of the model may become more difficult and inefficient. Considering the rich and implicit travel features stored in the AVI and the probe vehicles data, the path flows estimation can potentially be transformed from an optimization problem to a data-driven feature learning problem. Neural networks (NNs) have long been proven as an effective measure to learn hidden patterns from given samples and make further predictions.

Recently, with the emergence of deep learning algorithms, the NN has undergone a transition from shallow NN to deep NN (NN with multiple layers), which is much more powerful in learning complex and abstract features. Several researchers have explored the use of deep NNs for OD flow estimation. Huang et al. [7] used the long short-term memory model with time sequence of trajectory OD flows as the input. The temporal dependency between successive time steps is considered. The labels of partial OD pairs are provided by AVI data and the label propagation is adopted to infer uncovered OD pairs. An improved 3D-CNN model (Res3D), which can capture the features along both spatial and temporal dimensions, was designed by Tang et al. [13]. The model input is a cube of three stacked matrices, and for each matrix, the set of links installed by AVI detectors is used to represent the segment volume, matched volume, and matched travel time. The training samples are provided by a parallel simulation system and the model can be applied through transfer learning. In these two models, the AVI and trajectory data are still used separately as model inputs. The success of deep NNs relies heavily on supervised training and sufficient labeled data samples. The OD flow labels used by Huang et al. [7] and Tang et al. [13] are, respectively, obtained from AVI data and an exogenous simulator. However, it is difficult to guarantee that the label data are without noises, which is also common in the field of image recognition. And the label noise is potentially more harmful than feature noise, which affects the observed values of the feature [14]. It is inevitable to consider how to learn samples with noisy labels.

As suggested by Tang et al. [13], the 3D-CNN structure is effective for capturing the spatial-temporal patterns in link observation-based tensor. In this paper, the 3D convolutions are also introduced to design a deep NN for estimating the dynamic path flows, but the input tensor is constructed based on the turning movements of network nodes to express and fuse the multiple types of AVI and probe vehicle observations. The contributions of this study are highlighted below:

(1) The AVI and probe vehicle data produced by different detection ways are fused and the mobile detection of probe vehicles is changed to stationary detection by properly arranging the virtual AVI points. The principles for selecting the turning movements as virtual AVI points and a corresponding programming model are established, and with the distribution of real AVI points, data-driven feature learning characteristics and input tensor size related to model efficiency are comprehensively considered.

(2) Based on the selected virtual AVI points, the turning volumes, matched volumes, and matched travel time provided by AVI and probe vehicle data are represented and hierarchically combined in the input tensor to implicitly represent the travel patterns of networks. The concrete architecture of the 3D-CNN

model is designed to automatically recognize the hidden patterns behind these observations and establish a high-dimensional mapping to dynamic path flows.

(3) To cope with noisy path flow labels, especially with systematic errors, a self-correcting algorithm bootstrapping commonly applied for image classification tasks is extended to variable regression problem. The key is the re-labeling process gradually purifying the noisy labels, and the relationship between the generated path flows and the observed turning flows is used to determine whether to correct the original labels with model outputs.

2. Problem Statement

In this paper, we use the characters lowercase (a), bold lowercase ($\mathbf{a}$), uppercase (A), bold uppercase ($\mathbf{A}$), and uppercase calligraphy ($\mathcal{A}$) to represent scalar, vector, set, matrix, and tensor, respectively. Each set has an attribute of length that can be expressed by $|A|$.

Figure 1 illustrates the research background and relevant definitions and variables. We consider an urban road network specified by a directed graph $G = (N, L)$, where N denotes the node set of the network, $N = \{1, \ldots, n\}$, $n \in N$, and L denotes the link set of the network, $L = \{1, \ldots, l\}$, $l \in L$. P is the path set of the network, $P = \{1, \ldots, k\}, k \in P$. The nodes in N which directly connect to traffic zones are attracting or generating nodes and constitute the node set N^{se}. The remaining nodes are cross nodes that represent the real intersections and constitute the node set N^{c}. For each link of node, there are three circles. Along the approaching direction, the three circles represent left, straight, and right turns, from left to right. The white circle means this approach has no this turning movement; otherwise, the circle is filled with gray. A is the set of all turning movements, $A = \{1, \ldots, a\}, a \in A$. A^{se} is a subset of A and contains the turning movements of nodes in N^{se}. The purple circle represents a real AVI point where the turning vehicles can be detected by a video-based AVI system. Taking the link l as an example, several activated lines are drawn near the stop line in the shooting screens of cameras to automatically detect and record the passing vehicles' license plate numbers and time. The set of real AVI points is denoted by A^{r}.

The vehicle turning information for each a in A^{r} can be directly obtained, and Alibabai and Mahmassani [15] point out that using intersection turning movements as the basic field observation instead of links can result in more reliable estimates of dynamic OD matrices. Hence, the turning movements of nodes are selected as the basic network elements to construct the input tensor and represent the AVI and probe vehicle observations. However, compared with the stationary AVI detection points, the occurrence and movement of the probe vehicle are dynamic. It is necessary to study as to how to select the turning movements from A to constitute the virtual AVI point set denoted by A^{v}. For the virtual AVI point (circle filled with green), it can be thought that a virtual AVI detector is installed to specially detect the probe vehicles. The turning movement and passing time at the node for each probe vehicle with unique ID can also be got through the mapping between the GPS coordinates and links. The real and virtual AVI points can overlap (circles filled with yellow). Hence, for one network G, the selection of A^{v} based on the distribution of A^{r} is important not only for the representation of probe vehicle data but also for the fusion with AVI data.

The 3D-CNN is generally used to process consecutive video frames for action recognition. The corresponding input is a cube formed by stacking multiple contiguous frames together. For each frame, like a still image, it is represented by a three-dimensional tensor involving the height, width, and channels of the image. For CNNs applied to the transportation engineering field, in addition to the model architecture design, the first problem is to organize the traffic data based on the requirements of CNN. In the network traffic state estimation, it is common but a little "rough" to grid the network and use the heatmap as the input. Based on the virtual AVI point set A^{v}, the selection and combination of the turning movement-based observations provided by AVI and probe vehicle systems in input tensor of NN is important for the implicit representation of the road network.

The prior path flows commonly used in traditional mathematical models can be taken as one of the sources of path flow labels. From the viewpoint of estimation accuracy, it is desirable that the prior path flows should be a close estimate of the true path flows. However, Yang et al. [3] pointed out that prior OD flows may have random and systematic errors compared with the true values. This is also true for path flows. Some methods have been developed to deal with the problem of noisy labels and can be approximately divided into three categories: (1) noise-robust models, (2) data cleaning methods, and (3) noise-tolerant learning algorithms [14]. The noise-robust models rely on initializing the start point of gradient descent and early stopping mechanism to avoid (over) fitting to noisy labels. The second category aims to improve the quality of training data by identifying and removing the samples with incorrect labels. Considering the difficulties of obtaining training samples paired with labels and the desire to make full use of each sample's information, the third category's methods are focused and the bootstrapping method which has been used by Reed et al. [16] and Wu et al. [17] is extended from the classification problem to the variable regression problem.

3. Methodology

To address the problems summarized in the previous section, the modeling process is illustrated in Figure 2 and there are three main steps. In Step 1, the principles of optimally selecting the turning movements as virtual AVI points to fuse the AVI and the probe vehicle data is specified and modeled as a binary linear programming model. In Step 2, the AVI and probe vehicle observations based on turning movements are selected and organized in the input tensor. And the architecture of the designed 3D-CNN model and working process are explained. Finally, for path flow labels with errors, the bootstrapping method which can re-label

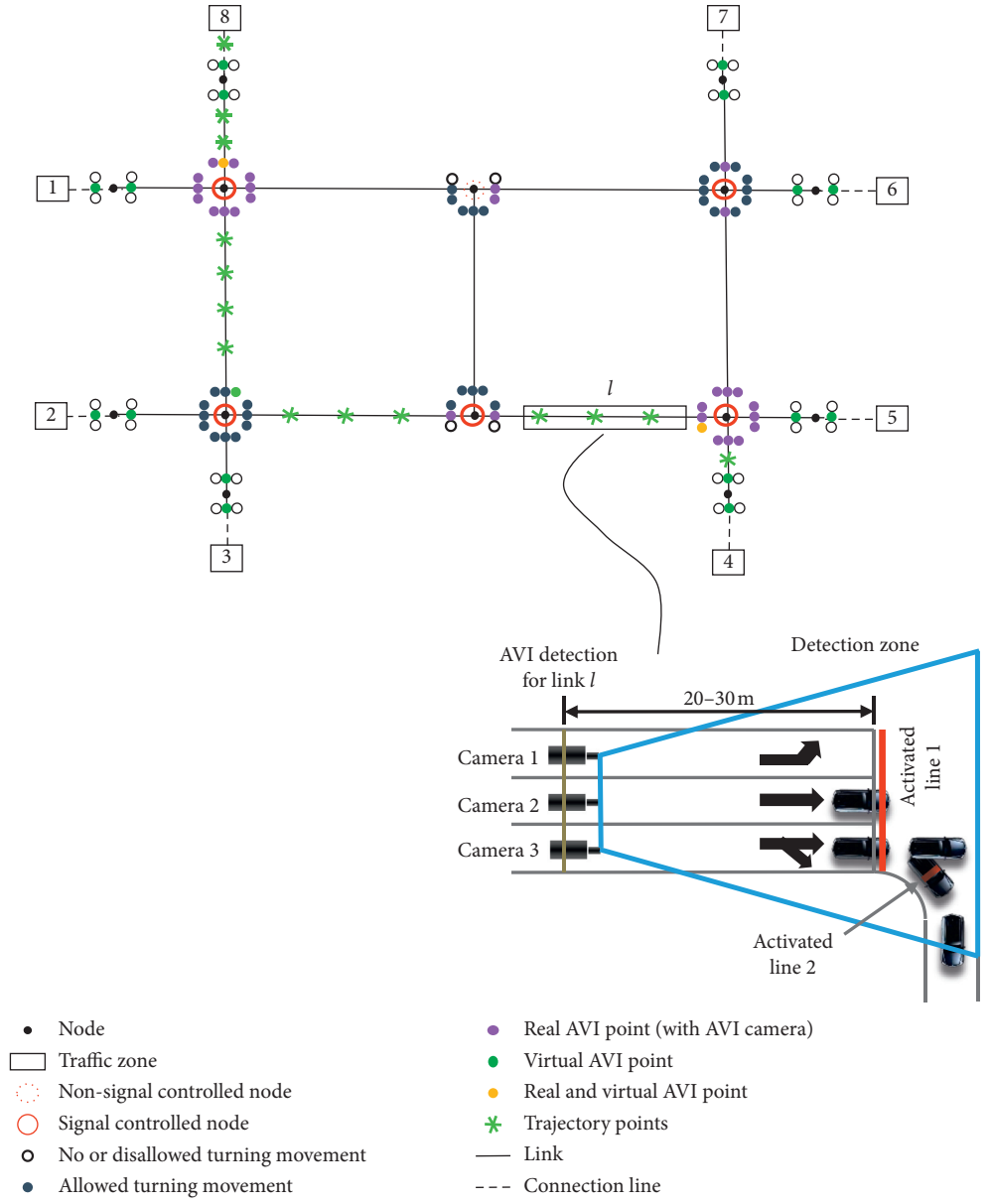


Figure 1: Illustration of the research problem.

and gradually purify the noisy labels is used to train the model.

Step 1. Virtual AVI Point Selection

The selected virtual AVI points are directly used to construct the input tensor of NN. Hence, except for the trip information of probe vehicles, the distribution of real AVI points, feature learning characteristics, and tensor size should also be considered.

Based on the algorithm proposed by Castillo et al. [2] about selecting links to be scanned for predicting path flows, a revised binary linear programming model is established. The model is shown as follows:

$$\min Z = \sum_{a \in A} \left(u_a^v + pa^{\max} * u_a^v * u_a^{se} - 0.5 * u_a^v * u_a^r\right). \quad (1)$$

subject to

$$\sum_{a \in A} u_a^v * b_a^k \geq 1, \quad \forall k \in P, \quad (2)$$

$$\sum_{a \in A} u_a^v * b_a^{k_1} * \mathrm{d}\left(k_1, k_2, a\right) \geq 1, \quad \forall k_1, k_2 \in P \text{ and } k_1 \neq k_2, \quad (3)$$

$$b_a^k = \begin{cases} 1, & \text{if path } k \text{ contains } a, \\ 0, & \text{otherwise}, \end{cases} \quad (4)$$

$$d\left(k_1, k_2, a\right) = \begin{cases} 1, & \text{if } b_a^{k_1} \neq b_a^{k_2}, \\ 0, & \text{otherwise}, \end{cases} \quad (5)$$

where u_a^v is the binary decision variable such that it takes the value 1 when the turning movement a is selected as the virtual AVI point, and 0, otherwise; u_a^{se} and u_a^r are 1 when a belongs to A^{se} and A^r, and 0, otherwise; $pa^{\max}$ is the maximum number of turning movements traversed by one path; b_a^k belongs to $\mathbf{B}$, which describes the relationship between the turning movements and each path of P.

The algorithm proposed by Castillo et al. [2] minimizes the number of scanned links with respect to two principles. Here, three principles are presented as follows:

(1) Any path k of P contains at least one virtual AVI point;

(2) For any two paths k_1 and k_2 of P, each path can differ from the other through the minimum selected virtual AVI points of itself;

(3) Attempt to select the minimum virtual AVI points that are not in A^{se} set and make the selected points overlap more real AVI points.

The first principle is similar to that given by Castillo et al. [2] and realized by Constraint (2). For the second principle, Castillo et al. [2] allowed the scanned link which is in path k_1 and not in path k_2 or vice versa to distinguish the two paths. This means that for the two pairs of paths (k_1, k_2) and (k_2, k_1), the selected links can be the same. This is reasonable because the relationship between the path flows and observed counts is expressed by mathematical formulations and each path is guaranteed to have at least one scanned link. However, too many scanned links can potentially be assigned to several paths or paths with low demands. For the deep NN used in this study, the information of each path should be independent and integrated. Hence, each path should depend on the turning movements of itself to differ from other paths and for (k_1, k_2) and (k_2, k_1), the selected turning movements are different. Equations (3) and (5) can ensure this principle.

The last principle is reflected in the objective function (1), which includes all the three terms. To minimize the number of selected virtual AVI points, the decision variable u_a^v must be added as the first term. The second term is to constrain the selected virtual AVI points not from A^{se}, except in the cases where only the turning movements that belong to A^{se} can distinguish the two paths. This setting can help focus on the operation of probe vehicles in real intersections. The third term urges more overlaps between the virtual and the real AVI points. When the real AVI points are also selected as virtual AVI points, the objective value can be further reduced while satisfying the first two principles.

After solving the optimization model above, only a small part of A^{se} is selected and the dependency of paths and the passed turning movements cannot be completely expressed in the input tensor. This information is also important and useful for the estimation problem. Thus, the residual virtual points of A^{se} are also added to A^v. Then each path has its own start and terminal virtual points and the number of probe vehicles between the start/terminal point and each of the other passed points is the same. Please note that this selection method relies on the exogenous and static path information (path set P and turning movement-path matrix $\mathbf{B}$), which can be obtained through other data sources (e.g., the probe vehicle trajectories) or methods (e.g., the path flow assignment from known OD flows). It is suitable for stable network with not heavy traffic congestions [1, 3, 18]. If the effect of traffic congestion on travel time is significant, the path flow dynamics should be considered inside the model through the flow-related constraints.

Step 2. Input Tensor Construction and Model Architecture Design

3.1. Data Fusion and Input Tensor Construction. There are several uniform and continuous time intervals for estimation denoted by $H = \{1, \ldots, h\}$, $h \in H$. The operation of network traffic within one time interval h can be seen as a video frame (also a still color image). The corresponding abstract expression is the three-dimensional tensor with the AVI and probe vehicle trajectory data merged, as shown in Figure 3. To implicitly express the travel patterns of one network, we define three main types of observations based on turning movements from these two data sources: turning volumes, matched volumes, and matched travel time. The last two observations can represent the local and global end-to-end trip information. As for the turning volumes, it helps to express richer information.

From Figure 3, we can see that there are three square matrices like the three channels of one color image and for each matrix, the rows and columns represent the selected turning movements of A^v. The matrix in channel 1, turning and matched volumes from AVI data, is represented by $\mathbf{Q}_h \in \mathbb{R}^{|A^v| \times |A^v|}$, where the diagonal element $Q_{a,a,h}$ ($Q \in \mathbb{R}^{|A^v| \times |A^v| \times |H|}$) denotes the number of detected vehicles passing turning movement a during time interval h and the off-diagonal element $Q_{a,a',h}$ denotes the number of matched vehicles from turning movement a to a'. Similar to Q_h, the matrix $Q'_h \in \mathbb{R}^{|A^v| \times |A^v|}$ in channel 2 represents the sampled turning vehicles and matched vehicles from the probe vehicle data. The matrix in the last channel, matched travel time, is represented by $\mathbf{T}_h \in \mathbb{R}^{|A^v| \times |A^v|}$, where only the off-diagonal elements may be nonzero, and $T_{a,a',h}$ ($T \in \mathbb{R}^{|A^v| \times |A^v| \times |H|}$) denotes the average travel time for the matched vehicles from the turning movement a to a' during interval h. Here, it is assumed that the travel time measured by the AVI system is more accurate than that of probe vehicles, because the AVI system can detect more regular vehicles. If the travel time from a to a' is positive, the reverse travel time and the number of matched vehicles are negative.

The stack of these three matrices forms a temporal cell $\Delta_h \in \mathbb{R}^{|A^v| \times |A^v| \times 3}$. Considering the complementary characteristics of AVI and the probe vehicle data in vehicle collection, the matrices $\mathbf{Q}_h$ and $\mathbf{Q}'_h$ are placed in adjacent channels. The travel time obtained from these two data sources is combined in one matrix $\mathbf{T}_h$ to avoid possible inconsistency between them. From the first channel to the last channel of Δ_h, we can see the matched regular vehicles

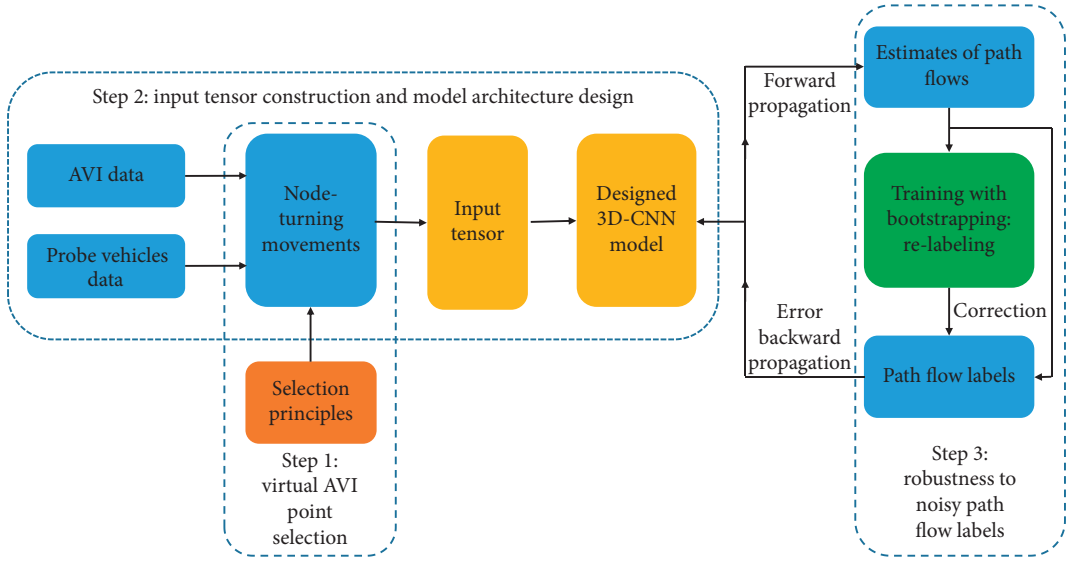


Figure 2: General modeling framework.

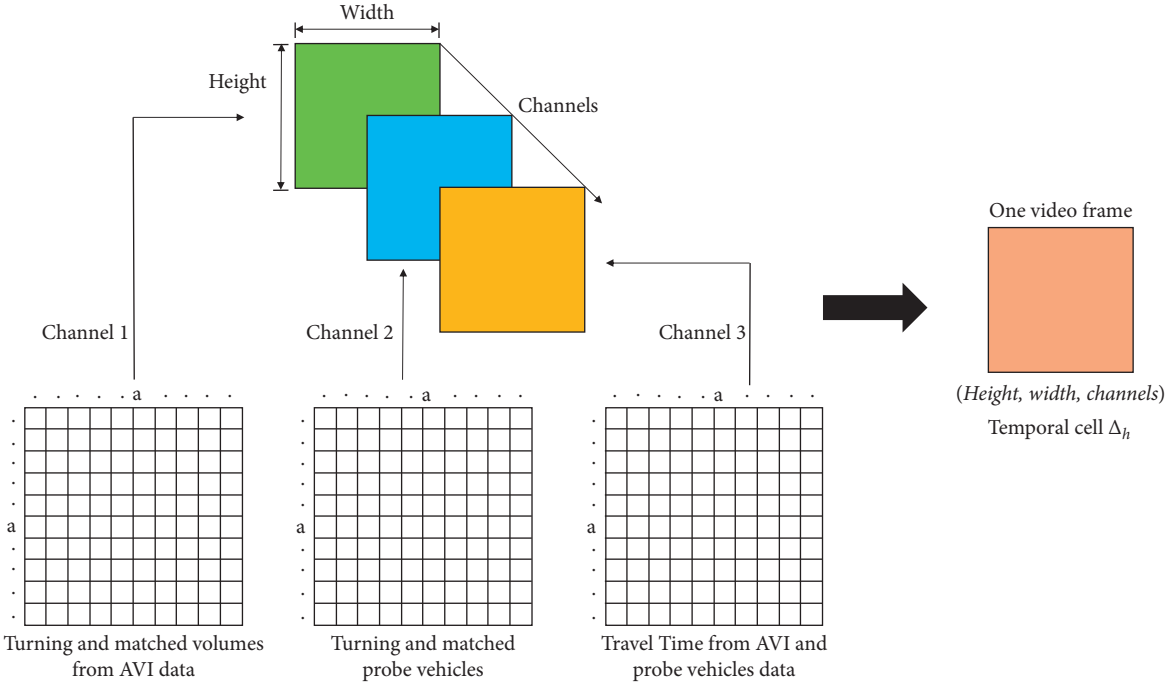


Figure 3: The input tensor for 3D-CNN model.

from the AVI data, the matched probe vehicles, and the corresponding average travel time between any two turning movements in order. The trip information contained in the collected AVI and the probe vehicle data within interval h can be well organized and expressed hierarchically.

3.2. 3D-CNN Architecture Design. For one network, the prior and estimated path flow matrices are represented by $\mathbf{Y} \in \mathbb{R}^{|H|\times|P|}$ and $Y' \in \mathbb{R}^{|H|\times|P|}$. In a dynamic estimation problem, the generated $Y_{h,k}$ vehicles in the origin of path k will be observed in the following time intervals h, h, ..., $h + w'$, and $(w' + 1)$ is the maximum number of time intervals required to travel any path of the network [18]. Hence, for the estimate of one row vector $\mathbf{Y}_{h,:}$, the temporal cells from h to $h + w'$ are needed and can be seen as a sequence of video frames of network operation.

Referring to VGG-Net, which is a powerful two-dimensional CNN model presented by Simonyan and Zisserman [19], a 3D-CNN model is designed and shown in Figure 4. The VGG-Net has a very deep convolutional (conv.) architecture and the model capability is increased as the network becomes deeper, but it

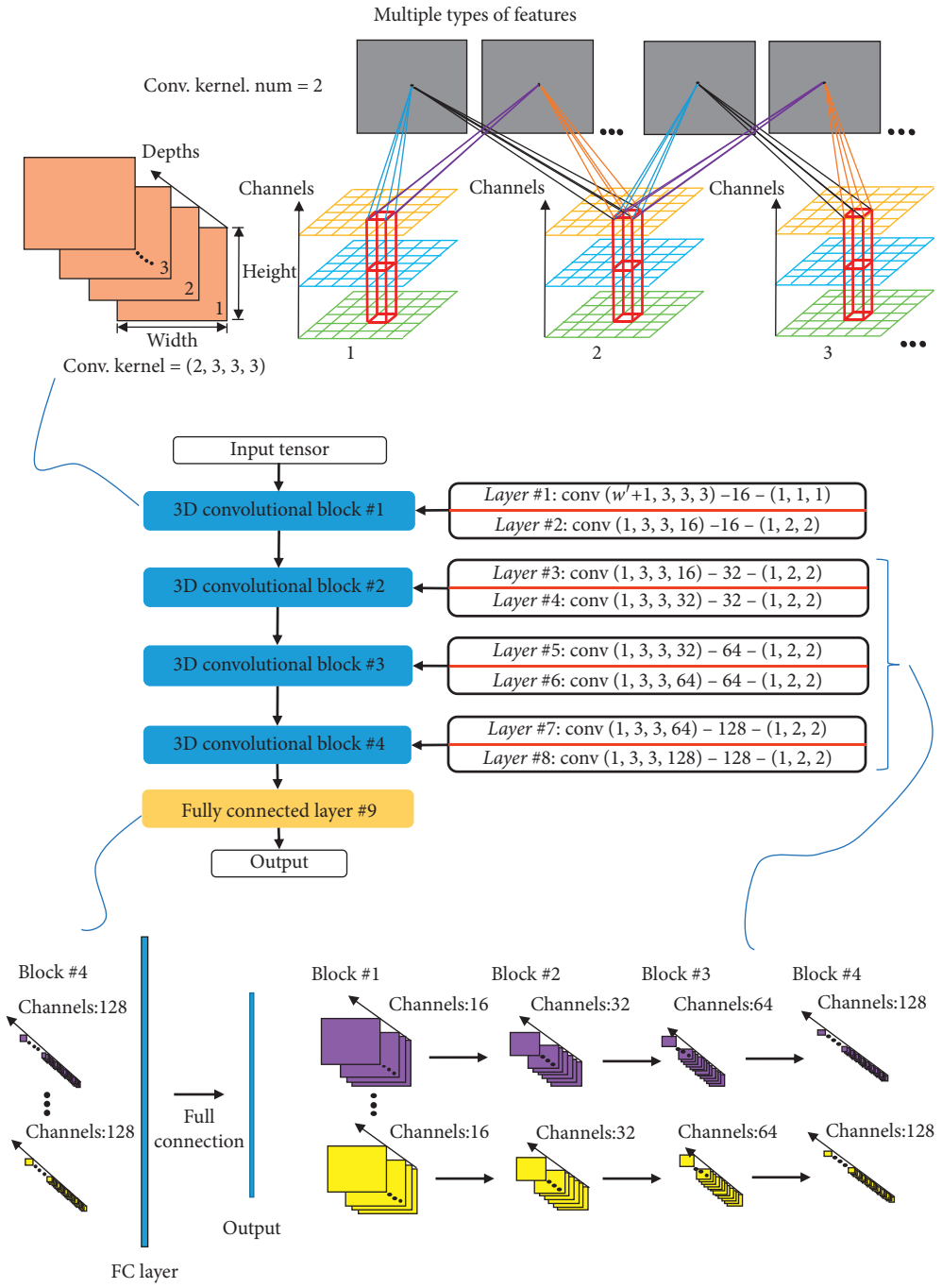


Figure 4: Designed 3D-CNN model.

leads to a heavier computational cost. Considering this tradeoff and experimental analysis, the number of 3D conv. blocks and weight layers (including conv. and fully connected (FC) layers) are reduced to 4 and 9, respectively. Each block has two conv. layers, and the corresponding parameters are denoted as "Conv (kernel size: depth, height, width, channels)–(number of kernels)–(conv. stride: depth, height, width)". Besides the first conv. layer, the conv. stride in the height and width directions, which is (1, 1) in VGG-Net, is changed to (2, 2). As the number of conv. layers decreases, the pooling players used in VGG-Net are completely removed, which may destroy the spatial features of maps.

