Received 24 January 2025, accepted 7 February 2025, date of publication 11 February 2025, date of current version 3 March 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3540950

RESEARCH ARTICLE

Real-Time Moving Vehicle Counting and Speed Estimation Toward Efficient Traffic Surveillance

DANESH SHOKRI[1,2], **CHRISTIAN LAROUCHE**[1,2], **AND SAEID HOMAYOUNI**[2,3], **(Senior Member, IEEE)**
[1]Département des Sciences Géomatiques, Université Laval, Québec City, QC G1V 0A6, Canada
[2]Centre de Recherche en Données et Intelligence Géospatiales (CRDIG), Université Laval, Québec City, QC G1V 0A6, Canada
[3]Centre Eau Terre Environnement, Institut National de la Recherche Scientifique, Québec City, QC G1K 9A9, Canada

Corresponding author: Danesh Shokri (danesh.shokri.1@ulaval.ca)

This work was supported by Mitacs under Grant IT30935.

ABSTRACT This paper presents a Spatial-Temporal Diagram (STD) algorithm for real-time vehicle counting and speed estimation in camera-based traffic surveillance. The algorithm consists of four main steps: STD graph generation highlighting vehicles as peaks, graph refinement using Gaussian Mixture Model likelihood optimization, peak detection through RANdom SAmple Consensus model fitting, and traffic parameter computation. Testing on over 11 million video frames from diverse sources, including 511 highway cameras, NVIDIA AI City Challenge, and Next Generation Simulation datasets, demonstrated the algorithm's robustness across varying illumination, weather conditions, and road infrastructures. The algorithm achieved average accuracies of 95.4%, 96.9%, and 96.1% for Precision, Recall, and F1-Score, respectively, outperforming traditional deep learning methods while requiring less computational resources.

INDEX TERMS Spatial-temporal diagram (STD), intelligent transportation system (ITS), vehicle counting, speed estimation, traffic flow, radiometric camera.

I. INTRODUCTION

Smart City technologies are essential for addressing the challenges of urbanization, with 70% of the global population projected to live in cities by 2050 [1]. Transportation improvements are a major focus of smart city initiatives, such as the US DOT's "Smart City Challenge" in 2015, which encouraged innovative solutions to urban transportation issues [2]. Data collection methods in smart cities are categorized into sensor-based methods, like LiDAR and radar, and vision-based methods using cameras [3], [4], [5]. Cameras are preferred due to their flexibility, contextual data capture, and wide coverage, enabling applications like vehicle counting, speed estimation, and traffic flow monitoring [6]. This study utilizes highway video observations to estimate traffic parameters such as volume, density, and flow, but challenges like lighting, weather conditions, and camera quality complicate this task. While high-resolution cameras offer superior data quality, their cost and computational demands pose practical limitations. Therefore, an effective algorithm must optimize cost, accuracy, and adaptability to varying conditions [7].

Since numerous organizations already have low-resolution camera networks for manual traffic monitoring, creating computer vision techniques that can use these existing resources makes economic sense. We can extract fresh data from these installed resources by applying centralized computer vision systems to these established camera networks. This approach allows us to continue using thousands of cameras that were deployed in the past, thereby saving on replacement expenses. Moreover, such a system reduces dependence on camera suppliers, ensuring consistent data collection regardless of the camera types used. For example, "511 highway camera" is a free traffic information service available in the United States and Canada [8]. It started in 1995 at the University of North Dakota, initially covering North and South Dakota highways. Over time, it expanded and received support from the Federal Highway Administration. As of 2021, 37 states have implemented 511 public traffic cameras. These cameras vary in number, quality, and location, mainly in high-traffic

The associate editor coordinating the review of this manuscript and approving it for publication was Jon Atli Benediktsson.

areas. They are valuable for traffic data analysis, but most are used for monitoring and information sharing rather than innovative data collection.

Previous research on vehicle counting and speed estimation is categorized into three phases: detection, tracking, and speed estimation. Detection focuses on identifying vehicles within video frames, while tracking ensures the same vehicle is consistently followed across consecutive frames. Speed estimation maps image space to real-world coordinates. Advanced deep-learning methods like Single Shot Detector (SSD), You Look Only Once (YOLO), and Region-based Convolutional Neural Networks (RCNN) have significantly improved vehicle detection by using bounding boxes to identify vehicles [9], [10]. One-step methods like SSD and YOLO are faster but less precise than two-step approaches like RCNN, which offer better classification but require more computation time [11]. Studies on highway videos evaluated these algorithms under various conditions, including low light, adverse weather, and diverse camera angles [12]. YOLOv7 demonstrated superior performance in detection and localization, while YOLOv8 excelled in processing speed. However, real-time applications remain challenging without GPUs. The study also revealed that RCNN and SSD are less effective in highway scenarios unless trained on extensive datasets. YOLO versions perform best with high-resolution videos (e.g., 4k) or cameras closer to traffic. Customizing deep learning models for vehicle detection requires large-scale training, which is time-intensive.