The input of 3D-CNN containing several temporal cells Δ_h is denoted by $X \in \mathbb{R}^{|H|\times|A^v|\times|A^v|\times 3}$. As shown at the top of Figure 4, the 3D convolution kernel is also a cube and can move not only in height and width but also in depth. The kernel size on height and width (3, 3) is consistent with VGG-Net to extract spatial features. However, on depth, the

kernel size is set as $(w' + 1)$ in the first conv. layer and 1 in the other conv. layers. Combining the moving stride on depth set as 1, it means that every $(w' + 1)$ time interval of X is treated as a group to extract the spatial-temporal features and estimate the path flows in the first interval. In addition, the number of feature maps for each group gradually increases in late conv. block with the rise of convolution kernels. The data-processing procedure in each block can be formulated as follows:

$$\begin{aligned} R^{(|H|-w',\ldots,16)} &= \text{Conv 3 d\_1}\left(X^{(|H|,\ldots,3)}\right), \\ R^{(|H|-w',\ldots,128)} &= \text{Conv 3 d\_4}\left(\text{Conv 3 d\_3}\left(\text{Conv 3 d\_2}\left(R^{(|H|-w',\ldots,16)}\right)\right)\right), \end{aligned} \tag{6}$$

where R is the output tensor of each conv. block.

After hierarchical feature extraction, combination, and transformation, multiple local and spatial-temporal traffic patterns are recognized and finally mapped to prior path flows through a single FC layer.

Step 3. Robustness to Noisy Labels

The key to bootstrapping is the designed judgment and the re-labeling process, which can be placed in the outer loop or incorporated in the loss function [17]. Through re-labeling the samples while training, more accurate labels may lead to a better model, which allows for further label clean-up, and the learner bootstraps itself in this way. An established framework of bootstrapping with the re-labeling process in the outer loop is described in Figure 5.

From Figure 5, you can see that there are two parts (2-1 and 2-2) in the training phase. For the pretraining part (2-1), its existence is not necessary and depends on whether the defined relabeling principle heavily relies on the output of the NN model itself. Taking the multi-class classification task of the NN model as an example, in the work of Liu et al. [20], a confidence policy curve, which is independent of the model output (the probabilities that the input sample belongs to each given class), is defined to determine the selection of training and prediction labels. But for the self-error-correcting CNN model proposed by Wu et al. [17], the label of one training sample would be re-labeled if the probability of the predicted label given by the model is greater than the threshold probability. In this case, the pretraining part is needed and the mini-batch stochastic gradient descent (SGD) algorithm is used. Combining the right part of Figure 5 and ignoring the elements marked in red, you can see there are t_e epochs and for each epoch, m_b samples are first randomly selected. Then the input tensor X^s and label matrix $\mathbf{Y}^s$ of each sample are stacked to form X and $\mathbf{Y}$, respectively. Finally, the SGD can optimize the model parameters based on the batch-size input tensor X and label data$\mathbf{Y}$. Through the pretraining with data sample set S_3 (the size of S_3 is not big but the labels of samples are of higher quality), the model parameters are initialized to enable the model to find the correct gradient descent direction during the next iterative training process.

In this study, the bootstrapping algorithm is used to handle the prior path flow labels with noises. Yang et al. [3] presented two indexes α and ρ, which indirectly measure the systematic and the random variations between the true OD matrix and the prior one. Compared with the random error, the systematic error may have a significant influence on model learning. The labels belong to different volume levels, similar to the images belonging to different classes. α is also used in OD matrix estimation to guarantee that the current flow levels are reproduced on an average [21]. And with the mini-batch SGD algorithm used, the gradient descent direction can still be found among the training samples whose labels contain random errors through iterative training. Hence, α is selected as the re-labeling indicator and considering the dynamic path flow estimation and the traffic counts of turning movements directly obtained from AVI system, the calculation of α within time interval h is changed as follows:

$$\alpha_h = \frac{1}{|A^r|} \sum_{a \in A^r} \frac{\sum_{w=h-w'}^{h} \sum_{k \in P} y_k^w \delta_{ha}^{wk}}{q_a^h}, \tag{7}$$

where q_a^h is the measured traffic counts of turning movement a within interval h; y_k^w is the flow of path k departing the origin in interval w; δ_{ha}^{wk} is defined as the contribution of y_k^w passing the turning movement a during interval h and the calculation can refer to the research of Ashok and Ben-Akiva [18].

The elements marked in red in Figure 5 represent the re-labeling process of part 2-2, which is key to the bootstrapping method. Combining equation (7), the calculation of α includes two steps: (1) estimate the flows of the turning movements (selected in X^s and with AVI detectors installed) based on the prior path flows $\mathbf{Y}^s$ and forward result of NN $\mathbf{Y}'^s$, respectively; (2) compare the estimated turning movement flows and the corresponding measured values in $\mathbf{X}^s$, and calculate the average ratio α. For the $\mathbf{Y}^s$ and $\mathbf{Y}'^s$, if the α'^s is closer to 1 than α^s, the original path flow labels are replaced by the estimated labels. It is noted that to use α, the number of temporal cells Δ_h of X^s must be larger than $(2 \times w' + 1)$. Only in this way, the path flows of more than $(w' + 1)$ time intervals in $\mathbf{Y}'^s$ can be estimated and α of at least one time interval can be calculated. For the α of several time intervals, the corresponding average value can be used in the re-labeling process.

3.3. Case Study. In this section, a realistic urban road network in Qingdao, China, is used to validate the proposed model. Considering the designed 3D-CNN model as the

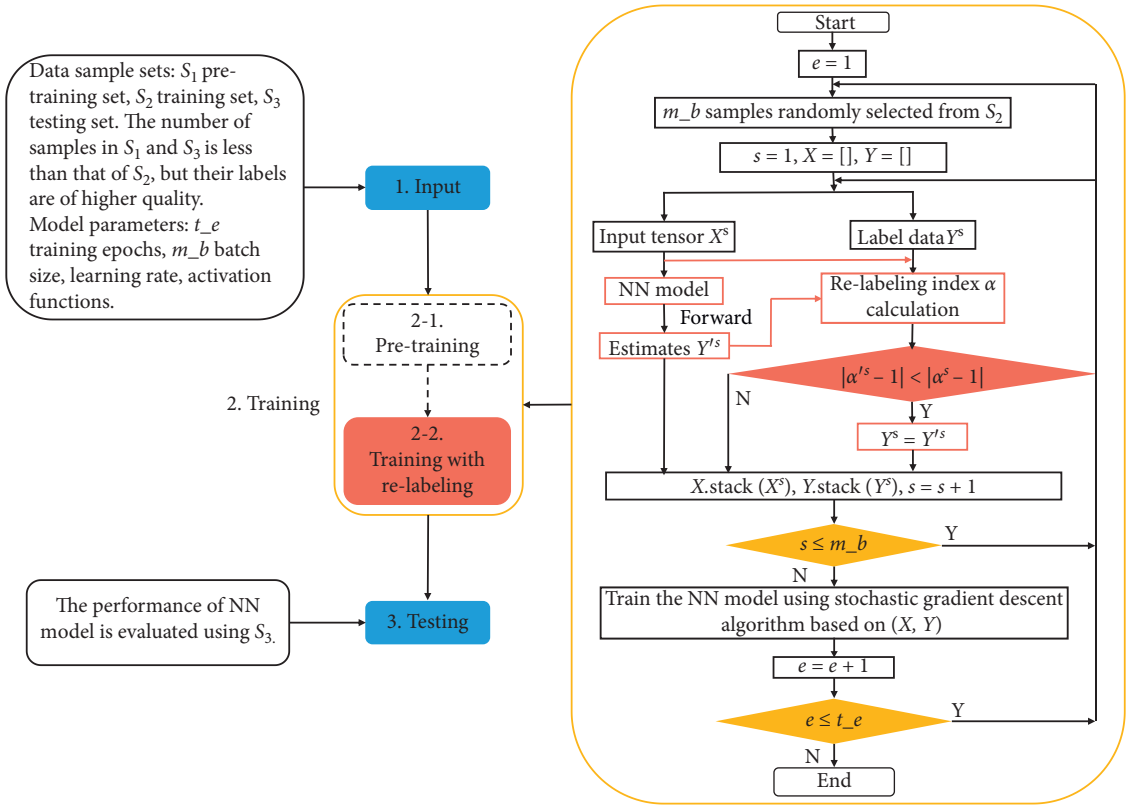


Figure 5: Framework of model training using bootstrapping algorithm.

basis of the entire model, the architecture of deep NN is not separately tested. The method of turning movement selection and the robustness of the model with bootstrapping algorithm to noisy labels are focused.

3.4. Network and Dataset Description. The topology of the validating network is shown in Figure 6. The network contains 297 turning movements and the number of real AVI points is 70, accounting for 31.5% of the cross nodes' turns. To generate the large dataset required for model testing, the study of Tang et al. [13] was referred to and the micro-simulation model of validating the network is first established and calibrated to prepare the basic dataset. The model calibration process commonly includes four steps [22]: (1) ensuring key input data (e.g., digital road network, traffic management and control plan) are accurate; (2) calibrating the traffic demands; (3) correcting the related parameters of the simulation model (e.g., car-following, lane-changing parameters, and the distribution of desired speeds); and (4) selecting the evaluation indexes for validation. Steps 1 and 2 are the base and are completed by using the macro and micro simulation software VISUM and VISSIM together. The VISUM is good at modeling a large network and has the TFlowFuzzy module [23], which can correct the OD and path demand matrix based on the observed traffic data. Except for the road network image with high resolution for network modeling, the AVI data for the real AVI points in validating the network were also collected from 7:30 AM to 8:30 AM from November 25 to December 1, 2016. The average flows of these turning movements were used as observed values in the VISUM and the improved static OD matrix and assigned path flows were be obtained. The established macro-VISUM model can be directly imported into VISSIM to further complete the Steps 1, 3, and 4. Considering the micro-simulation model in this paper mainly used to validate the path flow estimation method, the related operational parameters like the distribution of desired speeds are focused and calibrated in Step 3. The errors between the estimated and the actual turning flows are used in Step 4 to evaluate the accuracy of the simulation model. The calculated average relative error is 6.26% less than 15%, and it can be accepted based on the work of Antoniou et al. [22].

Then, the calibrated dataset during the morning peak period is taken as the basis to generate rich scenarios covering noncongested and more congested periods, by adjusting the input volumes and adding the Gaussian noises. There are six levels of the VISSIM input volumes from −0.3 to 0.2 at 0.1 intervals. And the positive or negative volume level represents the corresponding ratio the basic input volumes will increase or decrease by. Under each level, the Gaussian noises (ranging from 5% to 30%) are superimposed on the input volumes and path choice probabilities eight times. For each time, the simulation model with added noises will run five times with different random seeds. Hence, there are 40 scenarios for each level and the total number of scenarios is 240. The number of paths is 311 and

the market penetration for probe vehicles is set as 10%. The expanded dataset may be not real, but it is valid for model integrity training and testing.

Here, we define the estimation time interval to be 10 minutes and the one hour simulation period of any scenario can be divided into 6 intervals. The longest trip time among all scenarios is 17 minutes and the value of $(w' + 1)$ is thus 2. Considering at least $(2 \times w' + 1)$ temporal cells of input required for bootstrapping, the $X \in \mathbb{R}^{3\times|A^v|\times|A^v|\times 3}$ and the corresponding true path flows $\mathbf{Y}^+ \in \mathbb{R}^{2\times 311}$ can be paired to form a sample, and each scenario can produce four samples. The total number of samples is 960, and the sample size of pretraining, training, and testing sets are, respectively, 120, 720, and 120 samples.

3.5. Evaluation Metrics and Loss Function. Four error metrics are used for our evaluation: mean absolute error (MAE), relative MAE (%), root mean square error (RMSE), and relative RMSE (%). For m pairs of the true and estimated path flows ($\mathbf{Y}^+ \in \mathbb{R}^{|H|\times|P|}$, $Y' \in \mathbb{R}^{|H|\times|P|}$), the units of MAE and RMSE are veh/10 min, and the calculation formulas are as follows:

$$\begin{aligned}
\mathrm{RMSE} &= \sqrt{\frac{\sum_{s=1}^{m}\sum_{h=1}^{|H|}\sum_{k=1}^{|P|}\left|\mathbf{Y}_{h,k}^{+s}-\mathbf{Y}_{h,k}^{'s}\right|^2}{m\times|H|\times|P|}},\\
\mathrm{RMSE}(\%) &= \frac{\mathrm{RMSE}}{\left(\sum_{s=1}^{m}\sum_{h=1}^{|H|}\sum_{k=1}^{|P|}\mathbf{Y}_{h,k}^{+s}/(m\times|H|\times|P|)\right)},\\
\mathrm{MAE} &= \frac{\sum_{s=1}^{m}\sum_{h=1}^{|H|}\sum_{k=1}^{|P|}\left|\mathbf{Y}_{h,k}^{+s}-\mathbf{Y}_{h,k}^{'s}\right|}{m\times|H|\times|P|},\\
\mathrm{MAE}(\%) &= \frac{\sum_{s=1}^{m}\sum_{h=1}^{|H|}\sum_{k=1}^{|P|}\left|\mathbf{Y}_{h,k}^{+s}-\mathbf{Y}_{h,k}^{'s}\right|}{\sum_{s=1}^{m}\sum_{h=1}^{|H|}\sum_{k=1}^{|P|}\mathbf{Y}_{h,k}^{+s}}.
\end{aligned} \tag{8}$$

For a batch including m_b samples of which each is with a prior path flow $\mathbf{Y} \in \mathbb{R}^{|H|\times|P|}$, the mean squared error (MSE) is used as a loss function and the calculation is as follows:

$$\mathrm{MSE} = \frac{\sum_{s=1}^{m\_b}\sum_{h=1}^{|H|}\sum_{k=1}^{|P|}\left(\mathbf{Y}_{h,k}^{s}-\mathbf{Y}_{h,k}^{'s}\right)^2}{m\_b}, \tag{9}$$

3.5.1. Test 1. Turning Movement Selection. For the target network, 110 virtual AVI points, accounting for 37% of all the turning movements, are selected by solving the binary programming model. The number of selected turning movements belonging to cross and noncross nodes is 78 and 32, respectively. Thirt-seven of the virtual AVI points also belong to A^r and the overlap ratio is 33.6%. The selection of noncross nodes' turns can reduce the overall overlap ratio. To completely reflect the dependency of paths and the passed points, the residual 43 turning movements of noncross nodes are finally added to the solution set of the programming model, as shown in Figure 7.

To evaluate the efficiency of the proposed method, two sets of virtual AVI points are defined as follows:

(1) A1: the 297 turning movements of network are all selected

(2) A2: the 153 turning movements selected by the proposed method

The designed 3D-CNN model is used and only the model input is changed according to different sets of turning movements. Hereafter, the 3D-CNN models with A1 and A2 sets are denoted as Model-A1 and Model-A2. The pretraining set is added to the training set and the total number of training samples is 840. During model training, the major hyper parameters, including the training epochs, learning rate, and batch size, were determined based on a grid search experiment. Here, the best combination is used for general evaluation. The number of training epochs, learning rate, and batch size is 5000, 1e-6, and 80, respectively. The train losses varying with epochs for Model-A1 and Model-A2 are shown in Figure 8. The corresponding training durations are marked next to the lines.

Notably, the A2 set is the subset of A1; therefore, the input tensor using A1 set naturally provides much more information. It means that after the same number of training epochs, the Model-A1 can extract more abstract features associated with the labels from the input tensor, which underlies the phenomenon that the train loss of Model-A1 drops faster, as presented in Figure 8. However, because of the larger size of the input tensor using A1 the set, more time is needed to complete the convolutional calculations of conv. layers. Model-A1 and Model-A2 ran in the same environment. The training time needed by Model-A1 is 111 min, which is approximately 4 times that of Model-A2. After 2250 epochs, the drop speed of Model-A2's train loss is higher than that of Model-A1. The final train loss of Model-A2 is 7.1% lower than that of Model-A1.

For the four evaluation metrics (RMSE, RMSE%, MAE, MAE%), the testing results of Model-A1 and Model-A2 on the entire testing set are (3.65 veh/10 min, 45.79%, 1.92 veh/10 min, 24.04%) and (3.58 veh/10 min, 44.93%, 1.88 veh/10 min, 23.55%), respectively. It can be seen that the four metrices of Model-A2 are improved compared with those of Model-A1. The results reveal that the turning movement selection method can identify the critical turning movements for path flow estimation and largely reduce the computation cost.

3.5.2. Test 2: Model Learning on Noisy Labels. According to the label noise analysis in Part 3 of Methodology, the path flow labels may have systematic and random errors. The prior path flow $\mathbf{Y}_{h,k}$ with noises can be generated by

$$\mathbf{Y}_{h,k} = \mathbf{Y}_{h,k}^{+} \times (1.0 - \eta) \times \left(1.0 - c_{vf}\sigma_{hk}\right), \tag{10}$$

where $\mathbf{Y}_{h,k}^{+}$ is the true path flow of path k during time interval h, σ_{hk} is the independent normal random variable of $N(0, 1)$, η is the bias, and c_{vf} reflects the magnitude of random variation.

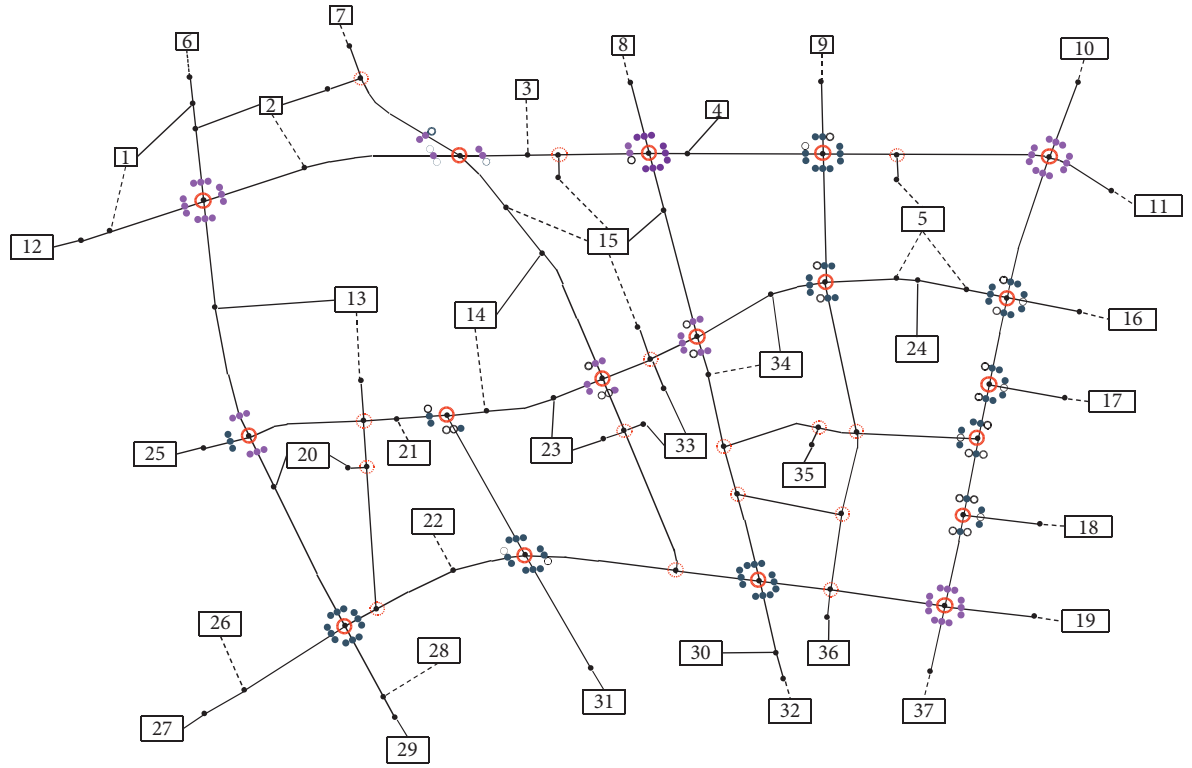


Figure 6: The topology of network and distribution of real AVI points.

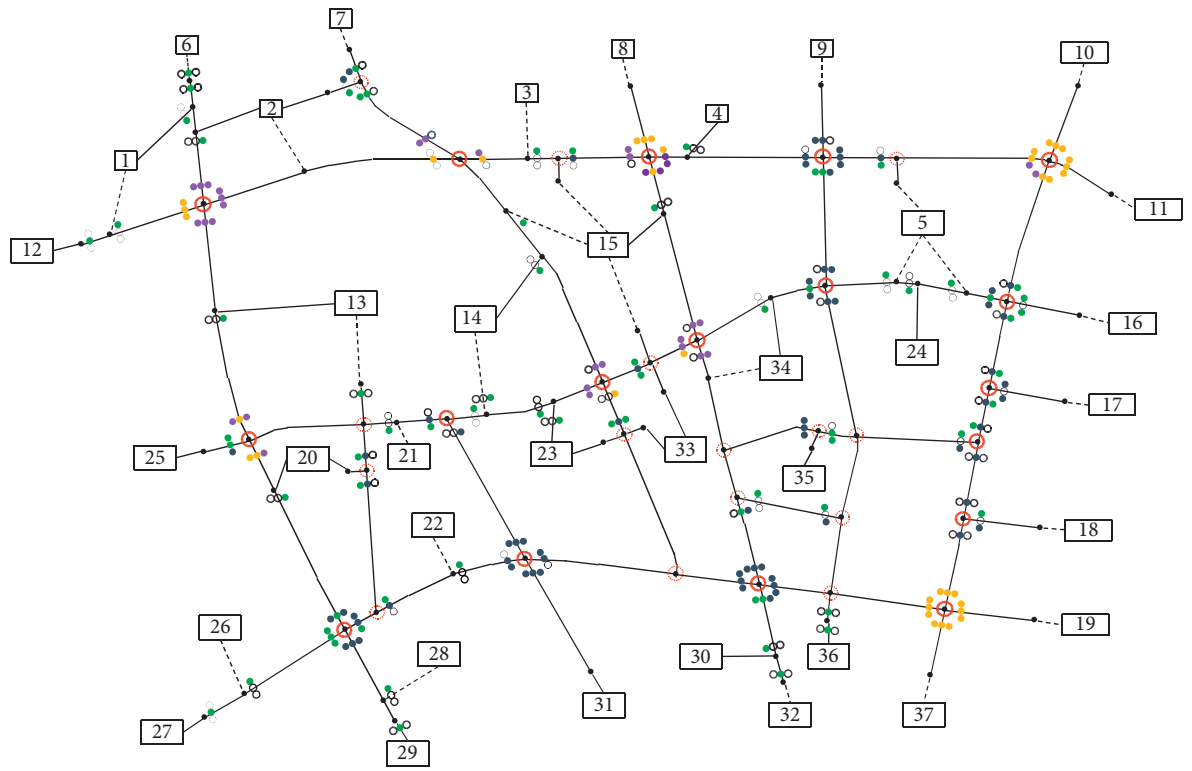


Figure 7: The distribution of optimally selected virtual AVI points.

Based on the results of Test 1, Model-A2 is used for validation here. Without using the bootstrapping training algorithm, Model-A2 is trained with training samples from pretraining and training sets whose labels are noisy. The number of training epochs and learning rate are kept the same, while the batch size is changed to 120. Table 1 presents

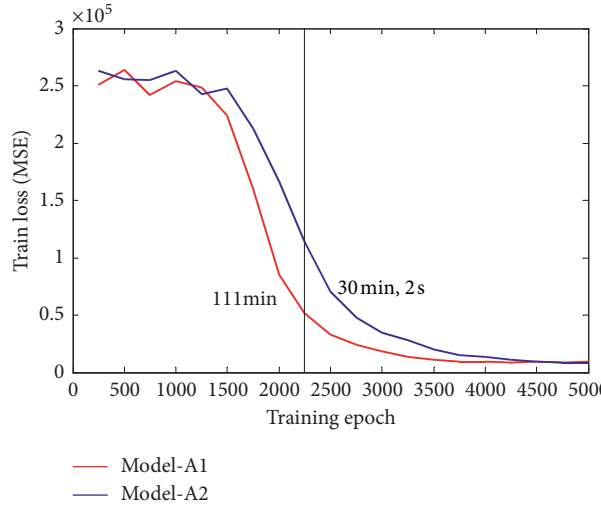


Figure 8: Train loss variation with increased train epochs.

Table 1: Performance of Model-A2 trained by samples with noisy labels.

η	0			0.2			0.4		
c_{vf}	0.2	0.4	0.6	0.2	0.4	0.6	0.2	0.4	0.6
RMSE veh/10 min	3.58	3.56	3.62	5.49	5.45	5.26	8.87	8.81	8.72
RMSE %	44.90	44.62	45.39	68.84	68.36	65.93	111.28	110.58	109.33
MAE veh/10 min	1.88	1.87	1.89	2.39	2.37	2.32	3.56	3.56	3.52
MAE %	23.54	23.48	23.76	29.95	29.73	29	44.69	44.6	44.12

the model performance in the testing set under different combinations of η and c_{vf}.

Table 1 reveals that the labels with systematic errors have a great influence on the model performance, while the model is generally insensitive to random errors. All four metrics increase significantly with an increase in η. However, under the same value of η, the variations of the four metrics are slight when c_{vf} increases from 0.2 to 0.6. It meets our expectation and the (stochastic) gradient descent algorithm is immune to training samples with random errors. However, it should be noted that owing to the over-parameterized NN, the CNN model has the capacity to (over) fit a subset of labels with random noises. The over-fitting should be avoided.

Considering the existence of the pretraining phase, the test for bootstrapping algorithm is divided into two parts. The model used in Part 1 is the pretrained Model-A2, while in Part 2, the pretraining phase is skipped. For the pre-training set, the labels of samples are just added to the random errors with the c_{vf} set as 0.2. Based on the model performance measurements of Table 1, the η and c_{vf} are, respectively, set as 0.4 and 0.2 to generate noises for the labels of samples in the training set. As for the percentage of noisy labels in the training set, 0.1, 0.3, 0.5, 0.7, and 0.9 are used.

In Part 1 and Part 2, the training sets with different percentages of noisy labels are used to train Model-A2 and the corresponding model performances on the testing set are listed in Table 2.

In Table 2, each parenthesis contains two metric values separated by a forward slash and the difference lies in whether the bootstrapping algorithm is used for Model-A2's training on the training set. For the pretrained Model-A2 used in Part 1, the corresponding four metrics on the testing set are 7.03 veh/10 min, 88.14%, 2.85 veh/10 min, and 35.7%. From the first values of parentheses in Part 1, it can be observed that the trained model performs gradually worse as the percentage of noisy labels in the training set increases. But for the Model-A2 trained on the training set with the bootstrapping algorithm, it performs better and the percentage of noisy labels has less influence on model performance. The trends of the values in Part 2 are similar to those in Part 1, but the values increase overall. It is not difficult to understand that the weight parameters are randomly initialized and the convergence speed is slow without the pretraining phase.

Take the label noise ratio of 70% as an example. For the models trained with bootstrapping in Part 1 and Part 2, the train loss, sample loss distribution within one epoch, and label replaced ratios are shown in Figure 9.

Figures 9(a) and 9(b) indicate that the loss on the testing set of Part 1 can converge to a good point within the total 5000 iterations, while that of Part 2 keeps decreasing. It reveals that the pretraining phase can effectively accelerate the convergence of the model. Owing to the pretraining phase, the label replacement ratio for training samples with noisy labels in Part 1 is nearly 100% within the entire training process, as shown in Figure 9(e). But in Figure 9(f),

Table 2: Test of bootstrapping algorithm for different percentages of noisy labels.

	Percentage	10%	30%	50%	70%	90%
Part 1 (with pretraining)	RMSE	(3.63/**3.55**)	(4.29/**3.55**)	(5.37/**3.56**)	(6.57/**3.58**)	(8.08/**4.45**)
	RMSE%	(45.56/**44.50**)	(53.84/**44.50**)	(67.35/**44.60**)	(82.37/**44.88**)	(101.35/**55.83**)
	MAE	(1.89/1.89)	(2.05/**1.89**)	(2.36/**1.89**)	(2.75/**1.91**)	(3.27/**2.06**)
	MAE%	(23.73/**23.72**)	(25.67/**23.72**)	(29.64/**23.75**)	(34.50/**23.94**)	(41.07/**25.89**)
Part 2 (without pretraining)	RMSE	(3.71/3.74)	(4.36/**3.8**)	(5.43/**3.99**)	(6.57/**4.48**)	(8.12/**7.79**)
	RMSE%	(46.60/46.87)	(54.72/**47.69**)	(68.18/**50.09**)	(82.42/**56.18**)	(101.89/**97.68**)
	MAE	(1.90/1.93)	(2.05/**1.94**)	(2.35/**1.97**)	(2.72/**2.04**)	(3.28/**2.68**)
	MAE%	(23.86/24.26)	(25.7/**24.34**)	(29.52/**24.8**)	(34.14/**25.65**)	(41.14/**33.66**)

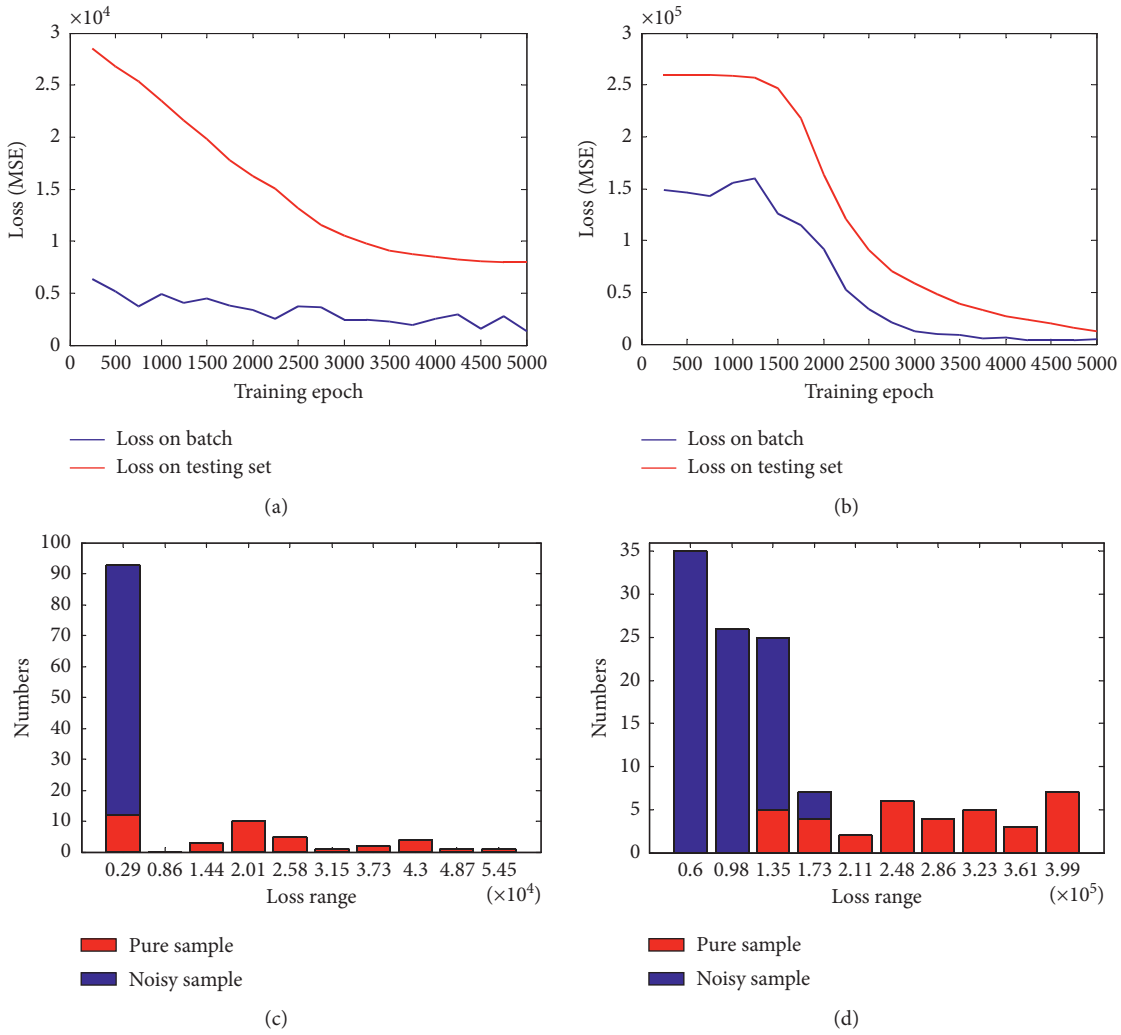


Figure 9: Continued.

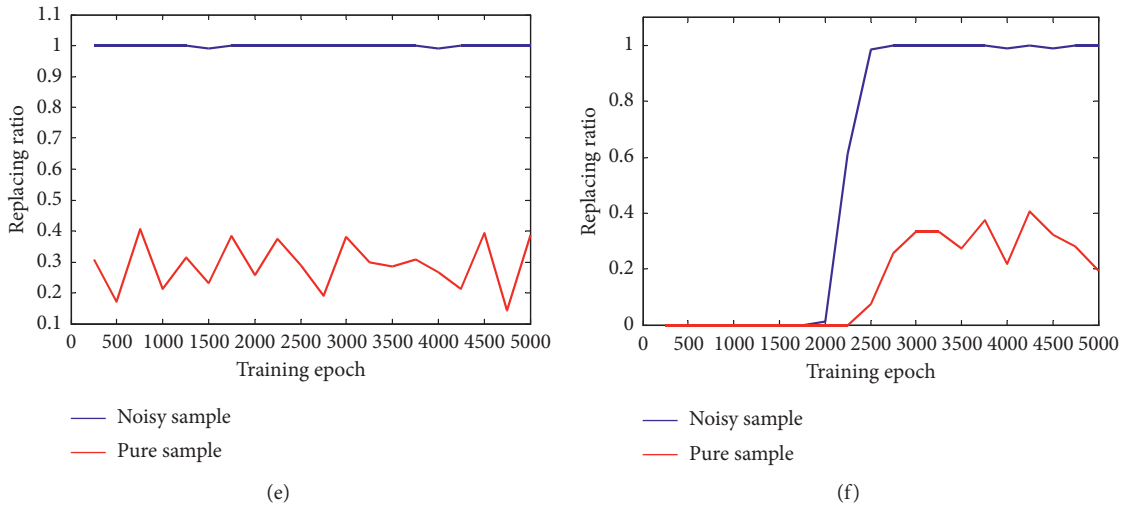


Figure 9: Train losses, sample loss distributions, and label replacing ratios in Part 1 and Part 2. (a, e) Part 1, (b, f) Part 2, (c) No. 250 epoch of Part 1, and (d) No. 250 epoch of Part 2.

during the initial iterations, the label replacement ratio for noisy samples is nearly zero and it increases to 100% after 2000 epochs. For the sample loss distribution at the epoch No. 250 shown in Figure 9(d), the training samples with noisy labels are not re-labeled and occupy most of the training batch; therefore, they become the focus of regression and the losses of pure samples are naturally higher than those of Part 1. Figure 9(c) can be used to explain why the bootstrapping algorithm can reduce the influence of noisy labels. After resampling the samples and relabeling the corresponding noisy labels, in the next epoch, the losses for noisy samples are initially by zero and the start point of gradient descent is placed in the pure samples. The fluctuations of loss in the batch in Figures 9(a) and 9(b) are reasonable, because the percentages of pure and noisy samples are different in every resampled sample. With the improvement in model estimation accuracy, the label replacement ratio for pure samples is between 0.2 and 0.4.

4. Conclusions

In this study, to make full use of the rich and complementary individuals' trip information provided by AVI and probe vehicle data, and to avoid intractable mathematical program solution, the dynamic path flow estimation is treated as a data-driven feature learning problem and these two data sources are fused at the data level. A 3D convolution-based deep NN is designed, and the turning movements at network nodes are used to represent the AVI and the probe vehicle observations in the input tensor. The principles for selecting the key turning movements and a corresponding programming model are also proposed. To make the NN robust to the noisy path flow labels during model training, a self-correcting algorithm named bootstrapping, which can use the model outputs to correct the noisy labels based on the defined re-labeling principle, is established.

In the case study, a realistic urban road network was used and the corresponding microscopic simulation model was built and calibrated by VISSIM to generate large data samples. Two distinctive tests, numbered 1 and 2, were carried out to validate the turning movement selection and bootstrapping methods. In Test 1, the designed 3D-CNN model with the input tensor constructed by the selected turning movements achieved a MAE of 1.88 veh/10 min, and compared with the model with all the turning movements used in input tensor, the computational time and estimation accuracy were both improved. This reveals that the designed architecture of 3D-CNN model presents satisfactory performance, and the virtual AVI point selection method can retain the key information for each path and remove redundant information. In Test 2, the path flow labels were artificially superimposed with systematic and random errors to test the model robustness. Without overfitting, the NN model trained with the gradient descent algorithm is almost immune to the labels with random errors. Systematic errors were mainly considered and the bootstrapping can make the model more robust to different percentages of labels with errors. The pretraining phase is not necessary in this study, but it can help improve the convergence speed and estimation accuracy. The defined re-labeling criteria are important and can limit the final estimation accuracy.

Despite the promising results, the study has certain limitations and further works could be focused on three aspects to extend the topic: firstly, the dynamic path choice in congested networks should be considered in the virtual AVI points selection model; secondly, the influence of various modes of market penetration of probe vehicles and feature noises (e.g., the missing detection phenomenon in AVI system) should be further investigated; last but not least, a real-world validation with filed AVI data and probe vehicle trajectories has to be conducted.

Data Availability

The AVI and probe vehicle trajectory data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors thank the National Key Research and Development Program of China (2019YFB1600202) for supporting this research.

References

[1] Y. Nie and D.-H. Lee, "Uncoupled method for equilibrium-based linear path flow estimator for origin-destination trip matrices," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1783, no. 1, pp. 72–79, 2002.

[2] E. Castillo, J. M. Menéndez, and P. Jiménez, "Trip matrix and path flow reconstruction and estimation based on plate scanning and link observations," *Transportation Research Part B: Methodological*, vol. 42, no. 5, pp. 455–481, 2008.

[3] H. Yang, T. Sasaki, Y. Iida, and Y. Asakura, "Estimation of origin-destination matrices from link traffic counts on congested networks," *Transportation Research Part B: Methodological*, vol. 26, no. 6, pp. 417–434, 1992.

[4] X. Zhou and H. S. Mahmassani, "Dynamic origin-destination demand estimation using automatic vehicle identification data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 105–114, 2006.

[5] P. Krishnakumari, H. van Lint, T. Djukic, and O. Cats, "A data driven method for OD matrix estimation," *Transportation Research Part C: Emerging Technologies*, vol. 113, pp. 38–56, 2020.

[6] M. P. Dixon and L. R. Rilett, "Real-time OD estimation using automatic vehicle identification and traffic count data," *Computer-Aided Civil and Infrastructure Engineering*, vol. 17, no. 1, pp. 7–21, 2002.

[7] T. Huang, Y. Ma, and Z. T. Qin, "Origin-destination flow prediction with vehicle trajectory data and semi-supervised recurrent neural network," in *Proceedings of the 2019 IEEE International Conference on Big Data (Big Data)*, IEEE, Los Angeles, CA, USA, December 2019.

[8] J. Yang and J. Sun, "Vehicle path reconstruction using automatic vehicle identification data: an integrated particle filter and path flow estimator," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 107–126, 2015.

[9] X. Yang, Y. Lu, and W. Hao, "Origin-destination estimation using probe vehicle trajectory and link counts," *Journal of Advanced Transportation*, vol. 2017, pp. 1–18, 2017.

[10] R. Ásmundsdóttir, *Dynamic OD Matrix Estimation Using Floating Car Data*, Delft University of Technology, Delft, The Netherlands, 2008.

[11] W. Wong, S. C. Wong, and H. X. Liu, "Bootstrap standard error estimations of nonlinear transport models based on linearly projected data," *Transportmetrica A: Transport Science*, vol. 15, no. 2, pp. 602–630, 2019.

[12] C. Osorio, "Dynamic origin-destination matrix calibration for large-scale network simulators," *Transportation Research Part C: Emerging Technologies*, vol. 98, pp. 186–206, 2019.

[13] K. Tang, Y. Cao, C. Chen et al., "Dynamic origin-destination flow estimation using automatic vehicle identification data: a 3D convolutional neural network approach," *Computer-Aided Civil and Infrastructure Engineering*, vol. 2020, pp. 1–17, 2020.

[14] B. Frénay and M. Verleysen, "Classification in the presence of label noise: a survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 845–869, 2013.

[15] H. Alibabai and H. S. Mahmassani, "Dynamic origin-destination demand estimation using turning movement counts," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2085, no. 1, pp. 39–48, 2008.

[16] S. Reed, H. Lee, and D. Anguelov, "Training deep neural networks on noisy labels with bootstrapping," 2014, https://arxiv.org/abs/1412.6596.

[17] X. Wu, R. He, Z. Sun, and T. Tan, "A light CNN for deep face representation with noisy labels," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2884–2896, 2018.

[18] K. Ashok and M. E. Ben-Akiva, "Estimation and prediction of time-dependent origin-destination flows with a stochastic mapping to path flows and link flows," *Transportation Science*, vol. 36, no. 2, pp. 184–198, 2002.

[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[20] X. Liu, S. Li, and M. Kan, "Self-error-correcting convolutional neural network for learning with noisy labels," in *Proceedings of the 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, IEEE, Washington, DC, USA, June 2017.

[21] L. G. Willumsen, "Estimating time-dependent trip matrices from traffic counts," in *Proceedings of the 9th International Symposium on Transportation and Traffic Theory*, VNU Science Press Utrecht, Delft, The Netherlands, July 1984.

[22] C. Antoniou, J. Barcelò, and M. Brackstone, "Traffic simulation: case for guidelines," 2014.

[23] A. G. PTV, *How to Work with TFlow Fuzzy*, Official Manual, PTV Vision Visum, Karlsruhe, Germany, 2008.

Hindawi
Journal of Advanced Transportation
Volume 2020, Article ID 8850541, 11 pages
https://doi.org/10.1155/2020/8850541

Research Article

CPT Model-Based Prediction of the Temporal and Spatial Distributions of Passenger Flow for Urban Rail Transit under Emergency Conditions

Wei Li, Min Zhou, and Hairong Dong

State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing, China

Correspondence should be addressed to Min Zhou; zhoumin@bjtu.edu.cn

Received 4 September 2020; Revised 10 November 2020; Accepted 4 December 2020; Published 15 December 2020

Academic Editor: Wen LIU

Emergencies have a significant impact on the passenger flow of urban rail transit. It is of great practical significance to accurately predict the urban rail transit passenger flow and carry out research on its temporal and spatial distributions under emergency conditions. Urban rail transit operating units currently use video surveillance information mainly to process emergencies and rarely use computer vision technology to analyze passenger flow information collected. Accordingly, this paper proposes a passenger flow-based temporal and spatial distribution model for urban rail transit emergencies based on the CPT. First, this paper clarifies the categories and classification of urban rail transit emergencies, analyzes the factors affecting passenger route selection, and establishes a generalized travel cost model for passengers under emergencies. Second, this paper establishes a passenger route choice behavior model for urban rail transit based on the cumulative prospect theory. Finally, taking Beijing as an example, this paper analyzes passenger travel behavior under emergencies based on multiple logistic regression models and analyzes the impact of emergencies on rail transit travel behavior. The research results show that the cumulative prospect theory can better describe the route choice behavior of rail transit passengers under emergencies than the existing models, and this model is of great significance for handling urban rail transit emergencies. The model proposed in this paper can provide a theoretical basis for the government and relevant departments to formulate traffic management measures.

1. Introduction

Urban rail transit, which is intensifying, is characterized by fast speed, large capacity, high efficiency, and low energy consumption and has become the main form of urban public transportation. With the development of urban rail transit networks, urban rail transit has become the first choice for commuter passengers, with concentrated passenger flows during the morning and evening travel peaks. The stability and reliability of rail transit operation are of great significance to the stable development of urban transportation. However, with an increasing number of lines, the strictness of equipment standards, the aging of equipment and other reasons, and the types of emergencies that occur are gradually diversifying, their frequency of occurrence is increasing, and the spatial scope is expanding [1]. Once an emergency occurs, the normal operation of the train will be affected. If emergencies are not dealt with in a timely manner, the transportation capacity of urban rail transit will be reduced, and traffic paralysis may occur. In the case of urban rail transit network operation, research is carried out on the real-time prediction and early warning of the temporal and spatial distributions of passenger flow in the case of rail transit emergencies to accurately grasp the temporal and spatial distributions of passenger flow in the rail transit network. The impact of the incident, the improvement of real-time passenger flow monitoring capabilities and passenger travel information service levels, the effective organization of subway passenger flow, and the strengthening of subway passenger flow control are all of great significance.

To predict the temporal and spatial distributions of urban rail transit under normal operating conditions and to

provide an early warning of large passenger flows, many studies have been carried out both at home and abroad [2–5]. At present, forecasting the temporal and spatial distributions of passenger flow is mainly aimed at high-density areas such as major event venues and transportation hubs. When investigating the route choice behavior of urban rail transit passengers under normal operating conditions, researchers generally combine historical data and traditional model methods. However, it is difficult to predict the selection behavior of urban rail transit passengers under emergency conditions, and it is difficult to conduct in-depth research with traditional models and methods. Research on rail transit passenger flow under emergencies mostly focuses on the prediction, propagation, evacuation, etc., of emergent passenger flow [6–13]. Current research mostly analyzes the impact of sudden passenger flow on urban rail operation from the perspective of train operation and passenger transportation organization, whereas relatively few studies have been conducted on the prediction and early warning of the temporal and spatial distributions of passenger flow in rail transit emergencies.

Preston et al. [14] analyzed the impact of train delays on passenger travel. Barron et al. [15] and Pnevmatikou et al. [16] considered the characteristics of passenger travel choice behavior under emergency situations. Tsuchiya et al. [17] developed a passenger support system to inform passengers of the best route to their destination in the event of an emergency. Sun et al. [18] established a model to evaluate the impacts of urban rail transit interruptions on travel times and delays based on a Bayesian method. Hong et al. [19] established a model to evaluate local interruptions in urban rail transit networks. Li and Liang [20] analyzed the impact of operating interruptions and proposed a quantitative calculation method for the scale of affected passengers. Huang et al. [21] studied the route selection of urban rail transit passengers under conditions of bounded rationality. Wang and Wu [22] analyzed the impact of emergencies on rail transit travel behavior. Wang [23], Luo [24], Qiao [25], and Wu [26] studied methods to calculate the affected passenger flows of basically unaffected passengers, bypass passengers, and passengers who are unable to reach their destination. Yu [27] proposed a passenger flow assignment method under interruption based on the MNL model. Liu [28] established a mathematical model to estimate the affected passenger flow under the interruption of urban rail transit. Xu et al. [29] established a passenger travel path selection model under urban rail transit emergencies based on the normal distribution probability model. Wang et al. [30] built an emergency logistics path selection model based on CPT, and the results showed that the parameter setting of the model has an important impact on the path selection model. Wang et al. [31] established a cumulative prospect model and concluded that travelers' attitudes toward risk are different under different decision-making behaviors.

In this paper, on the basis of defining the scope of emergency research, with the help of AFC data, historical emergency information, and questionnaire survey data, this paper uses historical emergency passenger selection behavior information to carry out data mining. At the same time, considering the limited rationality of passengers, based on the path selection model of cumulative prospect theory, this paper studies the spatial and temporal distributions of urban rail transit passenger flow under emergencies, carries out passenger flow guidance information release and early-warning research, and verifies the effectiveness of the technology combined with specific cases to provide a reference for passenger flow organization and evacuation.

2. Establishment of a Generalized Travel Cost Model

2.1. Definition and Classification of Emergencies. An urban rail transit emergency refers to an event that occurs suddenly within the operating scope of urban rail transit that may damage the personal property and health of passengers or employees and requires the urban rail transit operation unit to make responsible decisions to minimize losses.

Different types of urban rail transit emergencies have different impacts on train operation. Passengers will choose different routes according to their own conditions, thus forming different temporal and spatial distributions of passenger flow in the urban rail transit network. The scope of this study is defined as follows: peak hours in the morning and evening under the occurrence of natural disasters, equipment failure, passenger transport organization and management scenarios, and other sudden, temporary large passenger flow events. In this paper, the types of urban rail transit emergencies are divided into three categories, as shown in Table 1.

2.2. Establishment of a Generalized Travel Cost Model

2.2.1. Analysis of Factors Affecting the Passenger Route Choice. In the urban rail transit network, when passengers choose a route, they are not only affected by individual subjective factors, such as individual comfort sensitivity requirements and familiarity with rail transit networks, but also by objective factors such as travel time, cost, and distance. The final passenger's route choice is a result of a comprehensive consideration. When any one attribute changes, the path selection may change. The influencing factors of passenger route selection are shown in Table 2.

2.2.2. Establishment of a Generalized Travel Cost Model. The urban rail transit network path refers to the connecting path between any two stations in a transportation network. The generalized passenger travel path includes the inbound channel from the inbound gate to the platform, the starting station platform, the section, the transfer channel of the transfer station, the terminating station, the outbound channel to the outbound gate, and the paths between all connections. The generalized travel cost of passengers refers to the total cost of travel time and money for passengers using rail transit, reflecting the comprehensive cost of passengers choosing a certain route.

This paper establishes a generalized travel cost model based on different weights of influencing factors on

Table 1: Types of urban rail transit emergencies.

Types	Examples
Natural disasters	Heavy rain, blizzard, strong wind, thunder and lightning, heavy fog, and low temperature
Equipment failure	Signal failure, power supply failure, line failure, and vehicle failure
Passenger transport organization and management	People caught in the door, passengers falling off the platform, passenger conflicts, terrorist incidents, and transfer channels closed

Table 2: Influencing factors affecting the passenger route choice.

Type	Name	Description
Subjective factors	Comfort	Passengers will choose a more comfortable route to travel.
	Network familiarity	Passengers who are more familiar with the network will trust their own experience information more and choose new routes to travel. Passengers who are not familiar with the network will only choose the more familiar route to travel.
Objective factors	Time	Passengers will choose a route with a short travel time.
	Cost	Passengers will choose a less expensive route to travel.
	Distance	Passengers will choose a shorter route to travel.

passengers. The urban rail transit network route travel time is the main judgment parameter for route searching in the simulation model. To search for the path set between any two stations in the urban rail transit network, this paper establishes a generalized passenger travel cost model based on time. In an emergency, the model will dynamically update the interval of the period and the impedance on the node before each route search or passenger flow distribution.

$$C_k^{OD} = \sum_k^{o \longrightarrow d} C_{\text{block}} * \gamma + \sum_k^{o \longrightarrow d} C_{\text{node}} + \sum_k^{o \longrightarrow d} C_{\text{transfer}} + \sum_k^{o \longrightarrow d} C_{\text{delay}}. \tag{1}$$

In formula (1), C_k^{OD} is the comprehensive impedance cost function of the k-th path between the OD pair; C_{block} is the impedance cost of all sections of the path; C_{node} is the node impedance cost of all intermediate stations; C_{transfer} is the node impedance cost of all transfer stations; γ is the congestion impedance function of the road section passing through; and C_{delay} is the revised cost function under emergencies.

Each impedance fee can be expressed as follows:

(1) Section impedance cost C_{block}:

$$C_{\text{block}} = t_{\text{block}} * c_{\text{block}}. \tag{2}$$

In formula (2), t_{block} is the travel time of the section, and c_{block} is the travel cost per unit time, RMB/s.

(2) Node impedance cost of the intermediate station C_{node}:

$$C_{\text{node}} = t_{\text{node}} * c_{\text{node}}. \tag{3}$$

In formula (3), t_{node} is the stop time of the intermediate station, and c_{node} is the stop cost per unit time, RMB/s.

(3) Node impedance cost of the transfer station C_{transfer}:

$$C_{\text{transfer}} = \left(t_{\text{transfer}} + 0.5 * E(h)\right) * f(\tau, p) * c_{\text{ph}}. \tag{4}$$

In formula (4), t_{transfer} is the travel time of the transfer station; $E(h)$ is the expected time between rail transit arrivals; $f(\tau, p)$ is the magnification penalty function on the crowded people and the number of transfers; τ is the transfer penalty coefficient; p is the number of transfers; and c_{ph} is the transfer fee for passengers at transfer stations per unit time, RMB/s.

(4) Congestion impedance function γ:

$$\gamma = \begin{Bmatrix} 1; & \omega < \omega_0 \\ 1 + \phi_1(\rho) * (\omega - \omega_0), & \omega_0 < \omega < \omega_1; \\ 1 + \phi_1(\rho) * (\omega_1 - \omega_0) + \phi_2(\rho) * (\omega_2 - \omega); & \omega_1 < \omega < \omega_2 \\ 1 + \phi_1(\rho) * (\omega_1 - \omega_0) + \phi_2(\rho) * (\omega_2 - \omega_1) + \phi_3(\rho) * (\omega - \omega_2); & \omega_2 < \omega < \omega_3 \end{Bmatrix}. \tag{5}$$

In formula (5), $\phi(\rho)$ is the congestion penalty amplification function, and ω is the critical perception threshold of the full load rate, which can be adjusted according to the actual situation.

(5) Amendment fees for emergencies $\sum_k^o \longrightarrow {}^{\mathrm{d}} C_{\mathrm{delay}}$:

According to statistics, the delay time of emergencies obeys a similar discrete probability distribution. The occurrence of emergencies will have certain impacts on the corresponding lines, stations, and sections, such as slower trains and delays caused by congested transfers, which will increase the travel time of passengers who choose that route.

3. Prediction and Early Warning of the Temporal and Spatial Distributions of Passenger Flow

3.1. Passenger Route Choice Model Based on Cumulative Prospect Theory. Tversky and Kahneman [32] proposed cumulative prospect theory based on hierarchy-dependent utility theory. Cumulative prospect theory focuses on personal, psychological, and behavioral characteristics and replaces the decision weight in the prospect theory with a cumulative decision weight to better solve the problem of random dominance and the processing of multiple results.

Cumulative prospect theory divides the decision maker's risk selection process into two stages: editing and evaluation. The editing stage can be divided into setting reference points, establishing value functions, determining personal subjective probabilities, and establishing decision weight functions. In the evaluation stage, cumulative prospect theory considers the sort dependence of the probability of various possible outcomes of the alternatives and calculates and compares the prospects of the alternatives.

The prospect value is an index on which decision makers rely, as shown in the following formula:

$$V = v(x)\pi(p). \tag{6}$$

In formula (6), $v(x)$ is the value function, and $\pi(p)$ is the decision weight function.

(1) Value function:

The specific expression of the value function $v(x)$ is as follows:

$$v(x) = \begin{cases} a(x_0 - x)^{\alpha}, & x_0 \geq x, a > 0, \alpha > 0 \\ -b(x - x_0)^{\beta}, & x_0 < x, b > 0, \beta > 0 \end{cases} \tag{7}$$

In formula (7), x is the random event result, x_0 is the reference point, a is the profit pursuit coefficient, b is the loss avoidance coefficient, and $0 < a < b$.

α and β reflect the risk appetite level of the decision maker. The larger the values of α and β are, the more likely the decision maker is to take risks, and $0 < \alpha, \beta < 1$.

(2) Decision weight function:

According to cumulative prospect theory, a prospect event result set $(x_1, x_2, \ldots, x_n)$ corresponds to a probability set $(p_1, p_2, \ldots, p_n)$, the probability of the result x_i is p_i, the result set is arranged in the descending order as $(x_1 \geq x_2 \geq \cdots \geq x_n)$, w^+ represents the probability weight of the return, and w^- represents the probability weight of loss.

The expression of the probability weight function is

$$w^+(p_j) = \frac{p_j^{\gamma}}{\left[p_j^{\gamma} + (1 - p_j)^{\gamma}\right]^{1/\gamma}},$$
$$w^-(p_j) = \frac{p_j^{\delta}}{\left[p_j^{\delta} + (1 - p_j)^{\delta}\right]^{1/\delta}}. \tag{8}$$

The cumulative decision weight function $\pi(p)$ is generated by the probability weight function, and the expression is as follows:

$$\pi_i^+ = w^+(p_i + \cdots + p_n) - w^+(p_{i+1} + \cdots + p_n), \quad 0 \leq i \leq n,$$
$$\pi_j^- = w^-(p_{-m} + \cdots + p_j) - w^-(p_{-m} + \cdots + p_{j-1}), \quad -m \leq j \leq 0. \tag{9}$$

The prospect value of the alternative can be expressed as

$$V = \sum_{i-1}^{n} \pi^+(p_i)v(x_i) + \sum_{j-1}^{-m} \pi^-(p_j)v(x_j). \tag{10}$$

(3) Reference point setting:

As a single individual passenger is affected by many factors, such as personal experience and environment, different passengers have different feelings about the same event. According to research, when the reference point is set to the average value of the generalized cost of each route, it is more in line with the passengers' travel psychology. Therefore, this paper proposes using the average value of the generalized cost as the reference point. The calculation formula of the reference point is as follows:

$$T_{\mathrm{refer}} = \frac{1}{k}\sum_{i=1}^{k} C_i^{OD}. \tag{11}$$

3.2. Construction of the Multipath Probability Allocation Model Based on User Equilibrium. In this paper, based on the prospect value of each scheme calculated by the aforementioned cumulative prospect theory, the allocated passenger flow of each route is calculated according to the following calculation formula:

$$f_k^{OD} = q^{OD} * \frac{\exp\left(-\theta * V_k^{OD}\right)}{\sum_{l \in k\,\mathrm{path}} \exp\left(-\theta * V_l^{OD}\right)}. \tag{12}$$

In formula (12), θ is the randomness of the description model; q^{OD} is the passenger flow between OD pairs; f_k^{OD} is the distribution flow of the k-path between OD pairs; and V_k^{OD} is the cumulative prospect value of the k-path between OD pairs.