Vehicle counting, essential in many fields, involves estimating the number of objects in still images or video frames. Recent machine learning solutions, particularly supervised methods, are categorized into regression-based and detection-based approaches [26], [26]. Regression-based methods map image features to object counts or density maps without detecting individual instances. While effective in crowded scenes with unclear object boundaries, they struggle with perspective distortion and large objects, providing only general object locations. In contrast, detection-based methods like DeepSORT excel in tracking objects. DeepSORT builds on the SORT algorithm by integrating motion and appearance features for accurate tracking, assigning unique identifiers to vehicles, and minimizing identity errors [32]. The tracking process requires the vehicle's travel distance to calculate velocity, obtained through paired cameras or two known-distance lines [27]. The two-line method is faster and more cost-effective than photogrammetry, which demands overlapping cameras and is impractical for single-camera systems like 511. Additionally, the two-line method enables speed estimation and vehicle counting separately for each lane, enhancing efficiency and accuracy.

As previously discussed, an effective algorithm needs to strike a balance between factors like cost, accuracy, computational time, and adaptability to various conditions. Based on previous research, Table 1 summarizes the key criteria directly impacting these aspects.

Researchers have proposed practical vehicle detection, tracking, and speed estimation approaches to enhance traffic monitoring with radiometric cameras. However, there remains a gap in developing algorithms that perform efficiently under different lighting conditions, such as day and night transitions. Furthermore, limited research has addressed the challenges posed by varying illumination during sunset, which is the most changing illumination period. Many previous approaches also require time-consuming training data to adapt to specific cameras. Although GPU processors can reduce computation time, it comes at a cost. The expense of employing a dedicated GPU for each camera in systems like 511 is substantial. Moreover, there is a lack of focus on developing algorithms that can function effectively under diverse weather conditions. Hence, the primary contributions of our algorithm in this study are as follows:

- We present a robust spatial-temporal graph to represent the road environment. Additionally, a likelihood optimization with a normalization stage is developed to improve the quality of the generated graph. Subsequently, we fit a polynomial model using Random Sample Consensus (RANSAC) to this enhanced graph for desirable parameter extraction.
- Our proposed algorithm is capable of vehicle counting and speed estimation without requiring training datasets. Additionally, the algorithm does not necessitate the implementation of high-performance GPUs. Significantly, there is no need to install specific cameras; they operate efficiently on existing highway cameras, spanning from low-resolution cameras (e.g., 511) to high-resolution ones.
- Despite not utilizing GPU processors, our algorithm will process video sequences in real time and can handle multiple cameras simultaneously using an ordinary computer system.
- The algorithm will be evaluated using millions of video frames under various weather conditions, including rainy, snowy, and windy days, which may result in the camera recording blurred images. The analysis encompasses lighting scenarios with darker environments, such as daytime, nighttime, and sunset. Diverse road infrastructures, from the USA to Canada, have been chosen to assess the algorithm's performance in varying environments.

II. PROBLEM DEFINITION

Highway cameras are fixed above the ground, capturing static road infrastructures with vehicles as the only moving elements, enabling a focus on road sections with vehicle motion. By placing a selection point (red star) on each lane (Fig. 1), vehicle movement is tracked. When a vehicle passes the point (e.g., Frame = 782), the value at the point peaks compared to when no vehicle is present (e.g., Frame = 686). This generates the Spatial-Temporal Diagram (STD), with the horizontal axis representing frame numbers and

TABLE 1. Summary of the previous works.