This paper uses the MSA algorithm, and the algorithm steps are as follows:

Step 1: initialization.

Step 2: calculation of impedance based on the generalized travel cost of passengers, use of the logic distribution method to distribute the passenger flow, and calculation of the route passenger flow and section passenger flow.

Step 3: iterative calculation.

Step 4: judge whether the convergence is based on the convergence function, convergence value, and number of convergence steps. Step 2 is repeated if the requirements are not met.

3.3. Overall Process of Simulating the Passenger Flow Deduction. The overall process of predicting the passenger flow within an urban rail transit network in real time is shown in Figure 1. The specific steps are as follows:

Step 1: predict the initial stage

The data are prepared and connected to the database before being read into all the infrastructure tables and parameter tables in the simulation process.

Step 2: prediction phase

The current state of passenger flow is determined (normal passenger flow prediction or emergency passenger flow prediction). The prediction stage includes the inbound volume prediction, passenger flow *OD* prediction, and the passenger flow multipath distribution prediction. This paper employs real-time AFC statistical data and emergency line and length estimates by connecting to the database.

Inbound traffic forecast: at the beginning of each forecast time, based on the historical passenger flow inbound data table, the inbound traffic at each station is forecast within each forecasting period.

Passenger flow *OD* prediction: according to the historical passenger flow *OD* matrix, the inbound passenger flow is allocated according to the destination, and the passenger flow of each station's inbound passenger flow to the remaining stations, that is, the predicted passenger flow *OD*, is predicted in each forecast period.

Passenger flow multipath distribution prediction: the proportion of passenger flow that may be allocated for each route according to the cumulative prospect theory model is calculated, and the proportion of *OD* traffic to each route is allocated.

Step 3: simulation phase

A simulation multiagent model is constructed according to the interaction among the overall road network scene, station, and passengers and other agents with their respective behavior rules, and accurate simulations are conducted considering the time, dynamic deduction, and loading of passenger flow data according to the time, statistics, and output according to the demand simulation calculation results within a certain time range.

Step 4: index calculation and storage of the result stage

After the simulation calculation is completed, the calculation result data are quickly written into the database, and a report is generated.

Step 5: forecast end data update phase

In the data update stage, the corrected AFC data and *OD* input data are updated in the same period.

3.4. Passenger Flow Impact Index System and Calculation of Emergencies. After the dynamic passenger flow allocation and simulation, it is necessary to identify the impact of the emergency passenger flow and perform an early warning of passenger flow based on the magnitude of the impact. Based on the detection of passenger flow anomalies, this paper establishes an urban rail transit emergency passenger flow impact index calculation model and then conducts a dynamic evaluation of the impacts of different types and levels of emergency events on passenger flow.

The index includes three aspects: degree of influence, scope of influence, and duration of influence. From the perspective of the severity of passenger congestion, the index is divided into different levels of congestion, namely, comfortable, general, and congested; from the perspective of the scope of influence, the index is divided into three levels, namely, station, line, and network; and from a time point of view, the index is divided into real time, short term, long term, and other levels. The influencing factors of the passenger flow impact index of emergencies are shown in Figure 2.

There are four passenger flow data sources for the emergency passenger flow impact index: inbound passenger flow, outbound passenger flow, interval passenger flow, and interchange passenger flow. The index calculation method proposed in this paper is based on the original passenger flow data. Compared with step-by-step recursive calculation methods such as the "station-line-network" approach, using the original flow data avoids an accumulation of errors due to step-by-step merging and is therefore more accurate. The passenger flow impact index system and calculation process of urban rail transit emergencies are shown in Figure 3.

4. Case Analysis of Emergencies Based on the Beijing Rail Transit Network

4.1. Data and Preliminary Processing. Urban rail transit AFC data include the inbound number, outbound number, line number, inbound time, and outbound time. This paper extracts and analyzes AFC data to extract the inbound and outbound passenger flow, sectional flow, and other data.

This paper selects the three-day normal daily passenger flow data before and after the emergency day (excluding special times, such as national statutory holidays and large-scale events), and the track passenger flow time period is 15 minutes. To ensure the accuracy of the identification of

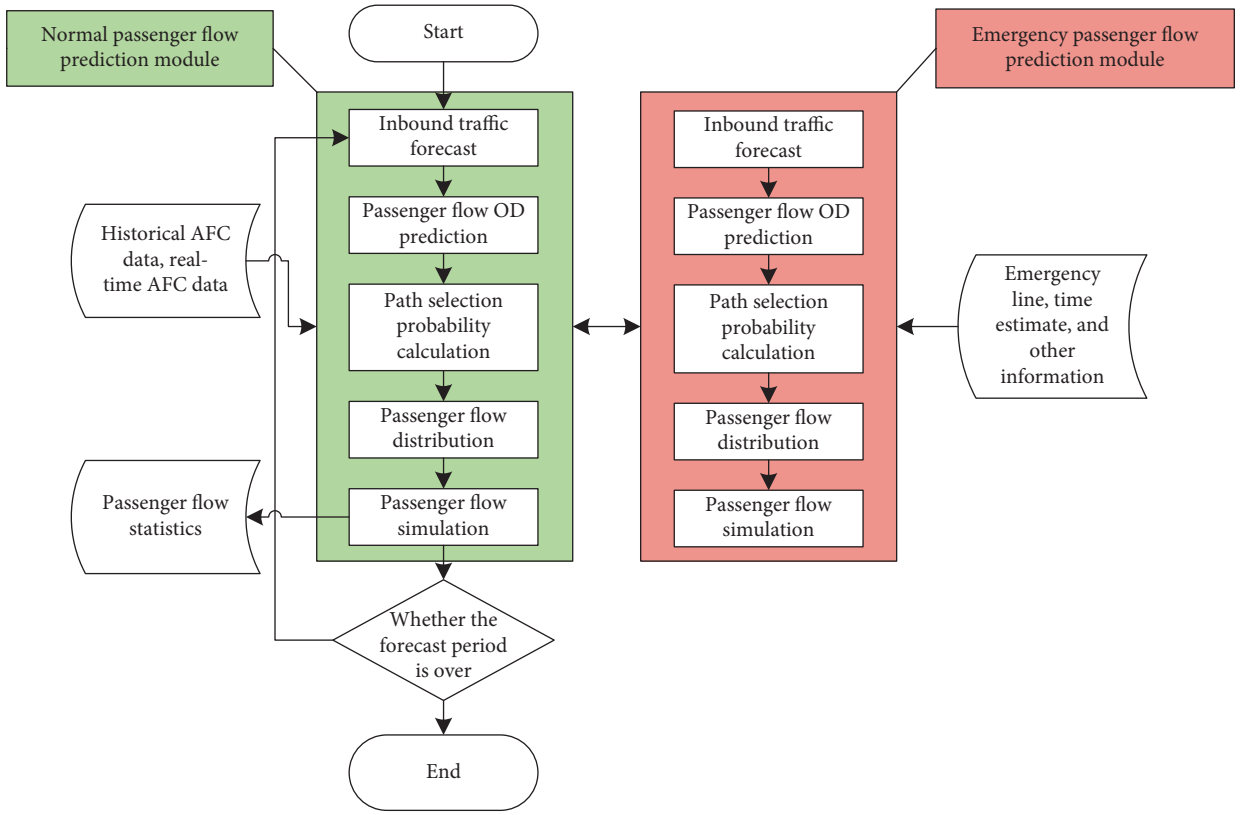


Figure 1: Overall process of the real-time prediction of passenger flow in an urban rail transit network.

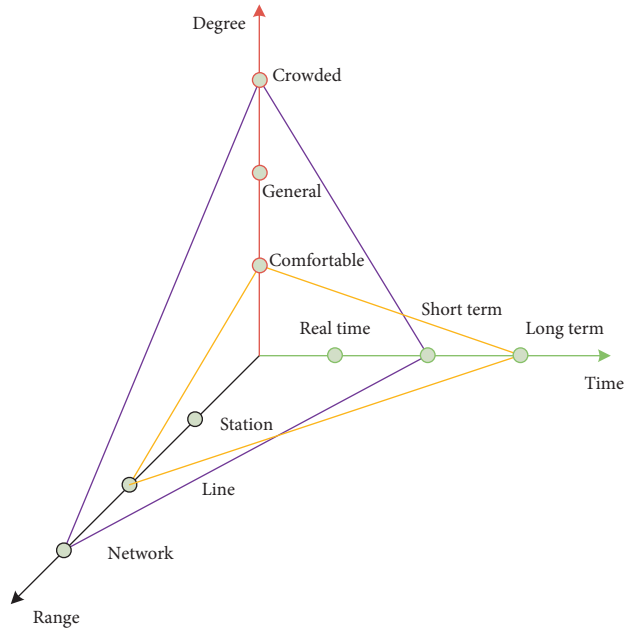


Figure 2: Factors affecting the passenger flow impact index under emergencies.

emergencies, based on previous experience and research, within a given period of time (excluding special times, such as national statutory holidays and large-scale events), the passenger flow data for a certain period on the same working day follow a normal distribution. This paper cleans the passenger flow data according to the 3σ principle and removes the abnormal passenger flow data from the normally distributed historical passenger flow data.

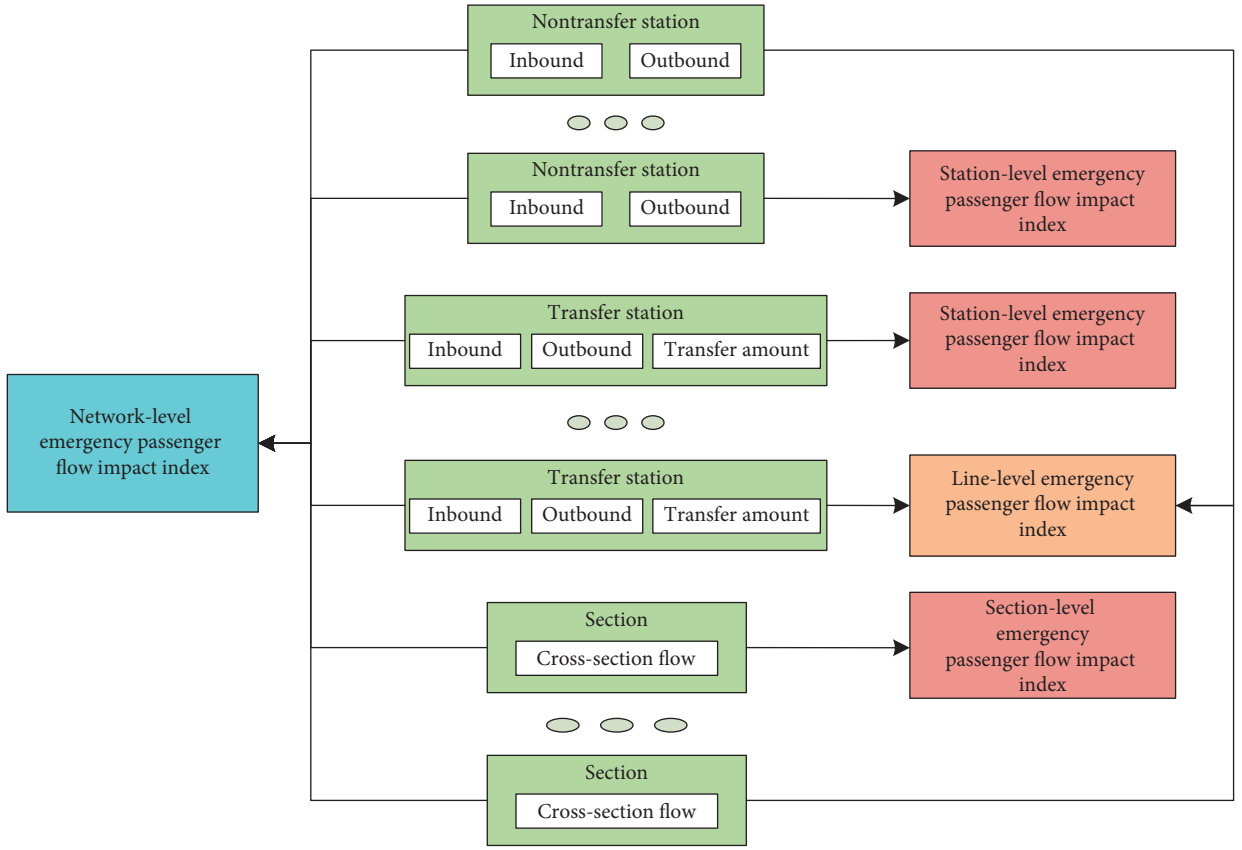


Figure 3: Passenger flow impact index system and calculation process of urban rail transit emergencies.

4.2. Analysis of Passenger Travel Behavior under Emergencies Based on a Multiple Logistic Regression Model. During an emergency, the travel choice behavior of passengers in the rail transit system will change. Based on questionnaire survey data and AFC passenger flow data, the study of passengers' travel choice behaviors provides basic support for *OD* passenger flow analysis and travel delay time estimation. This paper considers only the situation when passengers are already inside the rail transit system when an emergency occurs.

4.2.1. Multinomial Logistic Travel Choice Behavior Regression Modeling Principle. The choice behavior of passengers after an emergency is selected as the dependent variable, and the conditional probability of choosing the i-th choice behavior is $P_i, i \in (1, 2, \ldots, j-1, j+1, \ldots, m)$. The choice of the first passenger choice behavior is selected as the reference level, and the corresponding choice probability is P_j, $j \in (1, 2, \ldots, m)$. Then, the multiclass logistic regression model is as follows:

$$\ln\left(\frac{P_i}{P_j}\right) = \ln\left[\frac{P(y=i)|x}{P(y=j)|x}\right] = \alpha_i + \sum_{w=1}^{n} \beta_{iw} x_w. \quad (13)$$

In formula (13), x_w is the independent variable; n is the number of independent variables; m is the number of dependent variables; α_i, β_{iw} are the independent variable regression coefficient vectors; and $\ln(P_i/P_j)$ is the occurrence ratio of choice to choice.

4.2.2. Design of the Dependent and Independent Variables. According to the analysis, for the urban rail transit system, the travel choice behavior of passengers after an emergency occurs is divided into the following: leaving the station and changing to a bus, taxi, etc., to reach the destination; leaving the station and taking the rail transit from other rail stations but not leaving the station; changing the travel route and waiting for the incident to resume; and continuing to take the four types of rail transit travel. These behaviors are coded as shown in Table 3. This paper selects 15 candidate independent variables and encodes the categorical variables as shown in Table 4.

4.2.3. Selection Behavior Analysis. According to the passenger choice behavior survey questionnaire, a logistic regression model was used to analyze the travel behavior of urban rail transit passengers, and the following conclusions were obtained:

(1) The social and economic attributes of passengers have certain impacts on passengers' behavior choices when facing emergencies, but these impacts are relatively small and can be ignored.

Table 3: Description of dependent variables.

Chosen behavior	Code
Exit the station and change to a bus, taxi, etc., to reach the destination	1
Exit the station and take the rail transit from other rail stations	2
Do not leave the station and change the travel route	3
Wait for the emergency to end and continue to take the rail transit	4

Table 4: Description of independent variables.

Continuous variable		Travel distance
Categorical variables	Passenger socioeconomic attributes	Gender
		Age
		Job
		Monthly income
	Passenger travel attributes	Travel purpose
		Network familiarity
	Passenger's perception of emergencies	Have you encountered an emergency?
		The longest tolerable event
		The remaining travel time of passengers after the emergency
	Emergency attributes	Time of the occurrence
		Location of the occurrence
		Influential impact
	Information interference factors	Influencing station information
		Detour information

(2) When an emergency occurs, passengers who are familiar with the urban rail transit network tend to choose to detour inside the rail transit system. In contrast, passengers who occasionally take urban rail transit travel tend to stay and wait for the emergency to end because the urban rail transit network is relatively unfamiliar; these passengers are more flustered during emergencies, and there is a phenomenon of herding.

(3) The occurrence time, location, and impact of emergencies have important impacts on the choice of passengers. When predicting and guiding the passenger flow of emergencies, priority should be given to the selection of passengers under different occurrence times, locations, and impacts.

(4) In the case of giving passengers a certain amount of information to consider, most passengers will choose to change their initial choice behavior, and external information interference has an important influence on the choice of passengers.

4.3. Case Analysis of Emergencies Based on the Beijing Rail Transit Network. This article takes the Beijing urban rail transit network as an example with Tiantandongmen Station as the starting point and Zhangzizhonglu Station as the end point to analyze the route. The route contains three possible pathways, as shown in Figure 4.

When no emergencies occur, the attributes of the three alternative paths are shown in Table 5. It can be seen that the travel times of paths 2 and 3 are longer than the travel time of path 1 by 12 minutes, and both have three transfers, which are time-consuming. The probability of choosing path 1 is 100%.

This article assumes that an emergency occurs in the section from Chongwenmen Station to Dongdan Station, and the time to resume traffic cannot be determined. At this time, path 1 is not accessible, paths 2 and 3 are affected by emergencies, and the path travel events are delayed to a certain extent. The delay time of each path caused by the emergencies is set to follow the distribution shown in Table 6.

According to the calculation of cumulative prospect theory, the selection probability of path 2 is 54.63%, and the selection probability of path 3 is 45.37%, which are closer to the results of the questionnaire survey. According to this analysis, passengers are more sensitive to losses, and the occurrence of emergencies makes path 2 lose fewer passengers than path 3, which leads to different choices of travel routes for passengers.

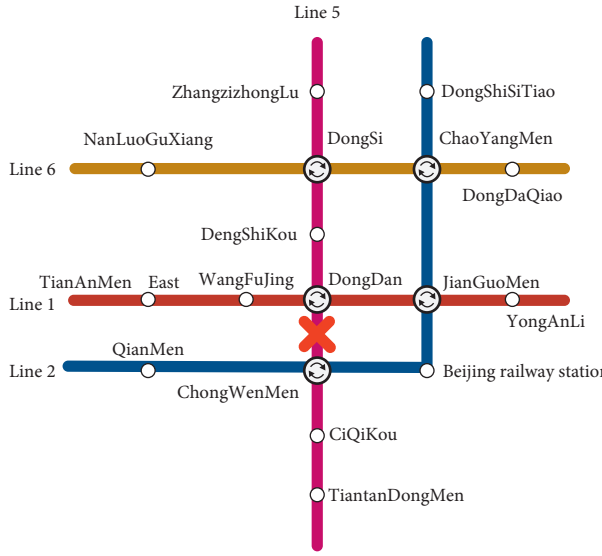


Figure 4: Travel route diagram.

Table 5: Path attribute table when no emergencies occur.

ID	Path	Travel time (s)	Number of transfers	Transfer time (s)	Selection ratio
1	Tiantandongmen-Ciqikou-Chongwenmen-Dongdan-Dengshikou-Dongsi-Zhangzizhonglu	730	0	0	100%
2	Tiantandongmen-Ciqikou-Chongwenmen-Beijing railway station-Jianguomen-Chaoyangmen-Dongsi-Zhangzizhonglu	860	3	1030	0
3	Tiantandongmen-Ciqikou-Chongwenmen-Beijing railway station-Jianguomen-Dongdan-Dengshikou-Dongsi-Zhangzizhonglu	924	3	840	0

Table 6: Probability distribution of delay time.

Path	Delay time (s)	Probability of delay (%)
2	300	80
	600	20
3	600	80
	900	20

5. Conclusions

To study the travel choice behavior of rail transit passengers under emergencies in depth, this paper proposes a prediction and early-warning model for the temporal and spatial distributions of rail transit passenger flow based on a cumulative prospect theory model, which makes up for the deficiencies of previous studies. Compared with the existing prediction methods, this method better simulates the travel decision-making process of rail transit passengers under emergencies, is closer to reality, and provides strong support for the accurate prediction of passenger flow between rail transit stations under emergencies. The model provides a theoretical basis for the government and related departments to formulate traffic management measures.

Specifically, this paper introduces cumulative prospect theory to study the route choice of urban rail transit passengers and performs the following work:

(1) The passenger's route choice preferences are analyzed under different influencing factors, and a generalized passenger travel cost model is established.

(2) The principle of establishing the reference point in the cumulative prospect theory model is analyzed, and a prediction and early-warning model for the temporal and spatial distributions of passenger flow during rail transit emergencies is constructed based on the cumulative prospect theory.

(3) According to the proposed prediction and early-warning model for the temporal and spatial

distributions of passenger flow during rail transit emergencies based on the cumulative prospect theory, a logistic regression model is used to analyze the travel behavior of urban rail transit passengers, and the Beijing urban rail transit network is taken as an example. The results show that the cumulative prospect theory can more accurately describe the decision-making behavior of passengers in route selection and can better reflect the needs of passengers traveling by rail transit under emergencies.

How to further update the data in the model to obtain more accurate results, study the changes in passenger behavior from day to day after short-term incidents, and develop a passenger flow organization method of subway stations when short-term incidents occur are all directions of future research.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest in this paper.

Acknowledgments

This work was jointly supported by the National Natural Science Foundation of China (no. U1834211), Postdoctoral Innovative Talent Project (no. BX20190029), and the State Key Laboratory of Rail Traffic Control and Safety (Contract no. RCS2020ZZ002).

References

[1] M. Zhou, H. Dong, B. Ning, and F. Wang, "Parallel urban rail transit stations for passenger emergency management," *IEEE Intelligent Systems*, 2019, In press.

[2] S. Liu, E. Yao, B. Li, and Y. Tang, "Forecasting passenger flow distribution between urban rail transit stations based on behavior analysis under emergent events," *Journal of the China Railway Society*, vol. 40, no. 9, pp. 22–29, 2018.

[3] X. Zhang, X. Li, J. Mehaffey, and G. Hadjisophocleous, "A probability-based Monte Carlo life-risk analysis model for fire emergencies," *Fire Safety Journal*, vol. 89, pp. 51–62, 2017.

[4] N. Zarboutis and N. Marmaras, "Searching efficient plans for emergency rescue through simulation: the case of a metro fire," *Cognition, Technology & Work*, vol. 6, no. 2, pp. 117–126, 2004.

[5] S. Seer, D. Bauer, N. Brandle, and M. Ray, "Estimating pedestrian movement characteristics for crowd control at public transport facilities," in *Proceedings of the 2008 11th International IEEE Conference on Intelligent Transportation Systems*, pp. 742–747, Beijing, China, October 2008.

[6] H. Wang, L. Hong, and R. Xu, "Analysis of the emergency evacuation guide at metro station," *Urban Mass Transit*, vol. 15, no. 1, pp. 70–74, 2012.

[7] W. Zhu, "Mass passenger flows at rail transit stations: formation, impacts, and countermeasures," *Urban Transport of China*, vol. 11, no. 3, pp. 55–61, 2013.

[8] X. Li, "Research on forecast passenger flow bursted in urban rail station on data mining," Ph.D. thesis, Beijing Jiaotong University, Beijing, China, 2017.

[9] Y. Zhang and Z. Li, "Subway passenger transport organization under the condition of large passenger flow," *Journal of Transportation Engineering and Information*, vol. 15, no. 2, pp. 58–63, 2017.

[10] L. Zhang and F. Chen, "On large passenger flow operating organizations at metro station," *Urban Mass Transit*, vol. 14, no. 5, pp. 87–90, 2011.

[11] M. Zhou, H. Dong, Y. Zhao, P. A. Ioannou, and F.-Y. Wang, "Optimization of crowd evacuation with leaders in urban rail transit stations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 12, pp. 4476–4487, 2019.

[12] M. Zhou, H. Dong, P. A. Ioannou, Y. Zhao, and F.-Y. Wang, "Guided crowd evacuation: approaches and challenges," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 5, pp. 1081–1094, 2019.

[13] F. Li, R. Xu, and W. Zhu, "Generation of emergency scheme for urban rail transit by case-based reasoning," in *Proceedings of the 12th International Conference on Computer System Design and Operation in Railways and Other Transit Systems*, pp. 529–536, Beijing, China, August 2010.

[14] J. Preston, G. Wall, R. Batley, J. N. Ibáñez, and J. Shires, "Impact of delays on passenger train services," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2117, no. 1, pp. 14–23, 2009.

[15] A. Barron, P. C. Melo, J. M. Cohen, and R. J. Anderson, "Passenger-focused management approach to measurement of train delay impacts," *Transportation Research Record*, vol. 2351, no. 1, pp. 46–53, 2018.

[16] A. M. Pnevmatikou, M. G. Karlaftis, and K. Kepaptsoglou, "Metro service disruptions: how do people choose to travel?," *Transportation*, vol. 42, no. 6, pp. 933–949, 2015.

[17] R. Tsuchiya, Y. Sugiyama, K. Yamauchi, K. Fujinami, and T. Nakagawa, "Route-choice support system for passengers in the face of unexpected disturbance of train operations," in *Proceedings of the Computers in Railways X the Tenth International Conference*, Prague, Czech Republic, July 2006.

[18] H. Sun, J. Wu, L. Wu, X. Yan, and Z. Gao, "Estimating the influence of common disruptions on urban rail transit networks," *Transportation Research Part A: Policy and Practice*, vol. 94, pp. 62–75, 2016.

[19] L. Hong, J. Gao, and R. Xu, "Calculation method of emergency passenger flow in urban rail network," *Journal of Tongji University*, vol. 39, no. 10, pp. 1485–1489, 2011.

[20] J. Li and P. Liang, "Quantitative analysis of affected passenger scale for rail transit operation cessation," *Urban Mass Transit*, vol. 16, no. 8, pp. 59–63, 2013.

[21] T. Huang, H. Zhu, and J. Yang, "The route choice model of urban rail traveler based on prospect theory," *Urban Rapid Rail Transit*, vol. 32, no. 2, pp. 67–71, 2019.

[22] X. Wang and J. Wu, "Cumulative prospect theory based urban railway traffic route choice for travelers in emergent events," *Shandong Science*, vol. 28, no. 2, 2015.

[23] Z. Wang, "Research on auxiliary technology of urban rail transit emergency decision," Ph.D. thesis, Tongji University, Shanghai, China, 2008.

[24] Q. Luo, "Research on theory and simulation of urban rail transit passenger flow distribution based on network operation," Ph.D. thesis, Tongji University, Shanghai, China, 2009.

[25] K. Qiao, "Research on urban rail transit network operation characteristics and train regulation," Ph.D. thesis, Beijing Jiaotong University, Beijing, China, 2015.
[26] L. Wu, "Study on passenger travel route matching and impact of emergence in urban rail transit network based on AFC data," Ph.D. thesis, Beijing Jiaotong University, Beijing, China, 2016.
[27] H. Yu, "A dynamic stochastic disequilibrium passenger flow assignment method and its application on a local disruption ofurban railway network," Ph.D. thesis, Beijing Jiaotong University, Beijing, China, 2015.
[28] F. Liu, "Influence of operation disruption on urban rail transit network and emergency strategy," Ph.D. thesis, Beijing Jiaotong University, Beijing, China, 2015.
[29] R. Xu, J. Ye, and H. Pan, "Method for early warning of heavy passenger flow at transfer station of urban rail transit network under train delay," *China Railway Science*, vol. 35, no. 5, pp. 127–133, 2014.
[30] W. Wang, C. Zhang, C. Zhu, and G. Fang, "Route selection method for emergency logistics based on cumulative prospect theory," *China Safety Science Journal*, vol. 3, 2017.
[31] H. Wang, H. Chen, W. Feng, and W. Liu, "Multi-dimensional travel decision model of heterogeneous commuters based on cumulative prospect theory," *Journal of Zhejiang University*, vol. 2, 2017.
[32] A. Tversky and D. Kahneman, "Advances in prospect theory: cumulative representation of uncertainty," *Journal of Risk & Uncertainty*, vol. 5, pp. 297–323, 1992.