Reference	Tested Areas	Number of Frames	Processing System	Real-Time	Weather Diversity			Illumination Change		
					Rainy	Snowy	Windy	Day	Night	Sunset
Detection										
[12]	nine	425,344	Corei 7 CPU	-	yes	yes	-	yes	yes	-
[15]	one	76,856	GPU	yes	-	-	-	-	yes	-
[16]	two	-	16 GB RAM	-	-	-	-	yes	-	-
[17]	one	19,200	-	-	-	-	-	yes	-	-
[11]	three	22,700	Titan Xp GPU	-	yes	yes	yes	yes	-	-
[18]	two	70,999	GTX1080 Ti GPU	yes	-	-	-	yes	-	-
[13]	one	9,580	24 GB GPU	yes	-	-	-	yes	-	-
[19]	one	293	-	-	-	-	-	yes	yes	-
[20]	one	41,125	2GHz CPU	yes	-	-	-	yes	-	-
Tracking										
[21]	one	$\sim$30,000	-	-	-	-	-	yes	-	-
[22]	three	104,400	-	-	-	-	-	yes	-	-
[23]	four	5,562	GPU	-	-	-	-	-	-	-
[24]	twelve	-	Raspberry pi	-	yes	-	-	yes	-	-
[25]	three	300	Cori7-CPU	yes	-	-	-	-	yes	-
[26]	one	11,129	-	-	-	-	-	yes	-	-
Speed Estimation										
[27]	five	30,000	-	-	-	-	-	yes	-	-
[28]	three	-	Titan X GPU	-	-	-	-	yes	-	-
[29]	one	-	-	-	-	-	-	yes	-	-
[30]	one	-	TITAN XP GPU	-	-	-	-	yes	-	-

the vertical axis showing the output aggregation value for the selected point. The STD must accommodate various challenging scenarios, including diverse weather conditions and illumination changes, from day to night, ensuring that vehicle peaks remain identifiable even in adverse weather.

The generation of the STD must account for changing illumination throughout the day and seasonal variations, particularly in winter, when cloud cover affects the light levels. Accurate offset detection is essential to avoid misidentifying the road surface as vehicles. Additionally, a line fitting model is crucial for aligning with the STD, helping to accurately identify vehicle peaks, which are counted to estimate traffic flow. The algorithm must also handle multiple peaks per vehicle, especially with larger vehicles like trucks, and assign unique identifiers to ensure reliable vehicle tracking. Moreover, the STD must allow for speed estimation by using two known-distance points on each lane, with time and distance data allowing for accurate speed calculation.

III. METHODOLOGY

Our algorithm works in four main steps. First, it creates an STD diagram where peaks show vehicle presence. Then, it refines this diagram using Gaussian Mixture Model (GMM) likelihood optimization. Next, it uses Random Sample Consensus (RANSAC) to fit an equation and detect peaks, aiding in vehicle counting and speed estimation. Finally, it calculates traffic parameters like flow and volume. Fig. 2 displays the flowchart, with a detailed discussion of each step.

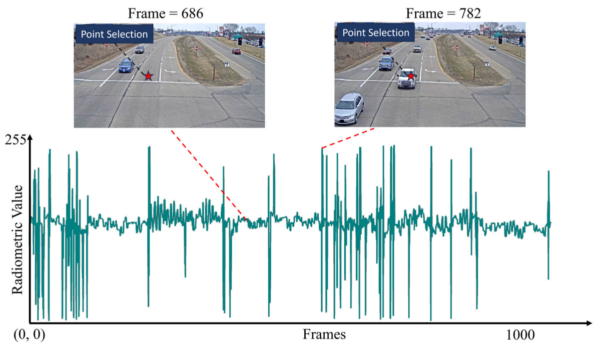

FIGURE 1. Displaying Spatial-Temporal Diagram (STD) of a point selection on the roadway.

A. SPATIAL-TEMPORAL DIAGRAM (STD) GENERATION

The process of generating an STD graph involves three steps: point selection, spatial pyramid feature extraction, and output aggregation, each of which is detailed below. Two steps of road detection and road marking extraction can do the point selection process. Regarding road detection, vision transformers (ViT) have recently had a noticeable performance in semantic segmentation or object classification [33]. Currently leading in various image recognition tasks within computer vision, Convolutional Neural Networks (CNNs) face a compelling rival in Vision Transformers (ViT). ViT models surpass the prevailing state-of-the-art (CNN) by nearly fourfold in accuracy and computational efficiency [34].

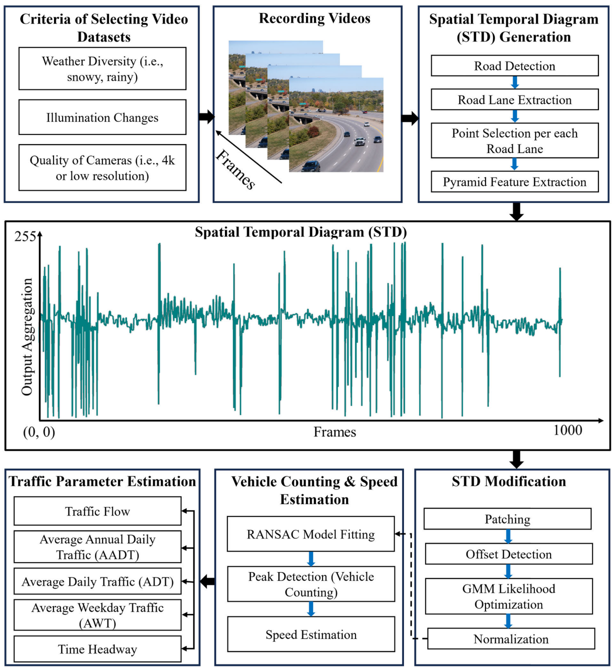
Criteria of Selecting Video Datasets
Weather Diversity (i.e., snowy, rainy)
Illumination Changes
Quality of Cameras (i.e., 4k or low resolution)
Recording Videos
Frames
Spatial Temporal Diagram (STD) Generation
Road Detection
Road Lane Extraction
Point Selection per each Road Lane
Pyramid Feature Extraction
Spatial Temporal Diagram (STD)
255
Output Aggregation
(0, 0)
Frames
1000
Traffic Parameter Estimation
Traffic Flow
Average Annual Daily Traffic (AADT)
Average Daily Traffic (ADT)
Average Weekday Traffic (AWT)
Time Headway
Vehicle Counting & Speed Estimation
RANSAC Model Fitting
Peak Detection (Vehicle Counting)
Speed Estimation
STD Modification
Patching
Offset Detection
GMM Likelihood Optimization
Normalization

FIGURE 2. Flowchart of the proposed methodology.

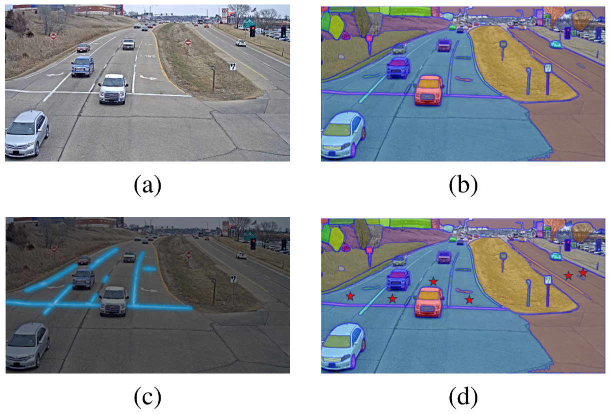

FIGURE 3. The processes of point selection: (a) a sample roadway image; (b) segmentation of the image by SAM methodology; (c) road marking detection on the road segments; (d) point selection between each two road markings and updating the point selection process manually.

A notable ViT-based model in computer vision is the Segment Anything Model (SAM) [35]. SAM transforms the input image into semantic segments, as illustrated in Fig. 3-b. Notably, distinct colors, such as road surfaces, are assigned to each object. SAM effectively separates two distinct roadways, facilitating counting of vehicles on different road infrastructures. In the next step, segments of roads are detected by applying this [19] method. If the camera has high resolution and a near part of the road area is going to be analyzed, the road markings are extracted from the next step by the SAM. The road markings are the linear segments seen in Fig.3-c. Finally, the middle point is a selection between the two adjacent road markings. Since separating road markings is somewhat challenging, particularly for cameras far away from the road surfaces, the point selection is manually updated on each side of the lanes. Fig. 3-d illustrates the six selected points in the regions. This point selection is done once per camera and can be done, for example, 100 cameras in less than 2 hours of a working day.

After selecting the points, kernels of odd sizes are employed, with the center pixel being the selected point. For instance, a $k \times k$ kernel contains pixel values of $P^{c}_{i,j}$ (where C is the number of image channels, typically 3 for RGB). Four kernel sizes, namely 1, 3, 5, and 7, were chosen through trial and error. Notably, this pyramid feature extraction method enhances the significance of central pixels, as they are part of all kernels. Subsequently, the obtained kernels are merged and flattened into a single layer. This layer has $84 \times C$ values and is called a fully connected layer. Following this, three parameters (Equ. 1, 2, 3)—average, skewness, and standard deviation of the fully connected layer—are calculated and incorporated into the subsequent layer. It is expected that when a vehicle is in the region of kernels, for example, in the frame of t, the two values of mean and standard deviation will be a large value in comparison to when the vehicle does not exist (frame of $t+n$, where n is some next frames which the vehicle is passed the selected point). Also, the skewness should be near zero when a vehicle is not in the pyramid sections. Finally, the aggregation output is determined based on these three values. Initially, the probabilities of these three values are calculated based on the SoftMax algorithm (Eq. 4). The probability value of the mean is expected to be near 1 when a vehicle is not in the kernels or pyramid sections because the values of standard deviation and skewness will be near zero. Therefore, the Output Aggregation equals the sum of the multiple of the obtained probabilities to their corresponding values. As seen in Fig. 1, for each selected point, an STD can be obtained in which the peaks indicate the location of the vehicles.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{1}$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^2} \tag{2}$$