Hindawi
Journal of Advanced Transportation
Volume 2020, Article ID 8867937, 14 pages
https://doi.org/10.1155/2020/8867937

Research Article

2D Lidar-Based SLAM and Path Planning for Indoor Rescue Using Mobile Robots

Xuexi Zhang[1], **Jiajun Lai**[1], **Dongliang Xu**[1], **Huaijun Li**[2] and **Minyue Fu**[3]

[1]*School of Automation, Guangdong University of Technology, and Guangdong Key Laboratory of IoT Information Technology, Guangzhou 510006, China*
[2]*School of Automobile and Engineering Machinery, Guangdong Communication Polytechnic, No. 789, Tianyuan Road, Tianhe District, Guangzhou 510630, China*
[3]*School of Electrical Engineering and Computer Science, The University of Newcastle, University Drive, Callaghan, 2308 NSW, Australia*

Correspondence should be addressed to Huaijun Li; lhj@gdcp.cn

Received 23 June 2020; Revised 29 July 2020; Accepted 21 October 2020; Published 17 November 2020

Academic Editor: Zhiguang Cao

As the basic system of the rescue robot, the SLAM system largely determines whether the rescue robot can complete the rescue mission. Although the current 2D Lidar-based SLAM algorithm, including its application in indoor rescue environment, has achieved much success, the evaluation of SLAM algorithms combined with path planning for indoor rescue has rarely been studied. This paper studies mapping and path planning for mobile robots in an indoor rescue environment. Combined with path planning algorithm, this paper analyzes the applicability of three SLAM algorithms (GMapping algorithm, Hector-SLAM algorithm, and Cartographer algorithm) in indoor rescue environment. Real-time path planning is studied to test the mapping results. To balance path optimality and obstacle avoidance, A^* algorithm is used for global path planning, and DWA algorithm is adopted for local path planning. Experimental results validate the SLAM and path planning algorithms in simulated, emulated, and competition rescue environments, respectively. Finally, the results of this paper may facilitate researchers quickly and clearly selecting appropriate algorithms to build SLAM systems according to their own demands.

1. Introduction

Mobile robots are capable of moving around in their environment and carrying out intelligent activities autonomously, thus having extensive realistic applications, including rescue works. A key enabling technology is simultaneous localization and mapping (SLAM) which allows the robot to estimate its own position using onboard sensors and construct a map of the environment at the same time. With the SLAM technology, real-time path planning can be performed to fulfill complex manoeuvring tasks in rescue works.

SLAM-enabled mobile robots have achieved much success in various scenarios. Peng et al. [1] studied the positioning problem and implementation of SLAM for mobile robots with RGB-D cameras. Shou et al. [2] employed a Raspberry Pi module as the core controller and built a mobile robot for map construction and navigation in indoor environment. Zhang et al. [3] proposed path prediction planning based on the artificial potential field to improve obstacle avoidance. Liu et al. [4] combined the Q-learning algorithm with the deep learning algorithm for path planning, which enabled robots to make reasonable walking paths under complex environmental conditions. Yu et al. [5] applied an improved A^* path planning algorithm to unmanned underwater survey ships, enabling quick obstacle avoidance and return to the preset route. However, these studies did not take into account the impact of the rescue environment on the SLAM algorithm. If these algorithms are directly applied to rescue robots, it may deteriorate the accuracy of path planning and even cause incorrect path planning results. At present, there are still rare systems that

can combine SLAM and path planning for indoor rescue. Therefore, it is necessary to study the impact of the rescue environment on the SLAM algorithm and path planning algorithm and evaluate and select the SLAM algorithm suitable for the rescue environment.

In this paper, we evaluated the results of some commonly used SLAM algorithms in both simulation and real-world environment, tested the path planning algorithms (A^* algorithm and DWA algorithm), and conducted a combined experiment of mapping and path planning regarding the RoboCup competition. These experiments revealed the demerits of some algorithms and provided a benchmark for subsequent algorithm improvement.

The rest of the paper is organized as follows. Section 2 introduces the basic system structure of mobile robots; Section 3 presents the rationale of three commonly used SLAM algorithms; Section 4 briefly describes the A^* algorithm and DWA algorithm for path planning; Section 5 provides and analyzes the experimental and simulation results; Section 6 gives the conclusion.

2. System Structure

The hardware part of the robot studied in this paper is mainly composed of motion control module, Lidar module, vision module, power module, and industrial computer module. The system structure is shown in Figure 1, and the physical map of the robot system is shown in Figure 2.

The main function of the STM32 microcontroller is to acquire and process wheel encoder data and gyroscope data. Map information, path planning, depth camera, and Lidar data are processed by the industrial computer. The industrial computer and STM32 are connected via USB cable to exchange data and instructions. The depth camera is calibrated by using a printed black and white checkerboard. The OpenCV function called by the robot operating system (ROS) is used to extract the corner information from camera images, and then internal and external parameters are obtained through calculations [6]. The industrial computer is fitted with Intel Core i5 processor, 4G memory, 128G access space, and the ubuntu16.04 system.

3. SLAM Algorithms

For a mobile robot, SLAM involves both localization and mapping in an iterative manner by continuously fusing various measurements from the onboard sensors [7, 8]. The sensor module in our system includes Lidar and depth camera to collect environmental information as well as internal measurements from the IMU. A 2D map is to be generated after processing by a mapping algorithm. Depending on the purpose of the map, different SLAM algorithms are available. For our purpose, we will focus on the task of path planning in real time. After various considerations, we decide to study in detail three most suitable SLAM algorithms: GMapping algorithm, Hector-SLAM algorithm, and Cartographer algorithm. GMapping algorithm is based on particle filter pairing algorithm, Hector-SLAM is based on scan matching algorithm, Cartographer is a scan matching algorithm with loop detection, and RGB-D algorithm is an algorithm for mapping using depth images. These several algorithms are representative and widely used algorithms.

3.1. GMapping Algorithm. The GMapping algorithm is a laser-based SLAM algorithm for grid mapping [9, 10]. This is probably the most used SLAM algorithm, currently the standard algorithm on the PR2 (a very popular mobile manipulation platform) with implementation available on *openslam.org*. The algorithm was initially proposed in [10], and the main idea is to use Rao–Blackwellized particle filters (RBPFs) to predict the state transition function. The algorithm is also known as the RBPF SLAM algorithm, named after the use of Rao–Blackwellized particle filters. In [11], two major improvements were made by optimizing the proposal distributions and introducing adaptive resampling, making the algorithm much more suitable for practical applications. It is then dubbed GMapping (G for grid) due to the use of grid maps.

3.1.1. RBPF. Onboard measurements include sensor data from Lidar or camera for images and odometer data from the IMU. A large number of particles are used for state transition function predictions, with each particle representing a possible position of the robot.

The sensor data are denoted by $(z_{1:t} = z_1, z_2, \ldots, z_t)$ and the odometer data by $(u_{1:t} = u_1, u_2, \ldots, u_{t-1})$ for the time period from 1 to t. They are used to estimate the joint posterior probability $p(x_{1:t}, m|z_{1:t}, u_{1:t-1})$ of the robot pose $(x_{1:t} = x_1, x_2, \ldots, x_t)$ and the grip map of the environment, represented by m. Using the Bayes' rule, the posterior probability can be decomposed into

$$\begin{aligned} &p(x_{1:t}, m|z_{1:t}, u_{1:t-1}) \\ &\quad = p(x_{1:t}|z_{1:t}, u_{1:t-1})\, p(m|x_{1:t}, z_{1:t}), \end{aligned} \tag{1}$$

where $p(x_{1:t}|z_{1:t}, u_{1:t-1})$ is the positioning problem, whereas $p(m|x_{1:t}, z_{1:t})$ is the mapping problem. The so-called importance sampling is used in the RBPF. The procedure is as follows:

(i) Sampling: according to the given (previous) proposal distribution, particles $(x_{t-1}^{(i)})$ from the previous generation are sampled. They are then improved by incorporating the most recent observations. Then, new particles $(x_t^{(i)})$ and proposal distributions are generated.

(ii) Weights: the weight $w_t^{(i)}$ of each current particle $x_t^{(i)}$ is calculated using

$$w_t^{(t)} = \frac{p\left(x_{1:t}^{(i)}|z_{1:t}, u_{1:t}\right)}{\pi\left(x_{1:t}^{(i)}|z_{1:t}, u_{1:t-1}\right)}, \tag{2}$$

where $\pi(\cdot)$ is the proposal distribution, which usually is a probabilistic odometry motion model.

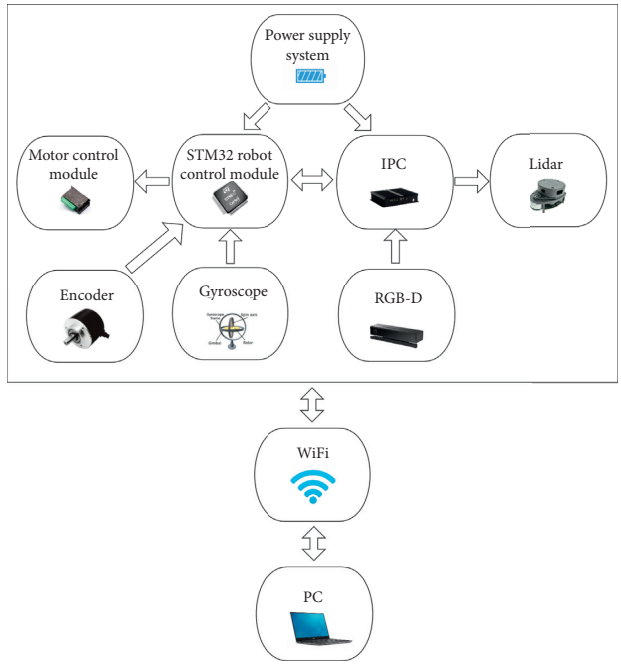
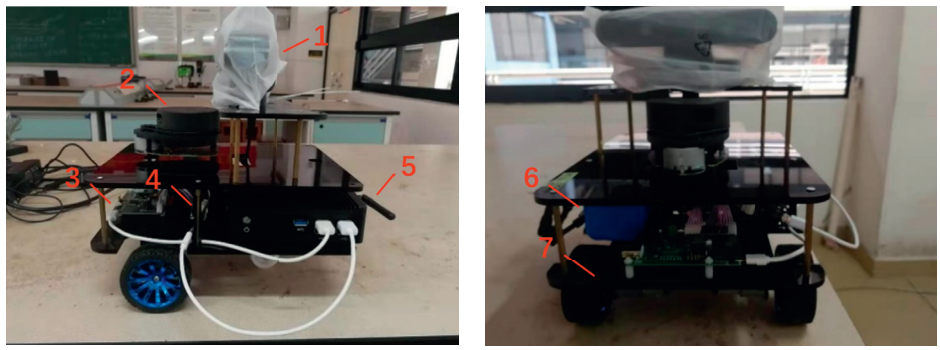


FIGURE 1: System structure of our mobile robot. "STM32 robot control module" gets information of motors from "encoder" and motion from "gyroscope" to control motors and transfer the motion and encoder information to "IPC" which is a microcomputer. Then, "IPC" gets information from Lidar, "STM32 robot control module," and RGB-D camera.



(a) (b)

FIGURE 2: Key modules of the robot. (1) "RGB-D," used to get RGB image and depth image; (2) "lidar," used to get 2D lidar point cloud; (3) "STM32 controller," used to control the movement of the car; (4) "gyroscope," used to get the attitude information; (5) "WI-FI," used to contact with the host computer; (6) "power supply system"; and (7) "motor and encoder," used to get the velocity feedback of the car movement.

(iii) Resampling: depending on the weights, particles with smaller weights are discarded and replaced by resampled particles, but the total number of particles in the resampled particle set is unchanged.

(iv) Map updating: the map update is implemented by the pose represented by each particle in combination with the current observation. To reduce the computational complexity, a recursive formula for weight update is used:

$$w_t^{(i)} = w_{t-1}^{(i)} \eta \frac{p\left(z_t | x_{1:t}^{(i)}, z_{1:t-1}\right) p\left(x_t^{(i)} | x_{1:t-1}^{(i)}, u_{1:t-1}\right)}{\pi\left(x_{1:t}^{(i)} | z_{1:t}, u_{1:t-1}\right)}, \quad (3)$$

where η is a normalisation factor.

3.1.2. Proposal Distribution. A large number of particles will cause a large amount of calculation and memory consumption. In order to reduce the number of particles, a proposal distribution is used. Our target distribution is the best distribution of the robot state according to the data of all sensors carried by the robot. Except for the odometer model, the laser observation data is the position information of 360-degree points, which is difficult to perform Gaussian modelling. Thus, there is no direct way to sample the target distribution, and the proposal distribution is used instead of the target distribution to extract the robot pose information at the next time instant. The proposal distribution considers not only the motion (odometer) information but also the most recent observation (laser) information. This can make the proposal distribution more accurate and closer to the target distribution.

The sensor observation information is added when calculating the proposal distribution, and the sampling process is concentrated in the peak region of the likelihood function to get the optimal proposal distribution:

$$p\left(x_t|x_t^{(i)}, m_{t-1}^{(i)}, z_t, u_{t-1}\right) = \frac{p\left(z_t|x_t, m^{(i)}\right)p\left(x_t|x_{t-1}^{(i)}, u_{t-1}\right)}{p\left(z_t|x_{t-1}^{(i)}, m^{(i)}, u_{t-1}\right)}. \tag{4}$$

Then, the weights are updated according to the above weight recursion formula:

$$\begin{aligned} w_t^{(i)} &= w_{t-1}^{(i)}\eta\frac{p\left(z_t|x_t, m^{(i)}\right)p\left(x_t|x_{t-1}^{(i)}, u_{t-1}\right)}{p\left(x_t^{(i)}|x_{t-1}^{(i)}, m^{(i)}, z_t, u_{t-1}\right)}, \\ &\propto w_{t-1}^{(i)}\eta\frac{p\left(z_t|x_t, m^{(i)}\right)p\left(x_t|x_{t-1}^{(i)}, u_{t-1}\right)}{p\left(z_t|x_{t-1}^{(i)}, m^{(i)}, u_{t-1}\right)}, \\ &= w_{t-1}^{(i)}p\left(z_t|x_{t-1}^{(i)}, m^{(i)}, u_{t-1}\right). \end{aligned} \tag{5}$$

The Gaussian distribution is used to approximate the approximated peak region of the observation, and the optimal proposal distribution is obtained. The Gaussian distribution parameters, i.e., the means $\mu_t^{(i)}$ and covariances $\sum_t^{(i)}$, are determined using K sampling points:

$$\begin{aligned} \mu_t^{(i)} &= \frac{1}{\eta^{(i)}}\sum_{j=1}^{K} x_j p\left(z_t|x_j, m_{t-1}^{(i)}\right)p\left(x_j|x_{t-1}^{(i)}, u_{t-1}\right), \\ \sum_t^{(i)} &= \frac{1}{\eta^{(i)}}\sum_{j=1}^{K} p\left(z_t|x_j, m_{t-1}^{(i)}\right)p\left(x_j|x_{t-1}^{(i)}, u_{t-1}\right), \\ &\cdot \left(x_j - \mu_t^{(i)}\right)\left(x_j - \mu_t^{(i)}\right)^T, \end{aligned} \tag{6}$$

where the normalising factor $\eta^{(i)}$ is given by

$$\eta^{(i)} = \sum_{j=1}^{K} x_j p\left(z_t|x_j, m_{t-1}^{(i)}\right)p\left(x_j|x_{t-1}^{(i)}, u_{t-1}\right). \tag{7}$$

3.1.3. Adaptive Resampling. Resampling may cause good particles to be removed from the filter, making the particles scarce. Therefore, it is necessary to judge the quality of the particles by the effective sampling scale standard and judging the time of resampling. The evaluation formula is as follows:

$$N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N}\left(\tilde{w}^{(i)}\right)^2}, \tag{8}$$

where N is the number of particles and $\tilde{w}^{(i)}$ is the weight of the ith particle. The worse the proposal distribution estimate, the smaller the N_{eff} is. When $N_{\text{eff}} < (1/2)N$, GMapping performs resampling.

3.2. Hector-SLAM Algorithm. The Hector-SLAM algorithm [12] differs from other grid-based mapping algorithms, as it does not require odometer information, but it needs laser data and a priori map. Hector-SLAM is based on the Gauss–Newton iteration formula that optimally estimates the pose of the robot as represented by the rigid body transformation $\xi = [p_x, p_y, \psi]^T$ from the robot to the prior map. The optimal estimation is done by optimally matching the laser data and the map in the sense that the optimal ξ^* below is solved:

$$\xi^* = \arg\min_\xi \sum_{i=1}^{N}\left[1 - M\left(S_i(\xi)\right)\right]^2. \tag{9}$$

Here, $M(S_i(\xi))$ is the value of the map at $S_i(\xi)$, and $S_i(\xi)$ is the world coordinate of scan end points $s_i = (s_{i,x}, x_{i,y})^T$, which obeys the following function:

$$S_i(\xi) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}\begin{bmatrix} s_{i,x} \\ s_{i,y} \end{bmatrix} + \begin{bmatrix} p_x \\ p_y \end{bmatrix}. \tag{10}$$

When an initial estimate of the pose ξ is given, an updated estimate $\xi + \Delta\xi$ is computed by approximating $M(S_i(\xi + \Delta\xi))$ using first-order Taylor expansion, and the result is as follows:

$$\Delta\xi = H^{-1}\sum_{i=1}^{N}\left[\nabla M\left(S_i(\xi)\right)\frac{\partial S_i(\xi)}{\partial\xi}\right]^T\left[1 - M\left(S_i(\xi)\right)\right], \tag{11}$$

where H is the Hessian matrix or some approximation of it, given by

$$H = \left[\nabla M\left(S_i(\xi)\right)\frac{\partial S_i(\xi)}{\partial\xi}\right]^T\left[\nabla M\left(S_i(\xi)\right)\frac{\partial S_i(\xi)}{\partial\xi}\right]. \tag{12}$$

3.3. Cartographer Algorithm. When the amount of data to process becomes too large, particle-based algorithms are not applicable due to their higher computing requirements on the processor. In this case, graph optimisation algorithms are more suitable.

Google's solution to SLAM, called Cartographer, is a graph optimisation algorithm. The Google open source code1 consists of two parts: Cartographer and Cartographer_ROS. The function of Cartographer is to process the data from Lidar, IMU, and odometers to build a map. Cartographer_ROS then acquires the sensor data through the ROS communication mechanism and converts them into the Cartographer format for processing by Cartographer, while the Cartographer processing result is released for display or storage. Impressive real-time results for solving SLAM in 2D have been described in [13] by the authors of the software.

3.4. Considering the Rescue Environment. In rescue environment, there are stairs and rugged surface which make the odometer inaccurate. It means we could not choose GMapping because it is very rely on odometer. Due to the

rugged surface, IMU is also inaccurate which means Cartographer may get bad results. So, we choose Hector-SLAM.

4. Path Planning

To achieve path planning is to solve three basic problems:

(1) The robot reaches the desired position

(2) The obstacle avoidance and completion of the strategic task are achieved in the moving process

(3) The optimal path is realized

However, in the actual environment, due to the accuracy of the robot sensor and the variability of the environment, the environmental information and location information of the map construction will be deviated [14]. Global planning in a static environment can meet the problem requirements, but to handle the deviation caused by the dynamic environment, local path planning needs to be introduced. That is to say, the local path planning pays more attention to obstacle avoidance, and the global path planning pays more attention to the shortest path. Therefore, the combination of local planning and global planning algorithms can successfully achieve accurate navigation of the robot. The global planning algorithm studied in this paper is a node-based A^* algorithm [15, 16], and the local planning algorithm is a dynamic window algorithm (DWA) [17]. Global path planning produces a high-level plan for the robot to follow to reach the goal location. Local path planning is responsible for generating velocity commands for the mobile unit to safely move the robot toward a goal. These properties are imbedded in the plan produced by the planners, using the cost function which takes into account both distance to obstacles and distance to the path.

4.1. Global Path Planning. Based on the global path planning of the grid method, the A^* algorithm is used to study the path planning. The A^* algorithm follows the cost function to make the robot to directionally search for the path toward the end point. The core valuation function of the A^* algorithm is

$$f(n) = g(n) + h(n), \tag{13}$$

where the node n is abstractly understood as the next target point, $f(n)$ represents the total valuation function of the current node n, $g(n)$ represents the actual cost of the starting point to the current point, and $h(n)$ represents the estimated cost of the current node to the end point. The value of $h(n)$ determines the performance of the algorithm. Typically, $h(n)$ uses the Euclidean distance or Manhattan distance between the two points in space. In the A^* algorithm, the Manhattan distance is used. The Manhattan distance between two points (x_1, y_1) and (x_2, y_2) is as follows:

$$D_{\text{Manhattan}} = |x_1 - x_2| + |y_1 - y_2|. \tag{14}$$

4.2. Local Path Planning. This paper mainly studies the corresponding action strategy and operation of robot for path navigation in indoor environment. For this reason, the DWA algorithm is selected as the main algorithm for local path planning. The DWA algorithm requires the robot to perform numerical simulation calculations on the path of the robot within a certain speed window. Thus, it is necessary to obtain the model state expression of the robot. The two-wheeled robot based on differential drive has no velocity in the y-axis direction. Since the robot is at the millisecond level in each sampling period of the program execution, the motion trajectory of the robot in the two adjacent sampling periods can be approximated as a straight line. In a period of time Δ, the robot travels a small distance at speed v, and it is at an angle θ_t to the x-axis; then, the movement increments Δx and Δy of the robot on the x-axis and the y-axis can be obtained, respectively:

$$\Delta x = x + v\Delta t \cos(\theta_t), \tag{15}$$

$$\Delta y = y + v\Delta t \sin(\theta_t). \tag{16}$$

The robot's movement trajectory is then given by

$$x_{t+1} = x_t + v\Delta t \cos(\theta_t), \tag{17}$$

$$y_{t+1} = y_t + v\Delta t \sin(\theta_t), \tag{18}$$

$$\theta_{t+1} = \theta_t + \omega\Delta t, \tag{19}$$

where ω is the angular velocity of the robot.

During the speed sampling of the robot, multiple sets of trajectory velocity values are collected. To make the robot safely perform path planning, some necessary speed limits are also needed. The speed velocity value and angular velocity value of the robot change within a certain range, and the range needs to be empirically calculated according to the physical characteristics of the robot and the operating environment. The range formula is as follows:

$$V_m = \{(v, \omega) | v \in [v_{\min}, v_{\max}], \omega \in [\omega_{\min}, \omega_{\max}]\}. \tag{20}$$

The robot has different torque performance parameters due to different motor models. When the current speed of the robot v_c and the angular velocity ω_c are known, the actual speed range for the next sampling time can be computed as

$$V_a = \{(v, \omega) | v \in [v_c - \dot{v}_d\Delta t, v_c + \dot{v}_a\Delta t], \omega \in [\omega_c - \dot{\omega}_d\Delta t, \omega_c + \dot{\omega}_a\Delta t]\}, \tag{21}$$

where $\dot{v}_a$ and $\dot{\omega}_a$ are the maximum accelerations and $\dot{v}_d$ and $\dot{\omega}_d$ are the maximum decelerations.

When the robot is safely avoiding obstacles in navigation, the speed (v, ω) during the whole locally planned trajectory must be within the range of speeds given by

$$V_d = \left\{(v, \omega) | \sqrt{2\text{dis}(v, \omega)\dot{v}_d} \geq v_c, \sqrt{2\text{dis}(v, \omega))\omega_d} \geq \omega_c\right\}, \tag{22}$$

where $\text{dis}(v, \omega)$ is the minimum distance from the current position to the point where the arc trajectory of v and ω intersects the nearest obstacle.

Performing the DWA algorithm for speed selection needs to satisfy equations (19)–(21) simultaneously. On the basis of the trajectory that satisfies these speed requirements, an evaluation function is used to measure a selected trajectory and to aim the selection of the optimal trajectory. The evaluation function is as follows:

$$G(v, \omega) = \sigma(\alpha\text{head}(v, \omega)) + \beta\text{dis}(v, \omega) + \gamma\text{vel}(v, \omega), \tag{23}$$

where $\text{head}(v, \omega)$ represents the angle difference between the estimated end of the route and the target; $\text{dis}(v, \omega)$ is the minimum distance from the obstacle to the planned trajectory, as explained above; $\text{vel}(v, \omega)$ indicates the moment speed evaluation; $\sigma(\cdot)$ is a smoothing function; and $\alpha, \beta, \gamma > 0$ are evaluation coefficients.

5. Experimental Results

5.1. Simulation Experiments

5.1.1. SLAM Simulation. In order to test the aforementioned algorithms, we carry out simulation experiments using the Gazebo platform to build the simulation environment shown in Figure 3. The virtual environment has real physical properties, and the simulation results have strong reference to the actual environment.

The following simulation experiment was performed according to the virtual environment. We first use Lidar as a sensor to simulate the GMapping, Hector-SLAM, and Cartographer algorithms. We then use depth camera (RGB-D) as a sensor to simulate the GMapping algorithm. Four mapping algorithms under the physical simulation platform of the ROS robot system are tested, and the simulation results are shown in Figures 4–7. The purpose of presenting these diagrams is to give an impression of the algorithms described above, which are parsed using text and formulas.

Figure 4 shows the robot simulation process using GMapping. Depth information of the Lidar is required. The robot is located in the lower left corner of simulation environment, the data collected by the Lidar is marked in red, and the established environment map is in light gray. Figure 5 shows the robot simulation process using Hector-SLAM. Figure 6 shows the robot simulation process using Cartographer. The area scanned by the Lidar changes from light gray to white until the whole map is completed.

Figure 7 shows the final grid map constructed using the RGB-D camera data. The depth data are first transformed using depthimage_to_laserscan before being applied to GMapping. The simulation results show that the constructed map is not ideal. This is because the RGB-D camera is affected by its own structure, causing a limited range of viewing angle [18]. In addition, the depth camera requires rich scene features to work. The simulated environment has smooth wall surfaces with very few scene features, making the depth camera unable to perform effective feature matching. We see that map construction cannot be successfully completed.

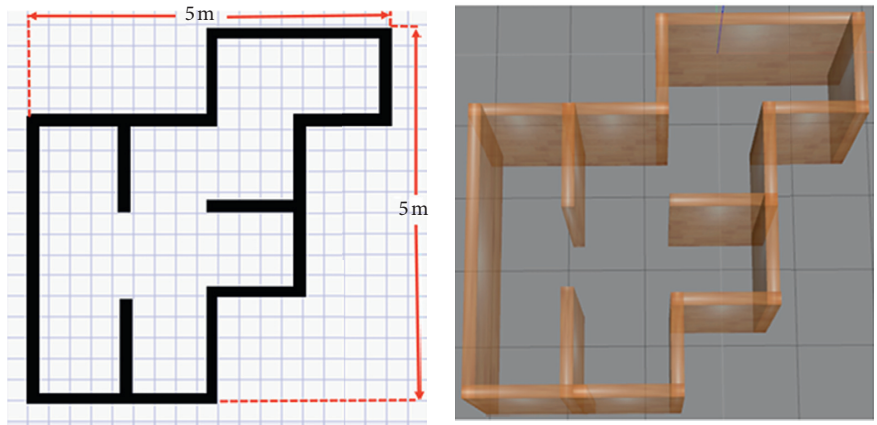
5.1.2. Path Planning Simulation. After obtaining the environment map, the map was loaded under the ROS framework for path planning purposes. We use the rviz package under the ROS framework for path planning and navigation simulation. The constructed map is shown in the left part of Figure 8. Here, we also see the black dot indicating an obstacle, and the cyan portion indicates the safe distance between the robot and the obstacle. The green arrow points to the target point and direction of the robot. The right part of Figure 8 shows the path planning and navigation results, as indicated by the green line. The starting position for the robot is slightly below the obstacle. We see that the simulated path not only successfully avoids the obstacle but also is a nearly straight path.

5.2. Algorithm Verification in Lab Environment

5.2.1. Emulated Rescue Experiment. In this experiment, a 1.25 m × 1.25 m square wooden structure was used to splicing the actual environment in an open laboratory, as shown in Figure 9. This was built with reference to the RoboCup rescue venue to simulate an enclosed indoor rescue environment after a disaster. The aforementioned SLAM algorithms are applied, and the results are shown in Figure 10. Comparing with the simulation results, the maps constructed in the actual environment using Lidar data (GMapping, Hector-SLAM and Cartographer) are all satisfactory. In the RGB-D experiment [19, 20], the environmental features are not sufficient for depth measurements, resulting in an overlapped map. Therefore, the mapping algorithm based on the depth camera needs further optimisation.

5.2.2. Lab Office Experiment. Next, we test the SLAM algorithms in a real lab office, as shown in Figure 11. The map in Figure 12 is constructed using the Hector-SLAM algorithm. We see that not only desks are clearly identified but also the chair legs are clearly shown.

Path planning is carried out using the A^* algorithm for global path planning and DWA algorithm for local path planning, and the results are shown in Figure 13. In the picture on the left, the navigation target point and direction of the robot are set by the green arrow. The small green patch at the bottom shows many arrows representing the particles (their positions and directions) of the starting pose of the robot. The right picture shows that, after the execution of the path planning and navigation, the robot moves to the target



(a) (b)

Figure 3: Simulation environment diagram and Gazebo display.

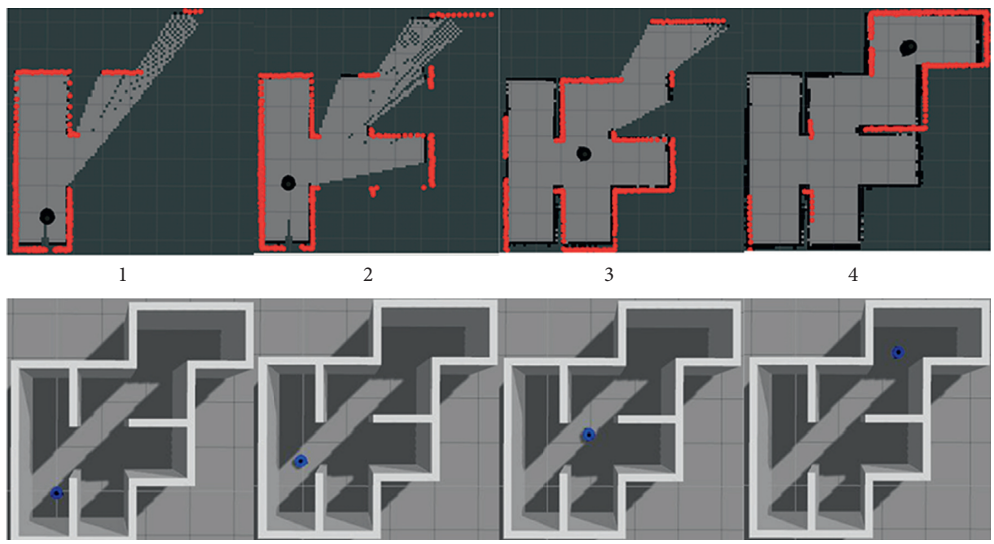


Figure 4: Mapping result of gmapping (simulation).

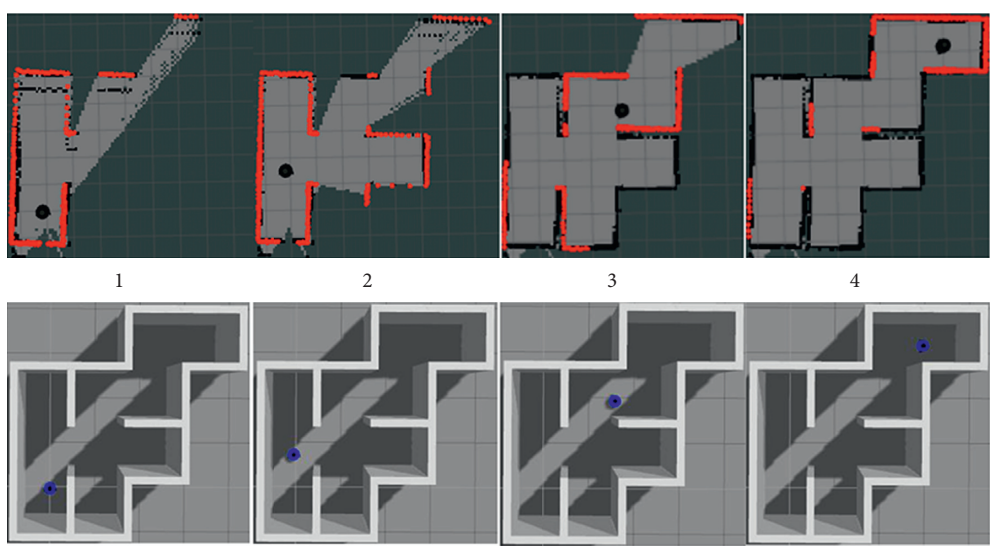


Figure 5: Mapping result of Hector-SLAM (simulation).

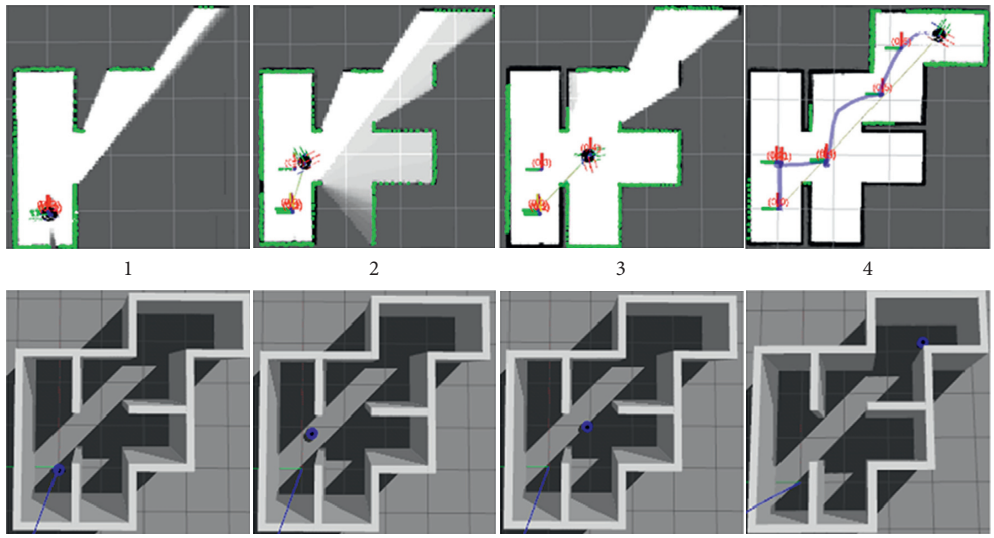


Figure 6: Mapping result of Cartographer (simulation).

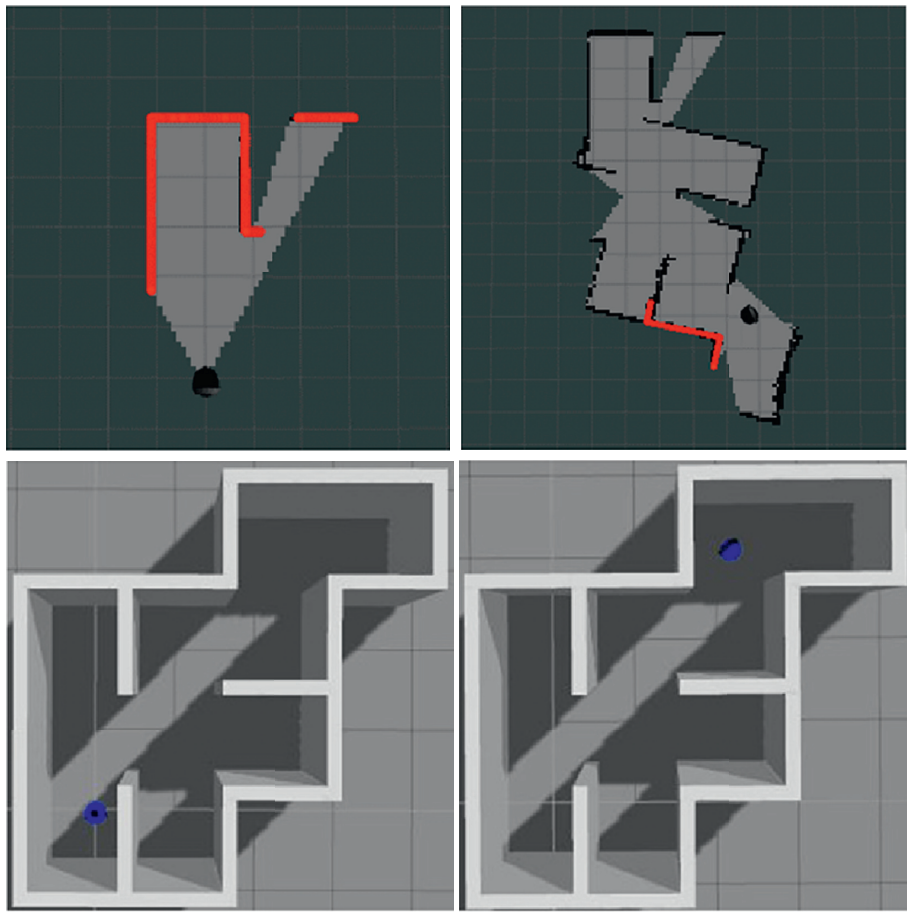
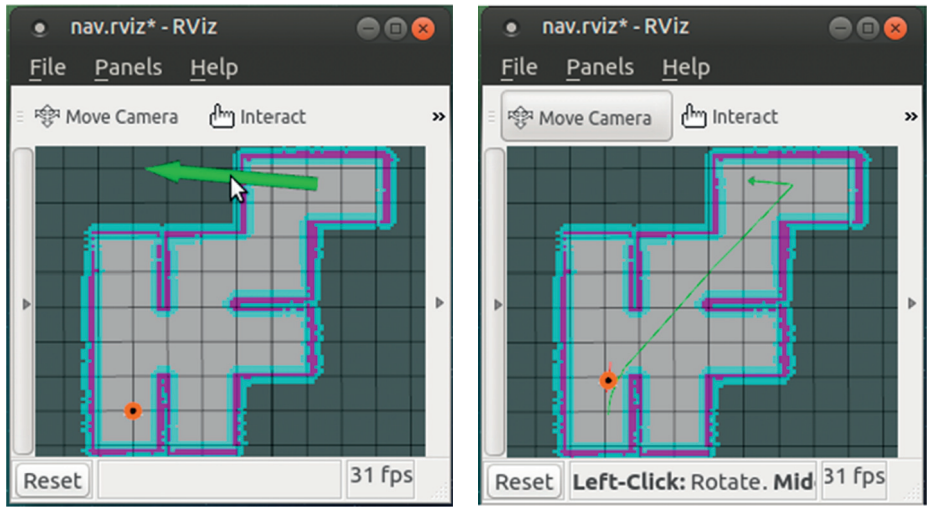


Figure 7: Mapping result of RBG-D (simulation).

point, as indicated by the (smaller) green patch representing the particles of the final pose. We see that the experimental results verify the effectiveness of the path planning algorithms.

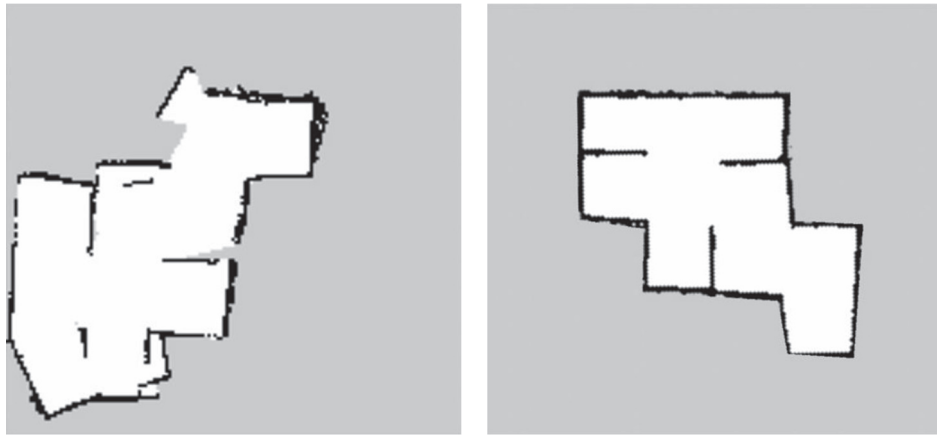
5.2.3. RoboCup Competition Test. The mapping and path planning algorithms above are used in our entry of the 2019 RoboCup competition for indoor rescue [21]. The competition venue is shown in Figure 14. The competition re-



(a) (b)

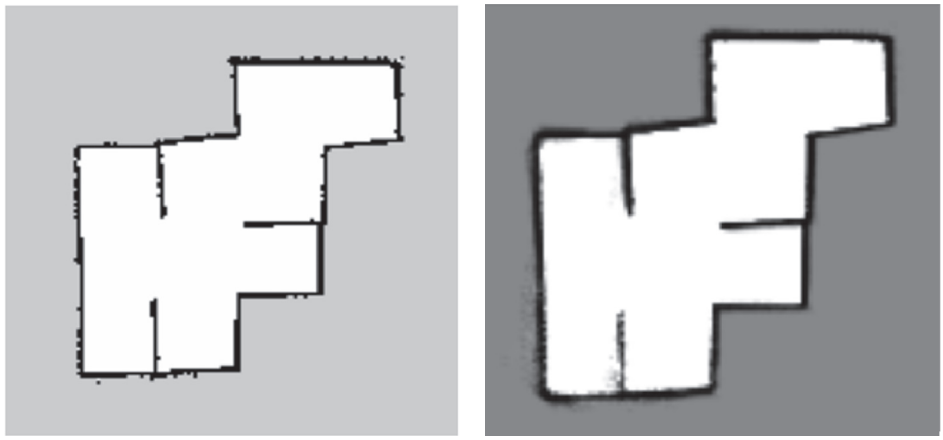
Figure 8: Simulation result of path planning. The green arrow on the left picture means the orientation of the robot when it reaches the end.

Figure 9: Emulated rescue environment.



(a) (b)

Figure 10: Continued.



(c) (d)

Figure 10: Constructed map: (a) RGB-D, (b) Hector-SLAM, (c) GMapping, (d) Cartographer.

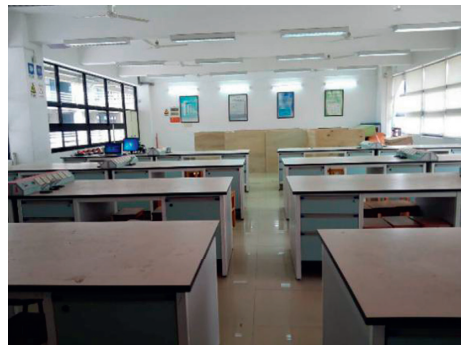
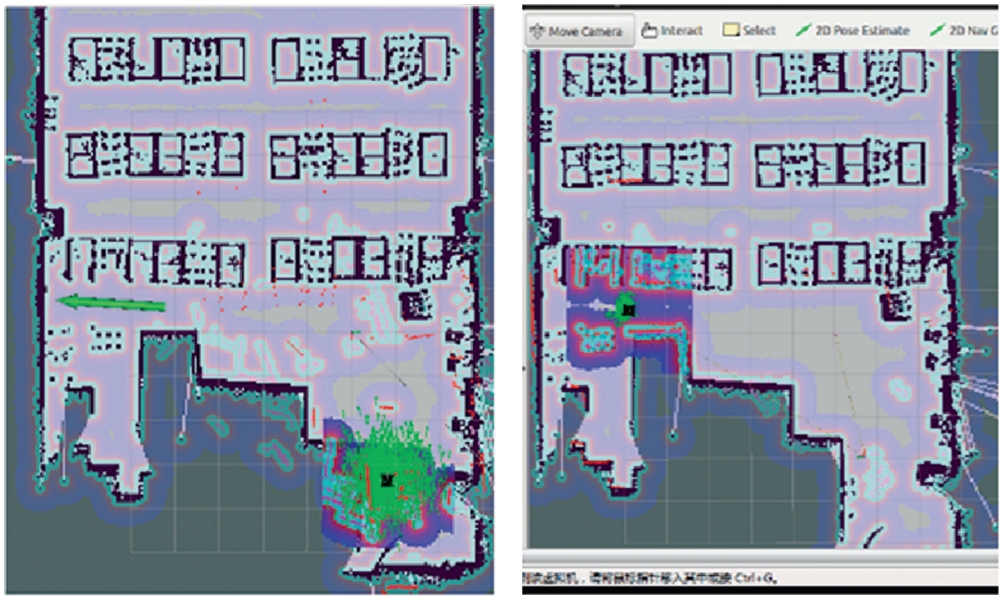


Figure 11: Lab office environment.

Figure 12: Hector-SLAM map for lab office.

quirements are as follows: robot constructs the map of the competition venue by self-exploration and completes the recognition of the doll and the marking of the QR code. The robot operates by either remote control or through an autonomous exploration algorithm. In addition, the robot must avoid obstacles. The quality of the mapping result is assessed by the number of closed grids identified in the constructed grid map. Because GMapping and Cartographer need IMU to assist positioning, and in uneven terrain in the competition, IMU data will have very large errors, which leads to very large errors in these two SLAM algorithms. The RGB-D algorithm also does not perform well in flat terrain,



(a) (b)

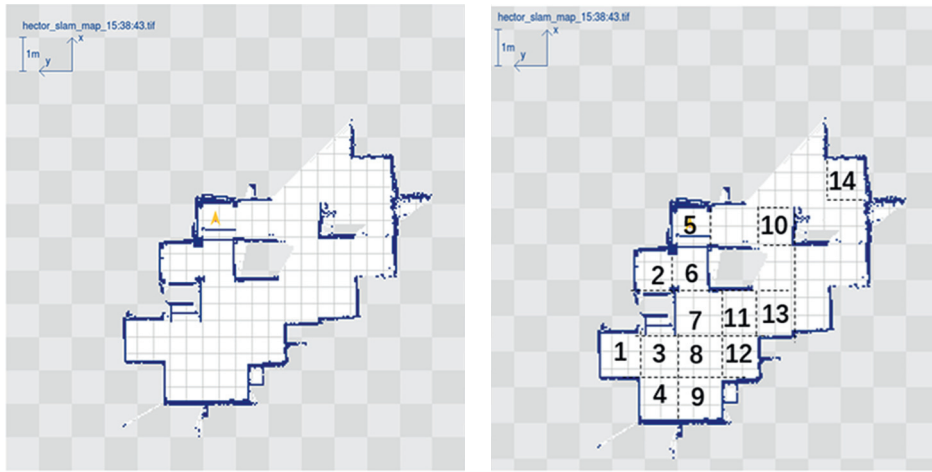
Figure 13: Path planning for lab office.

(a) (b)

Figure 14: RoboCup venue.

so we use the Hector-SLAM algorithm for mapping. And we use the Hector_navigation open source software package [22] for robot self-exploration. As shown in Figure 15, due to the complex environment of the competition venue, the robot is unable to pass some obstacles, such as the 15-degree slope and stairs, so certain parts of the venue cannot be scanned by the Lidar, and the constructed map is incomplete. Due to the limitation of the robot hardware, it is not possible to finish all the mapping. But all the places reached by the robot have been well-mapped. Finally, we scored 14 points (out of 27 points) in the self-exploration session.

5.2.4. China Robot Competition. Compared with the RoboCup competition venue, the 2019 China Robot Competition venue has a larger site area and a larger slope, which means that the terrain is more complicated, as shown in Figure 16. The competition requires rescue robots to independently explore the map of the field and identify the two-dimensional code on the box in the simulated post-disaster environment. In this experiment, we continue to use the Hector_navigation open source software package for robotic exploration. The route of the robot's autonomous navigation is shown in Figure 17. We used two servos to keep the Lidar level, which is used to automatically adjust the Lidar to a horizontal position. However, due to the fact that the competition field is not flat, which causes the robot to have large fluctuations during the movement, the Lidar is unable to adjust the pose in time. As shown in Figure 18, the yellow arrow indicates the starting point of the rescue robot, the purple line marks the movement trajectory of the rescue robot, the dark blue line represents the map constructed by the SLAM algorithm on the competition venue, and the dark blue circle with numbers represents the location of the QR code. However, the robot is navigated outside the wall. As a result, there is a positioning error. The map cannot be quickly updated for corrections, and the navigation algorithm may incorrectly navigate the robot into an obstacle.

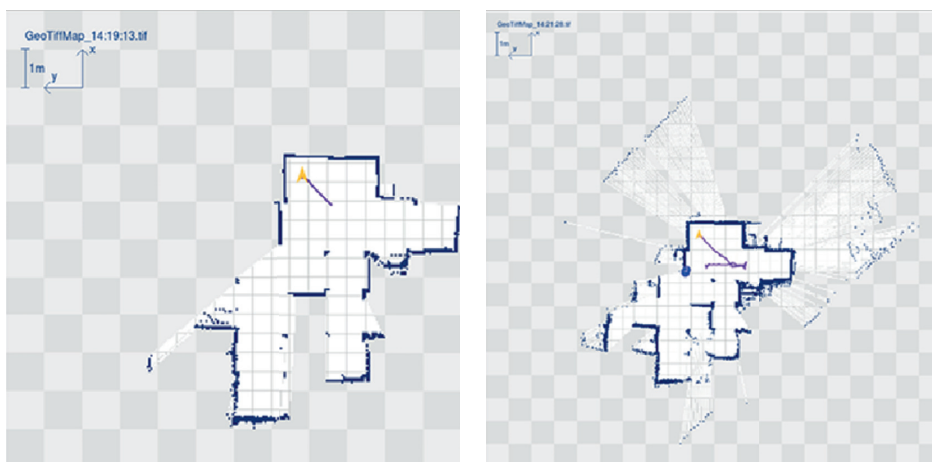


(a) (b)

Figure 15: Mapping result of Hector-SLAM (RoboCup).

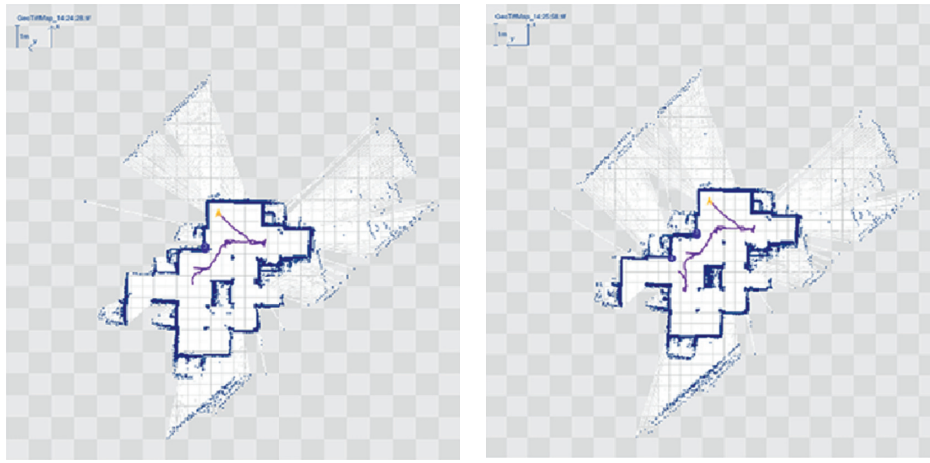
(a) (b)

Figure 16: China robot competition venue.



(a) (b)

Figure 17: Continued.



(c) (d)

Figure 17: Mapping result and autonomous navigation route (China robot competition). (a) 1, (b) 2, (c) 3, (d) 4.

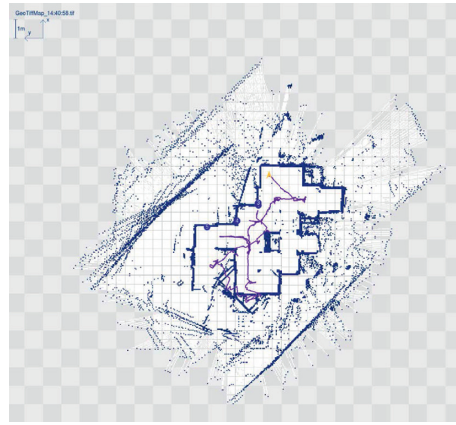


Figure 18: Mapping result of Hector-SLAM (China Robot Competition). Because the level keeping platform of the radar is not sensitive enough, and it cannot be adjusted and keep the platform in time when the robot crosses obstacles, the radar may scan the walls outside the field. Hector cannot judge and filter, so it is recorded on the map.

Therefore, the Hector-SLAM algorithm needs to be improved in the update speed, and the navigation algorithm should adopt a more cautious strategy according to the application of the rescue scenario. As shown in Figure 18, an unnecessary closed point appears on the periphery of the constructed map. In addition, the number in the figure is the position information of the recognised QR code.

In the above simulation experiments and field experiments for GMapping, Hector-SLAM, Cartographer, and RGB-D mapping algorithms, GMapping, Hector-SLAM, and Cartographer perform better in a flat indoor environment. The RGB-D mapping algorithm has poor mapping effect due to poor lighting conditions and lack of environmental features. Therefore, in a relatively flat indoor rescue environment, it is more appropriate to choose the first three algorithms as the basis of the SLAM system. In the RoboCup competition and the China Robot Competition, the uneven rescue environment seriously interfered with the IMU data, so the SLAM algorithm (GMapping and Cartographer) incorporating IMU data could not perform SLAM tasks normally. This causes the path planning algorithm that relies on map information to not work correctly. The Hector-SLAM algorithm, which does not rely on IMU data, can perform tasks in the competition terrain relatively correctly, but there are still problems with inability to filter wrong Lidar information and poor stability.

6. Conclusions

In this paper, the problem of indoor rescue using mobile robots was studied. Comparisons were done on the GMapping, Hector-SLAM, and Cartographer algorithms for SLAM. The path planning was done by combining the A^* algorithm for global path planning and the DWA algorithm for local path planning. Simulation, emulation, as well as real environment experiments were conducted to compare and validate the results on map construction and path planning. In the future, further optimisation needs to be carried out in the mapping algorithms to make them more suitable to the real rescue environment.

Data Availability

Our experiment data can be found in https://github.com/9393dl/Rescue_Robot.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant nos. 61633014, 61803101, and U1701264).

References

[1] W. Peng, F. Yuan, and Z. Zhou, "Positioning analysis and implementation of mobile robot based on RGB-D camera," *Intelligent Computer and Applications*, vol. 9, no. 3, pp. 168–170, 2019.

[2] J. Shou, Z. Zhang, Y. Su, and Z. Zhong, "Design and implementation of indoor positioning and navigation system of mobile robot based on ROS and lidar," *Machinery and Electronics*, vol. 36, no. 11, pp. 76–80, 2018.

[3] H. Zhang, X. Zhang, and Q. Wang, "Path planning based on path prediction artificial potential field method for automatic following trolley," *Computer Measurement and Control*, vol. 27, no. 1, pp. 237–240, 2019.

[4] Z. Liu, S. Jiang, W. Yuan, and C. Shi, "Robot path planning based on deep *Q*-learning," *Measurement and Control Technology*, vol. 38, no. 7, pp. 24–28, 2019.

[5] B. Yu, X. Chu, C. Liu, H. Zhang, and Q. Mao, "Path planning method for unmanned waterway survey ships based on improved A^* algorithm," *Geomatics and Information Science of Wuhan University*, vol. 44, no. 8, pp. 1258–1264, 2019.

[6] S. Sugiyama, H. Okuda, S. Inagaki, and T. Suzuki, "SLAM using ROS and operational assist for electric wheelchairs," in *Proceedings of the JSME Annual Conference on Robotics and Mechatronics (Robomec)*, November 2017.

[7] B. Sun, "Mobile robot SLAM technology," *Electronic Technology and Software Engineering*, vol. 1, no. 2, p. 95, 2018.

[8] Z. Lin and S. Zheng, "Research on image matching and camera pose resolution in mobile robot vision SLAM process," *Machine Design and Manufacturing Engineering*, vol. 46, no. 11, pp. 13–18, 2017.

[9] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[10] A. Doucet, J. F. G. de Freitas, K. Murphy, and S. Russel, "Rao-Blackwellized particle filtering for dynamic bayesian networks," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 176–183, Stanford, CA, USA, June 2000.

[11] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2149–2154, Sendai, Japan, September 2004.

[12] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable slam system with full 3D motion estimation," in *Proceedings of the 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, Kyoto, Japan, November 2011.

[13] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D Lidar SLAM," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2016)*, Stockholm, Sweden, May 2016.

[14] X. Wang, L. He, and T. Zhao, "Mobile robot for SLAM research based on lidar and binocular vision fusion," *Chinese Journal of Sensors and Actuators*, vol. 31, no. 3, pp. 394–399, 2018.

[15] P. Hart, N. Nilsson, B. Raphael et al., "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[16] Z. Cao, S. Jiang, J. Zhang, and H. Guo, "A unified framework for vehicle rerouting and traffic light control to reduce traffic congestion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1958–1973, 2017.

[17] M. Seder and I. Petrovic, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," in *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, pp. 1986–1991, Roma, Italy, April 2007.

[18] Z. Cao, H. Guo, J. Zhang, D. Niyato, and U. Fastenrath, "Finding the shortest path in stochastic vehicle routing: a cardinality minimization approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1688–1702, 2016.

[19] Y. Li, S. Wang, and N. Yu, "RGB-D-based SLAM and path planning for mobile robots," *CAAI Transactions on Intelligent Systems*, vol. 13, no. 3, pp. 445–451, 2018.

[20] J. Cai, K. Chen, and Y. Zhang, "Improved V-SLAM for mobile robots based on Kinect," *CAAI Transactions on Intelligent Systems*, vol. 13, no. 5, pp. 734–740, 2018, http://kns.cnki.net/kcms/detail/23.1538.tp.20180423.0957.004.html.

[21] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa, "RoboCup: the robot world cup initiative," in *Proceedings of the International Conference on Autonomous Agents*, Marina Del Rey, CA, USA, February 1997.

[22] S. Kohlbrecher, J. Meyer, T. Graber, K. Petersen, U. Klingauf, and O. Von Stryk, "Hector open source modules for autonomous mapping and navigation with rescue robots," in *Proceedings of the Robot Soccer World Cup*, pp. 624–631, Joao Pessoa, Brazil, January 2014.