$$\gamma = \frac{1}{N\sigma^3}\sum_{i=1}^{N}(x_i-\mu)^3 \tag{3}$$

$$p_i = \frac{e^{x_i}}{\sum_{j=1}^{3} e^{x_j}} \tag{4}$$

where x_i the value of the flatted fully connected layer. N the number of the fully connected layer where it equals $84 \times C$. p_i is three values of mean, skewness, and standard deviation, $i \in \{1, 2, 3\}$.

B. STD MODIFICATION

Since our algorithm will finally be implemented on 24/7 cameras, the STD graph has more than $30 \times 60 \times 60 \times 24 = 2,592,000$ frames in a single day if we consider the camera record 30 frames per Second (FPS). Indeed,

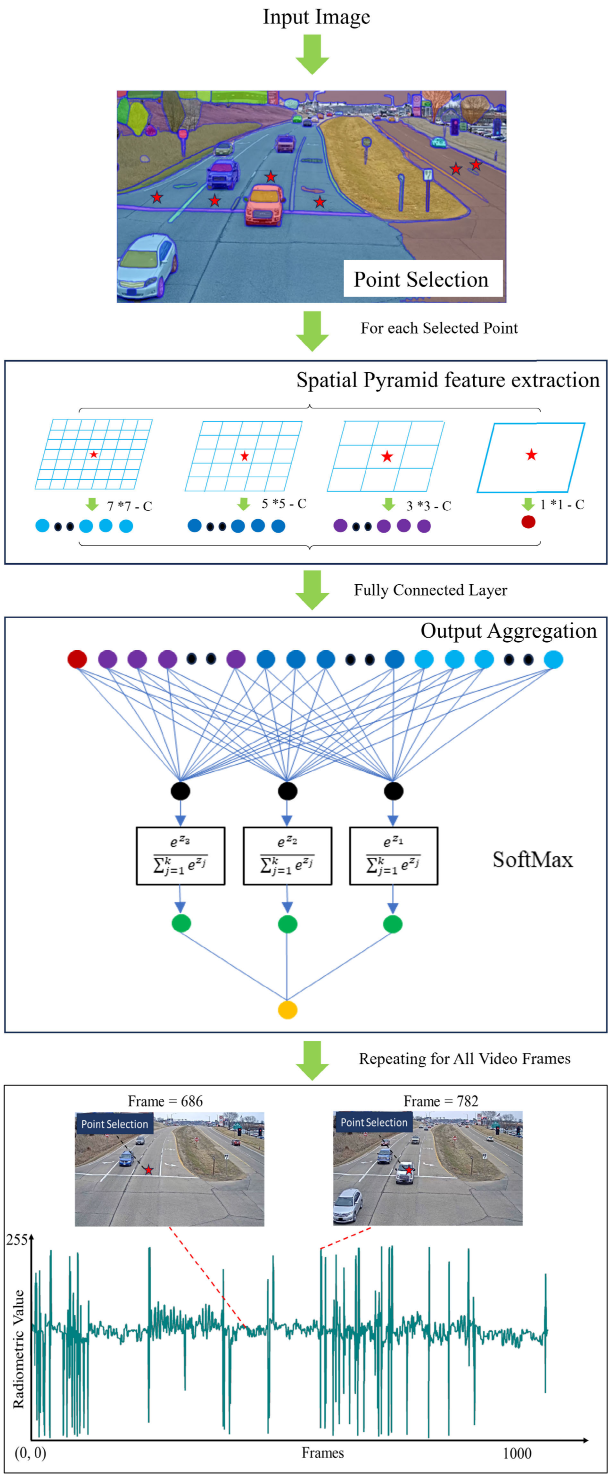

FIGURE 4. Displaying Spatial-Temporal Diagram (STD) steps.

finding a universal model fitting on this STD graph for peak detection is somewhat challenging; therefore, we propose a patching scenario and work