Hindawi
Journal of Advanced Transportation
Volume 2020, Article ID 8859689, 12 pages
https://doi.org/10.1155/2020/8859689

WILEY | Hindawi

Research Article

UB-LSTM: A Trajectory Prediction Method Combined with Vehicle Behavior Recognition

Haipeng Xiao,[1] Chaoqun Wang,[1] Zhixiong Li,[2] Rendong Wang,[3] Cao Bo,[1] Miguel Angel Sotelo,[4] and Youchun Xu[3]

[1]*Army Military Transportation University, Tianjin 300181, China*
[2]*School of Mechanical, Materials, Mechatronic and Biomedical Engineering, University of Wollongong, Wollongong, NSW 2522, Australia*
[3]*Institute of Military Transportation, Army Military Transportation University, Tianjin 300181, China*
[4]*Department of Computer Engineering, University of Alcalá, Alcalá de Henares (Madrid) 28801, Spain*

Correspondence should be addressed to Zhixiong Li; zhixiong_li@uow.edu.au and Rendong Wang; wrd1992@163.com

Received 3 June 2020; Revised 9 July 2020; Accepted 13 July 2020; Published 1 August 2020

Academic Editor: Wen LIU

In order to make an accurate prediction of vehicle trajectory in a dynamic environment, a Unidirectional and Bidirectional LSTM (UB-LSTM) vehicle trajectory prediction model combined with behavior recognition is proposed, and then an acceleration trajectory optimization algorithm is proposed. Firstly, the interactive information with the surrounding vehicles is obtained by calculation, then the vehicle behavior recognition model is established by using LSTM, and the vehicle information is input into the behavior recognition model to identify vehicle behavior. Then, the trajectory prediction model is established based on Unidirectional and Bidirectional LSTM, and the identified vehicle behavior and the input information of the behavior recognition model are input into the trajectory prediction model to predict the horizontal and vertical speed and coordinates of the vehicle in the next 3 seconds. Experiments are carried out with NGSIM data sets, and the experimental results show that the mean square error (MSE) between the predicted trajectory and the actual trajectory obtained by this method is 0.124, which is 97.2% lower than that of the method that does not consider vehicle behavior and directly predicts the trajectory. The test loss is 0.000497, which is 95.68% lower than that without considering vehicle behavior. The predicted trajectory is obviously optimized, closer to the actual trajectory, and the performance is more stable.

1. Introduction

Trajectory prediction is an important research direction in the field of autopilot [1, 2]. The research on the decision-making characteristics of the driver shows that factors such as the relative speed and relative distance between the car and the surrounding moving vehicles will greatly affect the driver's decision [3] and then affect the driving safety. For static vehicles in the driving environment, intelligent vehicles can drive safely along the planned trajectory; for dynamic vehicles, human drivers can use past experience and intuition to predict the behavior of other drivers to avoid potential accidents [4]. Intelligent vehicles need to improve their driving safety by predicting the trajectories of the moving vehicles around them in real time.

The existing methods are basically separate research on vehicle behavior recognition and vehicle trajectory prediction, and there are not many methods to combine the two to make a more accurate trajectory prediction [5–10]. Improving the accuracy of vehicle trajectory prediction is the most urgent problem to be solved. In fact, the accurate identification of vehicle behavior is very important to improve the accuracy of vehicle trajectory prediction, so this paper will take vehicle behavior recognition into consideration.

There are many existing trajectory prediction methods, such as the Markov model [11]. Its advantage is that it can

calculate the probability of maintenance capability and multiple degraded state systems, but it is not suitable for long-term prediction and is easily affected by the external environment. The advantage of Bayesian model [12] is that it is not sensitive to missing data but needs to know a priori probability, and a priori probability often depends on assumptions, and there can be many hypothetical models; therefore, at some point, the prediction effect is poor due to the hypothetical a priori model. The advantage of Kalman filter [13] is that the prediction in a short time (1 step or 2 steps) can be judged stably and accurately, but the trajectory prediction for a long time (such as more than 3 seconds or more than 5 steps) will seriously affect the prediction accuracy due to the increase of prediction error, and the model is very complex and easy to be affected by external noise.

LSTM has a long-term memory function and can learn to translate languages; control robots; make image analysis, document summaries, speech recognition, image recognition, and handwriting recognition; control chatbots, predict diseases; click rates and stocks; and synthesize music [14–17]. Vehicle trajectory prediction is also a time series problem, and there is a correlation between each track point and the historical track point, so it is very suitable to use LSTM to solve the trajectory prediction problem.

In order to solve the problem of vehicle trajectory prediction, a UB-LSTM vehicle trajectory prediction model combined with vehicle behavior recognition is proposed in this paper. Firstly, a many-to-one vehicle behavior recognition model is established based on LSTM, and then a trajectory prediction model is established based on one unidirectional LSTM and one bidirectional LSTM. The identified vehicle behavior information and the input information of the vehicle behavior recognition model are input into the trajectory prediction model to predict the horizontal and longitudinal speed and coordinates of the vehicle, and in the follow-up process, based on the predicted horizontal and longitudinal velocity, an acceleration trajectory optimization algorithm is proposed, and the predicted trajectory based on the optimization algorithm is more consistent with the actual trajectory and more stable. The contributions of this paper are as follows:

A better trajectory prediction model is proposed, and a small test loss is obtained. In this paper, the influence of the surrounding vehicles on the predicted vehicles (that is, interactive information) is taken into account in the model, and the prediction effect of the model is improved.

This paper considers a method that combines the vehicle behavior recognition model with the trajectory prediction model; that is, the behavior of the vehicle is input into the trajectory prediction model as one of the input information, which can obviously improve the accuracy of trajectory prediction.

Based on the predicted horizontal and vertical velocity, an acceleration trajectory optimization algorithm is proposed, which obviously improves the accuracy and stability of the predicted trajectory.

2. Related Work

At present, in the aspect of vehicle trajectory prediction, the main methods are trajectory prediction based on physical motion model, trajectory prediction based on driving behavior, and trajectory prediction based on intention recognition. Gambs et al. [18] put forward the trajectory prediction method of high-order Markov model, which has high accuracy, but high computational overhead, so it is difficult to meet the real-time requirements of intelligent vehicles. Chandra et al. [10] proposed a new method based on the combination of graph analysis and deep learning to predict vehicle trajectories in urban traffic scenes, and they learned how to predict future trajectories and behaviors from the extracted vehicle trajectories. In order to reduce the error of long-term prediction (3–5 seconds) and improve the accuracy of prediction, spectral clustering regularization method was introduced and experiments were carried out on Argoverse, LYFT, and Apolloscape data sets. Deo and Trivedi [19] proposed an LSTM encoder-decoder model, which uses a convolutional social pool as an improvement to the social pool level to learn the interdependence in vehicle motion stably. In addition, based on the variability of trajectories, the model also outputs the multimodal prediction distribution of future trajectories and uses US-101 and I-80 sections of NGSIM data sets to evaluate the model. Chang et al. [20] proposed the Argoverse data set, which is designed to support self-driving vehicle perception data including 3D tracking and motion prediction. Argoverse data set includes sensor data collected by self-driving teams in Pittsburgh and Miami, as well as 3D tracking notes, extracting 300000 vehicle tracks and rich semantic maps. Using this data set can greatly reduce trajectory prediction errors. Chandra et al. [21] proposed an end-to-end vehicle trajectory prediction algorithm-RobustTP, which uses a tracking algorithm to obtain noise sensor input tracks from RGB cameras (whether stationary or moving) to predict vehicle trajectories in dense traffic and regards noise as a deviation from the real track. Firstly, the online motion model and the case segmentation algorithm based on deep learning are combined to calculate the trajectory. These noise tracks are trained by LSTM-CNN neural network structure, which simulates the interaction between different traffic objects in dense and nonuniform traffic. Chandra et al. [22] used a new LSTM-CNN hybrid network to model the interaction between different traffic objects for trajectory prediction, including buses, cars, scooters, bicycles, or pedestrians. Giuliar et al. [23] proposed a transformer network for pedestrian trajectory prediction and achieved good prediction results. Monti et al. [24] proposed a new recursive generation model to predict the trajectory of obstacles, which takes into account not only the future target of a single obstacle but also the interaction between different obstacles, and in this

model, a graph neural network based on double attention is used to collect the interactive information between different obstacles, which is combined with the possible future trajectory of the obstacles, and a good prediction effect is obtained in the urban scene. Mohamed [25] proposed the social space-time graph convolution neural network (Social-STGCNN), which uses the method of modeling interaction as a graph instead of the aggregation method. The experimental results show that the final displacement error (FDE) is 20% higher than the existing methods, the average displacement error (ADE) is 8.5 times higher, and the reasoning speed is 48 times faster. Li et al. [26] proposed a universal generative neural system (Social-WaGDAT) for trajectory prediction of various obstacles and evaluated the system on three common trajectory prediction data sets, and the experimental results show that the model has a good prediction effect in terms of prediction accuracy, in which the types of obstacles include pedestrians, bicycles, and vehicles. Hao et al. [27] proposed an end-to-end generative model called attention map encoder network (AMENet), which can accurately and truly realize multipath trajectory prediction and achieve good prediction results.

In order to make an accurate prediction of moving vehicle trajectory, a UB-LSTM vehicle trajectory prediction model combined with vehicle behavior recognition is proposed to obtain a smaller prediction error. Finally, based on the predicted horizontal and vertical velocity, an acceleration trajectory optimization algorithm is proposed.

3. Methods

Long short-term memory (LSTM) network [28] is a recurrent neural network (RNN) structure proposed by Hochreitrer and Schmidhuber in 1997. LSTM mainly solves the problems of gradient explosion and gradient vanishing of RNN [29]. LSTM mainly adds forgetting gate, input gate, and output gate on the basis of RNN to realize selective forgetting and memory of information, thus realizing the function of long-term memory. LSTM realizes the function of long-term memory through long-term memory. Because there is only simple multiplication and addition on the track of long-term memory, and there is no nonlinear operation, information flows more smoothly at different times, which can effectively restrain the problem of gradient dissipation of long-term memory.

The process of this method is to first establish a vehicle behavior recognition model to realize vehicle behavior recognition and finally input the behavior identified by the vehicle behavior recognition model and vehicle information into the established trajectory prediction model to predict the horizontal and vertical speed and coordinates in the next 3 seconds. The overall flow chart is shown in Figure 1. The concrete realization is to establish a separate vehicle behavior recognition model and a vehicle trajectory prediction model and train them separately, in which the training data of the vehicle trajectory model contains the marked vehicle behavior information. Then, the vehicle behavior recognition model and the trajectory prediction model are tested together, and finally, the acceleration trajectory optimization algorithm is used to generate a more accurate prediction trajectory.

The following will introduce in detail the vehicle behavior recognition model, trajectory prediction model, and acceleration trajectory optimization algorithm in turn.

3.1. Vehicle Behavior Recognition Model. The behavior recognition model proposed in this section is mainly used to identify the five behaviors of each vehicle trajectory, including going straight, turning left and right, and changing lanes. Not only the state of the vehicle can affect vehicle behavior but also the surrounding vehicles, pedestrians, bicycles and so on. For example, avoiding pedestrians and bicycles will obviously affect the vehicle behavior. Therefore, the input features of each trajectory point include the vehicle state (i.e., coordinates, velocity, and acceleration, etc.) and interactive information features. Before predicting the trajectory, the input characteristics of each trajectory point are input into the vehicle behavior recognition model to get the behavior characteristics of the vehicle. Then, the behavior characteristics, vehicle state characteristics, and interactive information characteristics are input into the trajectory prediction model together.

In this paper, a vehicle behavior recognition model is established based on a many-to-one LSTM classifier, as shown in Figure 2, where the seq_length is the number of features of the input data (i.e., the number of LSTM units); batch training is used to load data, and the batch_size is the load size; the embedding is the corresponding vector length of the input LSTM unit, where embedding = 1; the hidden_size is the number of LSTM hidden layer nodes; the output_size is the output category size; the n_layers is the number of hidden layers of LSTM. Finally, a full connection layer FC is used to make the classification, and only the last node $y_{\text{seq\_length}}$ is taken as the classification result. The output layer does not use activation functions. The Adam is used to update weights. The loss function is CrossEntropyLoss Function L, as defined in

$$L = -\sum_{i}^{N} y^{(i)} \log \widehat{y}^{(i)} + \left(1 - y^{(i)}\right) \log\left(1 - \widehat{y}^{(i)}\right), \tag{1}$$

where $y^{(i)}$ is the actual value and $\widehat{y}^{(i)}$ is the predicted value.

3.2. Vehicle Trajectory Prediction Model. The vehicle behavior recognition model is shown in Figure 3, which is mainly used to predict the vehicle trajectory. It can be seen that not only the state of the vehicle but also the surrounding vehicles, pedestrians, and bicycles can affect the trajectory of the vehicle and that the current behavior of the vehicle can also determine the future trajectory. For example, the trajectories of straight lines and turns are obviously different. The behavior characteristics can be marked in advance in the dataset, or they can be identified by the trained vehicle behavior recognition model. In this paper, the trajectory point of the output trajectory includes the characteristics of horizontal and longitudinal velocity and horizontal and longitudinal coordinates, that is, to predict the future

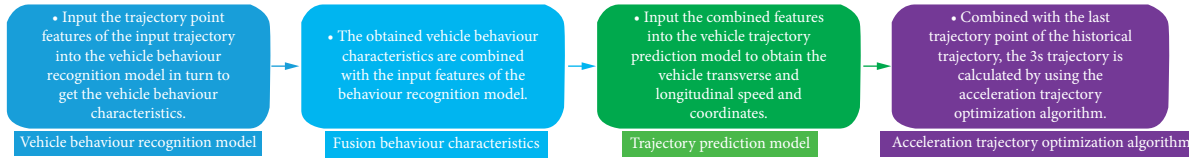


Figure 1: The basic flow of trajectory prediction.

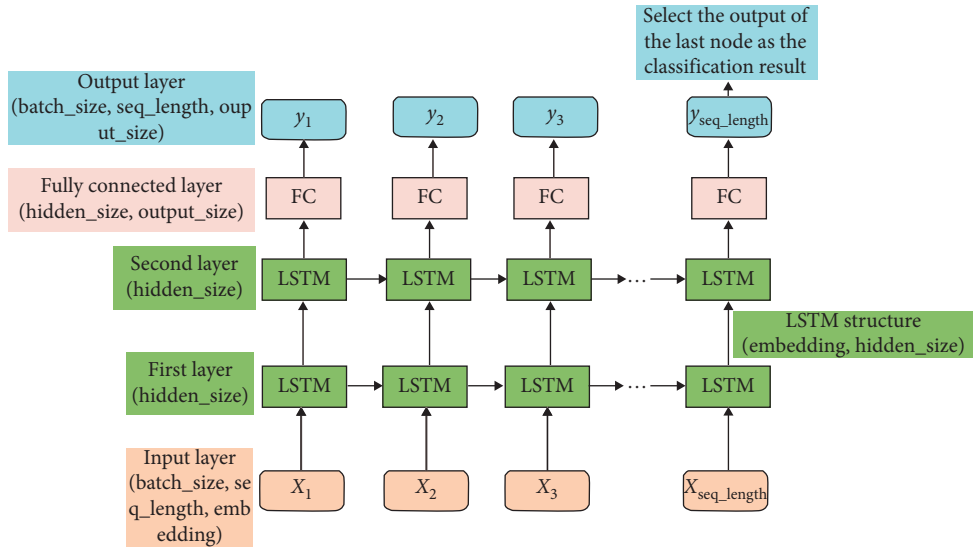


Figure 2: Vehicle behavior recognition model.

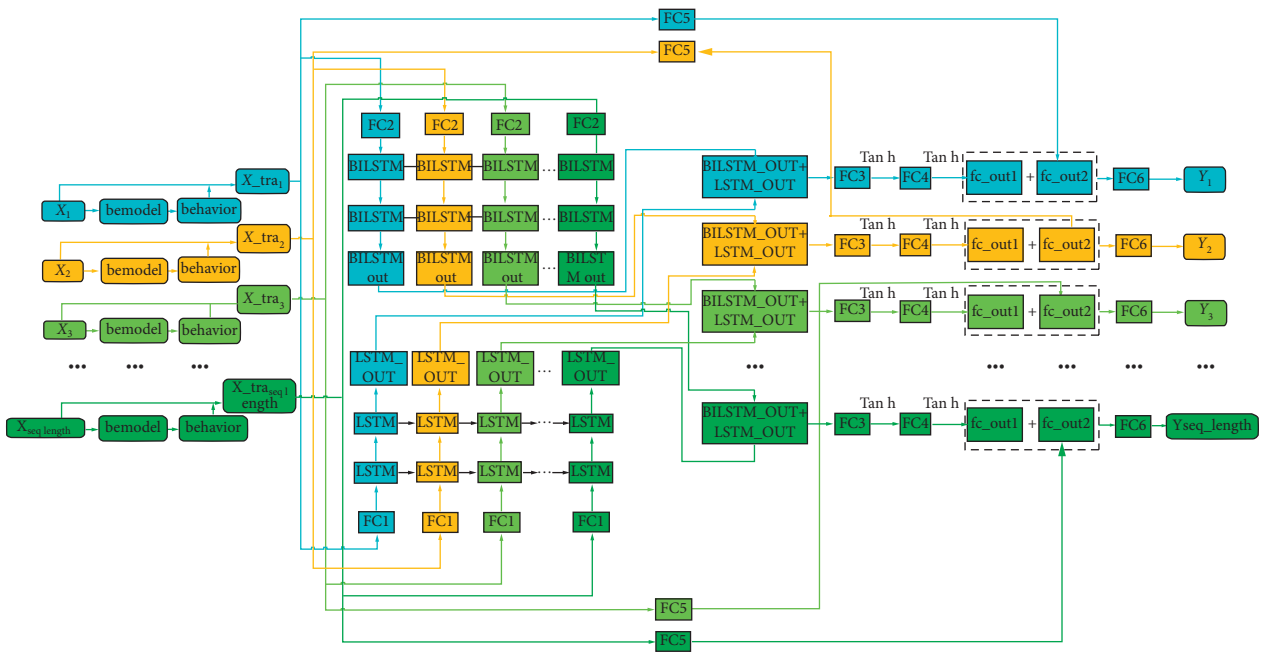


Figure 3: Trajectory prediction model.

horizontal and longitudinal speed and coordinates of the vehicle.

Compared with the traditional Unidirectional or Bidirectional LSTM trajectory prediction model, this paper uses UB-LSTM to establish the trajectory prediction model. Bidirectional LSTM can make use of not only the previous information but also some of the latter information to make the prediction results more accurate. Then, combining the prediction results of the two LSTM can further improve the overall prediction accuracy of the model. The results of the

(1) Input: x
(2) Output: out
(3) (batch_size, seq_length, n_feature-1) ⟵ x.shape;
(4) behavior ⟵ bemodel(x);
(5) x_tra ⟵ torch.cat(x, behavior);
(6) (batch_size, seq_length, n_feature) ⟵ x_tra.shape;
(7) Linear(n_feature, input_size) ⟵ FC1;
(8) LSTM(input_size, hidden_size, n_layers, bidirectional = False, dropout) ⟵ LSTM;
(9) Linear(n_feature, input_size) ⟵ FC2;
(10) LSTM(input_size, hidden_size//2, n_layers, bidirectional = True, dropout) ⟵ BILSTM;
(11) LSTM_OUT, hidden ⟵ LSTM(FC1(x_tra));
(12) BILSTM_OUT, hidden ⟵ BILSTM(FC2(x_tra));
(13) Linear(hidden_size, hidden_size ∗ 2) ⟵ FC3;
(14) Linear(hidden_size ∗ 2, hidden_size//2) ⟵ FC4;
(15) LSTM_FC_OUT ⟵Tanh(FC3(LSTM_OUT + BILSTM_OUT));
(16) fc_out1 ⟵ Tanh(FC4(LSTM_FC_OUT));
(17) Linear(n_feature, hidden_size//2) ⟵ FC5;
(18) fc_out2 ⟵ FC5(x_tra);
(19) Linear(hidden_size//2, output_size) ⟵FC6;
(20) out ⟵FC6(fc_out1 + fc_out2);
(21) end;
(22) **Return** out;

Algorithm 1: Trajectory prediction model.

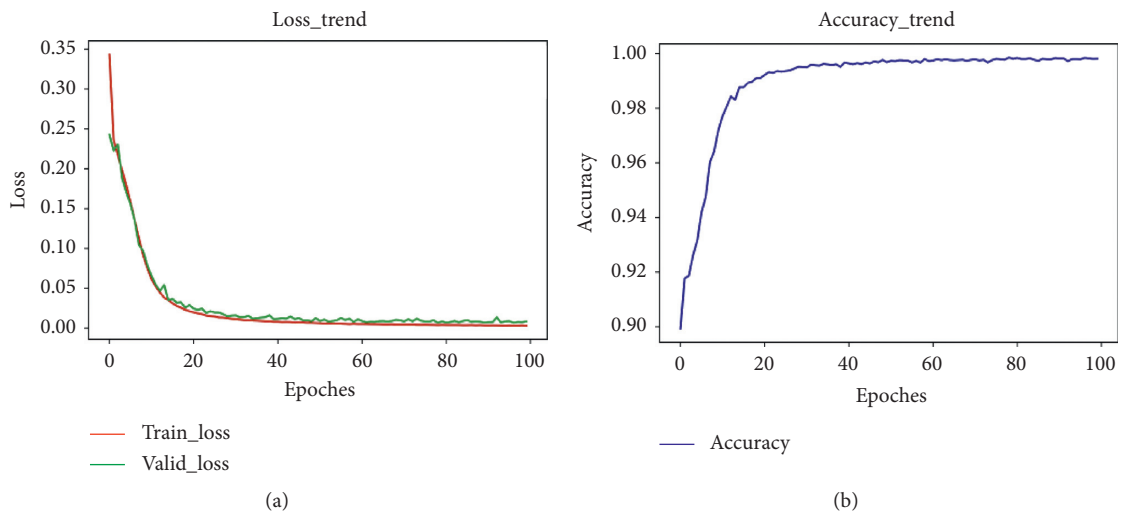


Figure 4: Training process of behavior recognition model. (a) Loss change of behavior recognition model. (b) Accuracy change of behavior recognition model.

Table 1: Test results of the behavior recognition model.

	Left_turn	Left_change	Right_turn	Right_change	Keep	All	Test_loss
Correct_n/ALL	9017/9042	8635/8691	1708/1720	1524/1552	152166/152195	173050/17320	0.0026
Accuracy (%)	99.72	99.36	99.30	98.20	99.98	99.91	

addition of the two LSTM output layers are input into the two full connection layers in turn. Because the value range of the Tanh function is from −1 to 1, the range of the output value is limited, so the input data is directly input into a full connection layer to get an output data and then process the output value by the Tanh activation function. Finally, a full connection layer is used to get the output result. The pseudo-code of the trajectory prediction model is as follows: (Algorithm 1).

The bemodel is the vehicle behavior recognition model mentioned in Part A of Section 3. The seq_length is the

number of input trajectory points. The n_feature is the number of features contained in the input trajectory points. The input_size is the length of the vector input into LSTM cells, the hidden_size is the number of hidden layer nodes of LSTM, and the n_layers is the number of hidden layers. Using dropout method to prevent overfitting, the dropout is in the discarding rate. The behavior is the vehicle behavior identified by the vehicle behavior recognition model, and bidirectional indicates whether the LSTM is a bidirectional LSTM. The Tanh is a hyperbolic tangent activation function. The loss function uses the mean square error loss function (MSELoss), as shown in

$$\text{MSELoss} = \frac{1}{2n}\sum_{i=1}^{n}\left(y_i - \widehat{y}_i\right)^2, \tag{2}$$

where n represents the total number of variables, y_i represents the actual value, and $\widehat{y}_i$ represents the predicted value.

3.3. Acceleration Optimization Trajectory Algorithm. Based on the trajectory prediction model, an acceleration trajectory optimization algorithm is proposed in this paper. This algorithm refers to the acceleration and displacement formula in physics, as shown in

$$s = \frac{\left(v^2 - v_0^2\right)}{2a}, \tag{3}$$

where s is the displacement, v is the final velocity, v_0 is the initial velocity, and a is the acceleration at v_0. The transverse and longitudinal acceleration is calculated according to

$$\begin{cases} a_{x_{t-1}} = \dfrac{\left(v_{x_t} - v_{x_{t-1}}\right)}{\Delta t}, \\ a_{y_{t-1}} = \dfrac{\left(v_{y_t} - v_{y_{t-1}}\right)}{\Delta t}, \end{cases} \tag{4}$$

where Δt represents the interval time between two points, v_x and v_y represent the horizontal and longitudinal speeds of the vehicle predicted by the trajectory prediction model, and t represents the time of the trajectory point. The horizontal and longitudinal displacements are calculated according to

$$\begin{cases} s_{x_{t-1}} = \dfrac{v_{x_t}^2 - v_{x_{t-1}}^2}{2a_{x_{t-1}}}, \\ s_{y_{t-1}} = \dfrac{v_{y_t}^2 - v_{y_{t-1}}^2}{2a_{y_{t-1}}}. \end{cases} \tag{5}$$

The coordinate values are calculated according to

$$\begin{cases} x_t = x_{t-1} + s_{x_{t-1}}, \\ y_t = y_{t-1} + s_{y_{t-1}}, \end{cases} \tag{6}$$

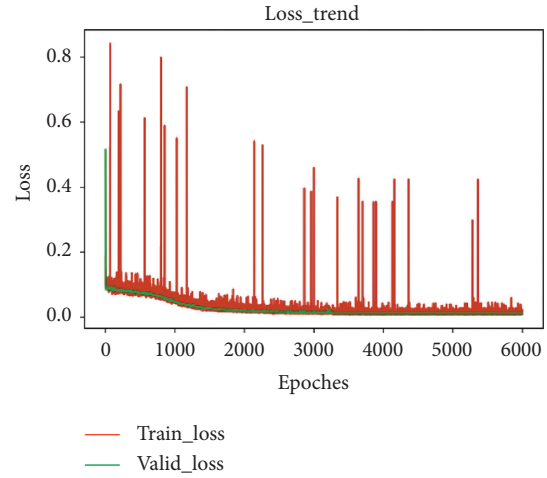


Figure 5: Training process of trajectory prediction model without vehicle behavior.

where x_{t-1} and y_{t-1} represent the coordinates of the previous trajectory point, and x_t and y_t represent the coordinates of the trajectory point. In this paper, the last point of the historical trajectory point is taken as the starting trajectory point and then calculated in turn until the coordinates of all the predicted trajectory points are calculated. In this paper, the time interval of trajectory points is 0.1 s.

4. Experimental Validation

4.1. Experiment Platform. In this paper, the experiment is carried out on the ubuntu16.04 system, the GPU is Tesla V100-PCIE-32GB, and the model is built on Jupyter Notebook based on PyTorch.

4.2. Data Sources and Preprocessing. This paper uses the NGSIM data set for experiments. The NGSIM data set is derived from the Next Generation Simulation (NGSIM) program initiated by the Federal Highway Administration of the United States. The sampling frequency is 10 Hz and records information including vehicle coordinates, speed, acceleration, vehicle type, and lane number [30].

4.2.1. Calculate Angle, VEL_X, VEL_Y, and Behavior. The vehicle heading angle θ is calculated using

$$\theta = \arctan\left(\frac{x^i - x^{i-1}}{y^i - y^{i-1}}\right), \tag{7}$$

where (x^i, y^i) represents the coordinates of the vehicle at i time, and (x^{i-1}, y^{i-1}) indicates the coordinates of the vehicle at $i-1$ time.

According to the change rate of heading angle θ between the two trajectory points of the vehicle, ω marks the vehicle behavior Label, as in formula (8). A total of five behaviors are marked, including going straight, turning left and right, and changing left and right lanes, which are represented by 0, 1, 2, 3, and 4, respectively:

$$\omega = \frac{\theta_i - \theta_{i-1}}{\Delta t}, \tag{8}$$

where Δt represents the interval time between two points.

The transverse and longitudinal velocities Vel_x and Vel_y of the vehicle are calculated according to

$$\begin{cases} v_{x_t} = \dfrac{(x_t - x_{t-1})}{\Delta t}, \\ v_{\text{avg}_{x_t}} = \dfrac{\left(v_{x_t} + v_{x_{t+1}}\right)}{2}, \\ v_{\text{begin}} = 2v_{x_0} - v_{\text{avg}_{x_0}}, \\ v_{\text{end}} = 2v_{x_{\text{end}}} - v_{\text{avg}_{x_{\text{end}}}}, \\ v_{x_0} = v_{\text{begin}}, \\ v_{x_1} = v_{\text{avg}_{x_0}}, \\ v_{x_2} = v_{\text{avg}_{x_1}}, \\ \vdots \\ v_{x_{\text{end}}} = v_{\text{end}}, \end{cases} \tag{9}$$

$$\begin{cases} v_{y_t} = \dfrac{(y_t - y_{t-1})}{\Delta t}, \\ v_{\text{avg}_{y_t}} = \dfrac{\left(v_{y_t} + v_{y_{t+1}}\right)}{2}, \\ v_{\text{begin}} = 2v_{y_0} - v_{\text{avg}_{y_0}}, \\ v_{\text{end}} = 2v_{y_{\text{end}}} - v_{\text{avg}_{y_{\text{end}}}}, \\ v_{y_0} = v_{\text{begin}}, \\ v_{y_1} = v_{\text{avg}_{y_0}}, \\ v_{y_2} = v_{\text{avg}_{y_1}}, \\ \vdots \\ v_{y_{\text{end}}} = v_{\text{end}}, \end{cases} \tag{10}$$

where Δt denotes the interval time between two points, (x, y) represents the coordinate of the trajectory point, t represents the time of the occurrence of the trajectory point, and end represents the last trajectory point of the vehicle.

4.2.2. Acquisition of Interactive Information. The main purpose is to obtain the vehicles that may exist in the left-top, left-bottom, middle-top, middle-bottom, right-top, and right-bottom positions around the vehicle, which are represented as L_Top, L_Bot, C_Top, C_Bot, R_Top, and R_Bot, respectively. Firstly, the lateral coordinate difference dis_x and the longitudinal coordinate difference dis_y between the surrounding vehicle and the predicted trajectory vehicle at a certain time are calculated, and then the position of the surrounding vehicle relative to the predicted trajectory vehicle is judged according to

$$\text{relative\_position} = \begin{cases} \text{L\_Top} & -5 < \text{dis\_}x < -1, 0 < \text{dis\_}y < 5, \\ \text{L\_Bot} & -5 < \text{dis\_}x < -1, -5 < \text{dis\_}y < 0, \\ \text{C\_Top} & -1 < \text{dis\_}x < 1, 0 < \text{dis\_}y < 5, \\ \text{C\_Bot} & -1 < \text{dis\_}x < 1, -5 < \text{dis\_}y < 0, \\ \text{R\_Top} & 1 < \text{dis\_}x < 5, 0 < \text{dis\_}y < 5, \\ \text{R\_Bot} & 1 < \text{dis\_}x < 5, -5 < \text{dis\_}y < 0. \end{cases} \tag{11}$$

Then, we record the id, relative distance, speed, acceleration, heading angle, and behavior of the vehicle that meets the conditions. If there is no vehicle in a certain position, the above information is all set to 0, so the interactive information of the vehicle is obtained.

4.2.3. Generate Trajectory and Behavior Dataset. In the experiment, the Savizkg-Golag smoothing algorithm is firstly used to smooth the coordinates of vehicle trajectory points to eliminate the noise, then remove the vehicles with less than 130 trajectory points or two trajectory points with a distance of more than 5 meters, and at the same time, limit the maximum number of trajectories of each vehicle to 61. By doing so, the number of trajectories of each vehicle will keep consistent. Finally, a total of 17,320 trajectory data are generated, and each trajectory contains 100 trajectory points, including a total of 321 vehicles. The trajectory is standardized using

$$x = \frac{\widehat{x} - \text{mean}}{\text{std}}, \tag{12}$$

where $\widehat{x}$ represents a single value in each column, x represents the changed value, mean represents the average of each column, and std represents the standard deviation.

In the experiment, a total of 26 features are selected as the input trajectory points. At the same time, a total of four features including the vehicle horizontal and longitudinal velocity and coordinates of the trajectory point 3 seconds behind the input trajectory point are selected as the output trajectory point.

The vehicle behavior data is first restored according to equation (13), then the vehicle behavior feature of each input trajectory point is separated as a tag, and the rest of 25 features are used as input features:

$$x = \widehat{x} * \text{std} + \text{mean}. \tag{13}$$

In order to better observe the training process, the vehicle trajectory data is divided into a training set, verification

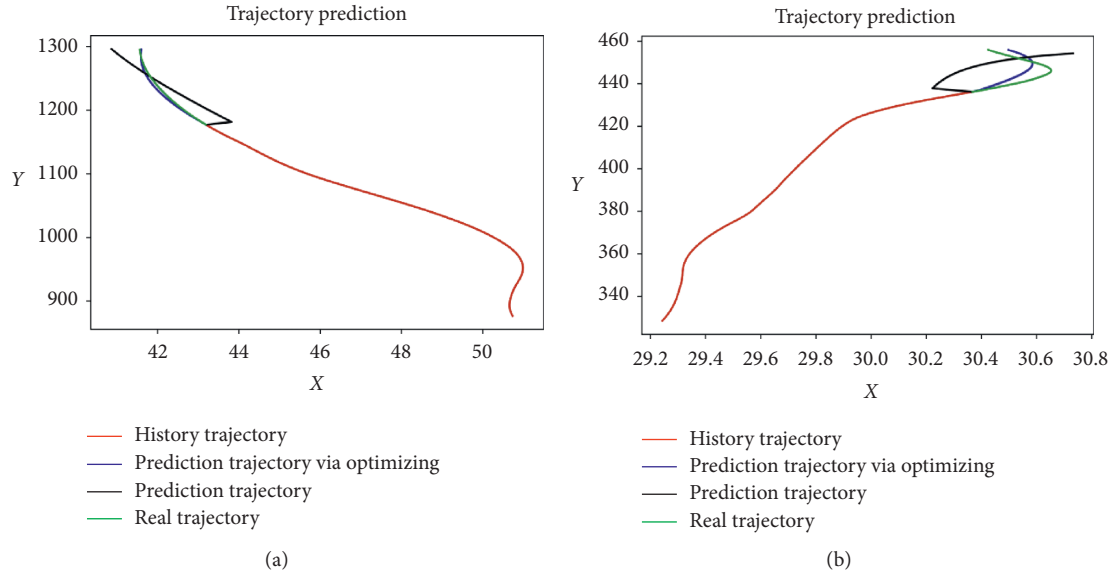


Figure 6: Trajectory prediction results without vehicle behavior. (a) Prediction trajectory 1. (b) Prediction trajectory 2.

set, and test set according to the proportion of 8 : 1 : 1. The training set is used to train the model, the verification set is used to monitor the training process in real time, and the test set is used to evaluate the effect of the model after the training.

4.3. Vehicle Behavior Recognition Model Experiment. Set batch_size = 100, seq_length = 25, embedding = 1, hidden_size = 256, n_layers = 2, output_size = 5, and learning rate = 0.0001. Only the model with minimum validation loss is saved with a total of 100 iterations. The training process is shown in Figure 4.

The training time is 3.943 hours, and the minimum verification loss is 0.0066. The accuracy of vehicle behavior recognition is the ratio of the correct recognition number of each vehicle behavior to the total number of each vehicle behavior. The test results are shown in Table 1. As can be seen from the test results, the recognition accuracy of each behavior of the vehicle is higher than 98%.

4.4. Trajectory Prediction Model Experiment. In order to verify the influence of vehicle behavior, the trajectory prediction accuracy is compared with and without considering the vehicle behavior in this study. In the experiments, the trajectory data is restored using equation (13). In order to reflect the superiority of the acceleration trajectory optimization algorithm proposed in this paper, the trajectory generated by the coordinate of the predicted trajectory point is compared with that generated by the trajectory optimization algorithm.

4.4.1. Trajectory Prediction Model Experiment without Vehicle Behavior. Set batch_size = 128, seq_length = 100, n_feature = 25, input_size = 256, hidden_size = 256, n_layers = 2, output_size = 4, dropout = 0.5, and we used the

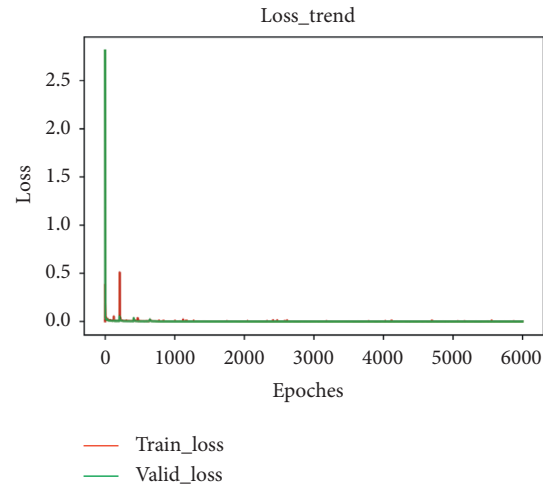


Figure 7: Training process of the trajectory prediction model considering behavior.

ReduceLROnPlateau method to adjust the learning rate. The initial learning_rate is 0.001. When the verification loss does not decrease for 20 iterations, the learning rate is adjusted to 10% of the existing rate. Only the model with minimum validation loss is saved, within a total of 1,000 iterations. The training process is shown in Figure 5, where vehicle behavior is not considered here.

The training time is 3.944 hours, and the minimum verification loss is 0.0114. It can be seen from Figure 5 that the loss decreases slowly during the training. In the testing process, the test loss is 0.0115, the MSE between the predicted trajectory and the actual trajectory is 4.3, and the MSE produced by the optimization algorithm is 2.4. Figure 6 shows the predicted results, where the axis x represents the lateral coordinates of the predicted trajectory point, and the axis Y represents the longitudinal coordinates of the predicted trajectory point. As can be seen from Figure 6, the

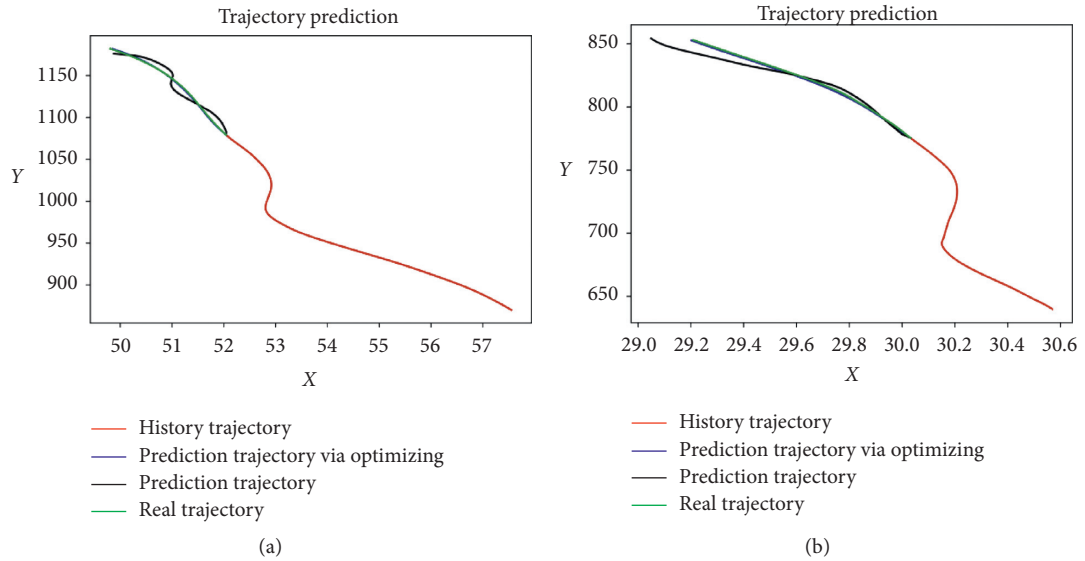


Figure 8: Trajectory prediction results using the two models in the series. (a) Prediction trajectory 1. (b) Prediction trajectory 2.

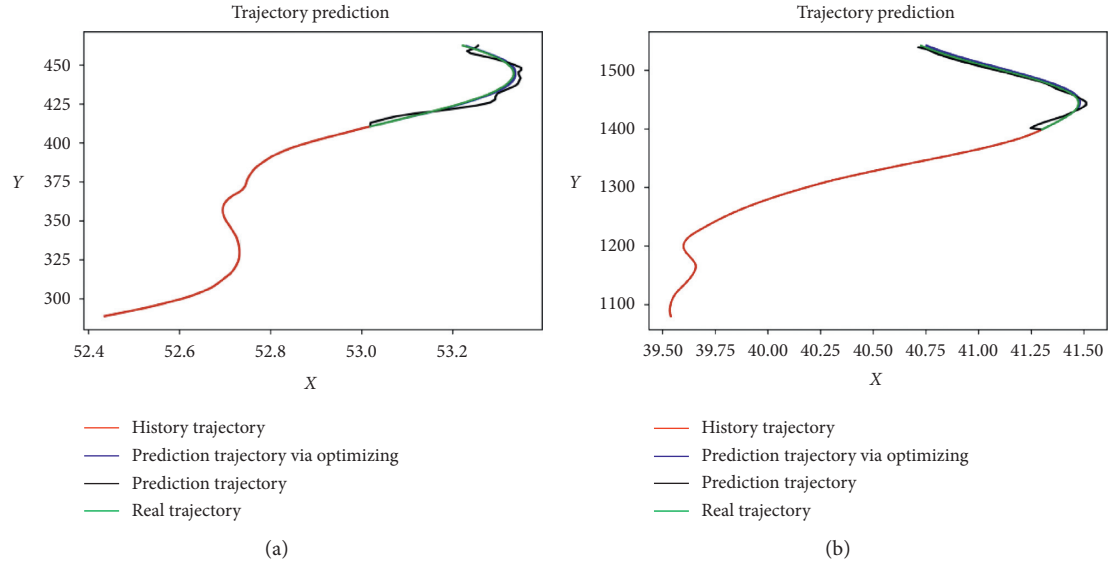


Figure 9: Trajectory prediction results using the two models in parallel. (a) Prediction trajectory 1. (b) Prediction trajectory 2.

trajectory generated by the optimization algorithm is very close to the actual trajectory; but the prediction performance without the optimization algorithm is far away from satisfactory.

4.4.2. Trajectory Prediction Model Experiment Combined with Vehicle Behavior. Set batch_size = 128, seq_length = 100, *n*_feature = 26, input_size = 256, hidden_size = 256, *n*_layers = 2, output_size = 4, dropout = 0.5, and initial Learning_rate = 0.001. Figure 7 depicts the training result.

The training time is 7.12 hours, and the minimum verification loss is 0.00041732. In order to test the fusion effect of the vehicle behavior model and the trajectory prediction model, first of all, the trajectory prediction model is used alone after the behavior recognition model, and then two models are used in parallel.

The vehicle trajectory prediction model predicts the horizontal and longitudinal speed and coordinates of the vehicle and uses the transverse and longitudinal velocity and the last trajectory point of the historical trajectory to realize the acceleration trajectory optimization. First of all, the experiment is carried out without using the vehicle behavior recognition model. The test loss is 0.000486. The MSE between the predicted trajectory and the actual trajectory is 1.361. The MSE generated by the optimization algorithm is 0.122. Figure 8 shows the predicted results, where the axis x represents the lateral coordinates of the predicted trajectory point, and the axis Y represents the longitudinal coordinates of the predicted trajectory point.

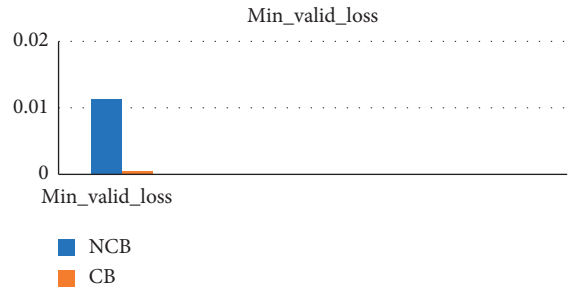


Figure 10: Comparison of minimum verification loss.

Comparing Figures 6 and 8, one can find that the prediction effect is obviously improved; thus, it is very necessary to consider vehicle behavior in the prediction model. The prediction trajectory obtained by the optimization algorithm is more consistent with the actual trajectory, which indicates that the usage of the optimization algorithm makes the prediction performance stable.

When the behavior recognition model and the trajectory prediction model are integrated, the test loss is 0.000497, the recognition accuracy of the behavior recognition model is 99.9%, the MSE without the optimization algorithm is 1.36, and the MSE with the optimization algorithm is 0.124. It can be found that the MSE and test losses after fusion are slightly higher than those of the individual use of the prediction model in Figure 8. Figure 9 shows the predicted results, where the axis x represents the lateral coordinates of the predicted trajectory point, and the axis Y represents the longitudinal coordinates of the predicted trajectory point. It can be seen from Figure 9 that the trajectory prediction effect of the proposed method is good, which is basically consistent with the actual trajectory, and is stable, without large fluctuations, and basically achieves the expected effect. It shows that the trajectory prediction method proposed in this paper has high prediction accuracy and good stability.

4.5. Discussion. As can be seen from Figures 5, 7, and 8, the trajectory generated by the optimization algorithm is much better, more reasonable, and more consistent with the actual trajectory than the trajectory directly predicted in this paper. It shows that the trajectory optimization algorithm proposed in this paper contributes a very good improvement effect.

In order to examine the behavior recognition effect, the following comparisons are further conducted. Herein, the test without considering behavior recognition is abbreviated as NCB; the test with considering behavior recognition is abbreviated as CB; the alone test model considering vehicle behavior is abbreviated as CB_A; the fusion test model considering vehicle behavior is abbreviated as CB_F; the test based on the coordinate method without considering behavior recognition is abbreviated as NCB_CB; the test based on optimization algorithm without considering behavior recognition is abbreviated as NCB_OB; the alone test based on coordinate method considering vehicle behavior recognition is abbreviated as CB_A_CB; the alone test based on optimization algorithm considering vehicle behavior recognition is abbreviated as CB_A_OB; the fusion test based on coordinate method considering vehicle behavior recognition is abbreviated as CB_F_CB; the fusion test based on optimization algorithm considering vehicle behavior recognition is abbreviated as CB_F_OB. The comparison results are demonstrated as follows.

Table 2: Comparison of minimum verification loss.

	NCB	CB
Min_valid_loss	0.0114	0.000417

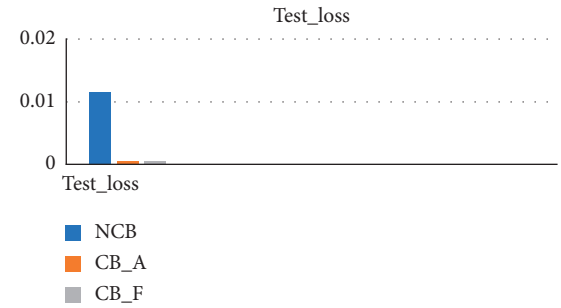


Figure 11: Comparison of test losses.

Table 3: Comparison of test losses.

	NCB	CB_A	CB_F
Test_loss	0.0115	0.000486	0.000497

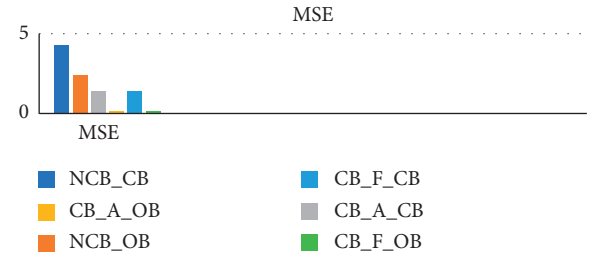


Figure 12: MSE comparison.

4.5.1. Minimum Verification Loss. The minimum verification loss is shown in Figure 10 and Table 2. As can be seen, the minimum verification loss with consideration of behavior recognition is much smaller than that without considering it, which indicates that it is necessary to consider vehicle behavior.

4.5.2. Test Loss. The test losses are shown in Figure 11 and Table 3. It can be seen that the test loss of the model without considering the behavior recognition is the largest and the test loss of the individual test model is the smallest when considering the behavior, which shows the importance of vehicle behavior recognition. Hence, it suggests that in the real world application, it is very important to use a vehicle behavior recognition model to identify vehicle behavior before predicting the trajectory.

Table 4: MSE comparison.

	NCB_CB	CB_A_CB	CB_F_CB	NCB_OB	CB_A_OB	CB_F_OB
MSE	4.3	1.361	1.36	2.4	0.122	0.124

4.5.3. MSE Comparison. The MSE size under each method is shown in Figure 12 and Table 4. It can be seen from the results that the MSE generated by the optimization algorithm is much lower than that of the coordinate method, which means that the optimization algorithm can greatly improve the accuracy of trajectory prediction. It can be also seen that the MSE of the model considering behavior recognition is always lower than that not considering vehicle behavior. In addition, by considering vehicle behavior, the MSE produced by the series configuration of the recognition model and prediction model is very close to that of the parallel configuration of the two models, which shows that combining the two models is feasible.

To sum up, the UB-LSTM based trajectory prediction model, which combines the vehicle behavior recognition and acceleration trajectory optimization algorithm, is able to predict the trajectory accurately and stably.

5. Conclusions and Future Plan

This paper proposes a UB-LSTM trajectory prediction model, which inputs vehicle state information, vehicle behavior information, and interactive information into the trajectory prediction model to predict the vehicle transverse and longitudinal speed and coordinates. A vehicle behavior recognition model is trained to make a prediction combined with the vehicle trajectory prediction model. An acceleration trajectory optimization algorithm is proposed to improve the trajectory prediction accuracy. Experimental results show that the model test loss obtained by the proposed UB-LSTM is 0.000497, and the prediction MSE is 0.124, which is 97.2% lower than that without considering the vehicle behavior information. As a result, the proposed UB-LSTM method is suitable for practical usage.

In the next step, we will consider predicting the trajectory for a longer time and collect more experimental data to improve the model training effect. Although a real car is not available at this moment, the real vehicle experiment will be considered in the near future. The NGSIM dataset used in this experiment records the time stamp, id, coordinates, speed, and acceleration of the vehicles. However, a LIDAR can also provide the time stamp, id, coordinates, velocity, and acceleration of the surrounding obstacles, and the types of obstacles include vehicles, pedestrians, and bicycles. Thus, the interactive information provided by a LIDAR is more comprehensive. Moreover, environmental information can be obtained through high-precision maps, so more input information can be used in the process of real cars. Based on the above characteristics, the real vehicle dataset, similar to NGSIM dataset, can be collected by a LIDAR to carry out real car experiments. The intelligent car uses the C++ language to realize automatic driving, while the training model in this paper is realized by Python language based on PyTorch, so we will consider using LibTorch to migrate the model to the intelligent vehicle. The basic process is as follows: the LIDAR is firstly used to collect the vehicle trajectory dataset, then train the UB-LSTM model, and lastly, transfer the trained model to the intelligent vehicle for real vehicle experiments. However, the difference between the real vehicle experiment and the simulation is that the data processing should be carried out in real time. This will be carried out in the future.

Compared with the physical model method, the disadvantage of the proposed method is that the calculation speed is slower due to a large number of calculations, and it is basically necessary to use GPU to meet the real-time requirements. In the next work, we will try our best to improve the real-time performance of the model.

Data Availability

The code of this article has been uploaded to https://github.com/xhpxiaohaipeng/Vehicle-trajectory-Prediction-combined-with-vehicle-behavior-recognition-.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This research was supported by the National Key Research and Development Program of China (2016YFB0100903), NSFC (No. 51979261), Fundamental Research Funds for the Central Universities (No. 201941008), Applied Fundamental Research Project of Qingdao (No. 2019-9-1-14-jch), and Australia Research Council (No. DE190100931).

References

[1] C. Badue, R. Guidolini, R. V. Carneiro et al., "A survey," 2019, http://arxiv.org/abs/1901.04407.

[2] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planningand-decision-makingforautonomousvehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 187–210, 2018.

[3] Y. Liao, S. E. Li, W. Wang, Y. Wang, G. Li, and B. Cheng, "Detection of driver cognitive distraction: a comparison study of stop-controlled intersection and speed-limited Highway," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1628–1637, 2016.

[4] Y. Hu, W. Zhan, and T. Masayoshi, "Probabilistic prediction of vehicle semantic intention and motion," in *Proceedings of the Intelligent Vehicles Symposiun*, April 2018.

[5] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, "Traphic:Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8483–8492, Long Beach, CA, USA, June 2019.

[6] RohanChandra, UttaranBhattacharya, X. Li TrishaMittal, A. Bera, and D. Manocha, "Graphrqi:classifying driver behaviors using graph spectrums," 2019, http://arxiv.org/abs/1910.00049.

[7] R. Chandra, U. Bhattacharya, C. Roncal, A. Bera, and D. Manocha, "Robusttp : end-toend trajectory prediction for heterogeneous road-agents in dense traffic with noisy sensor inputs," 2019, http://arxiv.org/abs/1907.08752.

[8] E. Cheung, A. Bera, E. Kubin, K. Gray, and D. Manocha, "Identifying driver behaviors using trajectory features for vehicle navigation," in *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3445–3452, IEEE, Madrid, Spain, October 2018.

[9] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," 2018, http://arxiv.org/abs/1805.06771.

[10] R. Chandra, T. Guan, S. Panuganti et al., "Forecasting Trajectory and Behavior of Road-Agents Using Spectral Clustering in Graph-LSTMs," 2019, http://arxiv.org/abs/1912.01118.

[11] J. Firl, H. St¨ubing, S. A. Huss, and C. Stiller, "Predictive maneuver evaluation for enhancement of car-to-x mobility data," in *Proceedings of the Intelligent Vehicles Symposium (IV)*, pp. 558–564, IEEE, Alcala de Henares, Spain, June 2012.

[12] St´ephanie Lef`evre, C. Laugier, and J. Ibañez Guzm´an, "Exploiting map information for driver intention estimation at road intersections," in *Proceedings of the Intelligent Vehicles Symposium (IV)*, pp. 583–588, IEEE, Baden-Baden, Germany, June 2011.

[13] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[14] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proceedings of the INTERSPEECH 2012 13th Annual Conference of the International Speech Communication Association*, Portland, OR, USA, September 2012.

[15] Z. Wu, O. Watts, and S. King, "Merlin: an open source neural network speech synthesis system," in *Proceedings of the 9th ISCA Speech Synthesis Workshop*, Sunnyvale, CA, USA, September 2016.

[16] A. L. Bianne, F. Menasri, R. Al-Hajj, C. Mokbel, C. Kermorvant, and L. Likforman-Sulem, "Dynamic and contextual information in HMM modeling for handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 10, pp. 2066–2080, 2011.

[17] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of the Eleventh Annual Conference of the International Speech Communication Association*, Makuhari, Chiba, Japan, September 2010.

[18] GAMBS Sebastien, K. Marc-Oliver, MIGUEL Nunez del et al., "Next place prediction using mobility Markov chains," in *Proceedings of the 1st Workshop on Measurement, Privacy, and Mobility*, vol. 3, pp. 1–6, New York, NY, USA, 2012.

[19] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Salt Lake City, UT, USA, June 2018.

[20] M.-F. Chang, J. Lambert, P. Sangkloy et al., "Argoverse: 3D tracking and forecasting with rich maps," in *Proceedings of the CVPR (Computer Vision and Pattern Recognition)*, Long Beach, CA, USA, June 2019.

[21] R. Chandra, U. Bhattacharya, and D. Manocha, "RobustTP: end-to-End trajectory prediction for heterogeneous road-agents in dense traffic with noisy sensor inputs," in *Proceedings of the ACM Computer Science in Cars Symposium on-CSCS '19*, Kaiserslautern, Germany, May 2019.

[22] R. Chandra, U. Bhattacharya, and D. Manocha, "TraPHic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in *Proceedings of the CVPR (Computer Vision and Pattern Recognition)*, Long Beach, CA, USA, June 2019.

[23] F. Giuliari, I. Hasan, M. Cristani, and F. Galasso, "Transformer networks for trajectory forecasting," 2020, http://arxiv.org/abs/2003.08111.

[24] A. Monti, A. Bertugli, S. Calderara, and R. Cucchiara, "DAG-Net: double attentive graph neural network for trajectory forecasting," 2020, http://arxiv.org/abs/2005.12661.

[25] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel, "Social-STGCNN: a social spatio-temporal graph convolutional neural network for human trajectory prediction," in *Proceedings of the the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 2020.

[26] J. Li, H. Ma, Z. Zhang, and M. Tomizuka, "Social-WaGDAT: interaction-aware trajectory prediction via wasserstein graphdouble-attention network," 2020, http://arxiv.org/abs/2002.06241.

[27] H. Cheng, W. Liao, M. Y. Yang, B. Rosenhahn, and M. Sester, "AMENet: attentive maps encoder network for trajectory prediction," 2020, http://arxiv.org/abs/2006.08264.

[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[29] S. Hochreiter, *Untersuchungen zu dynamischen neuronalen Netzen. Diploma*, Technische Universität München, , June 1991.

[30] Federal Highway Administration, "NG-SIM-Next Generation Simulation[EB/OL]," https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm.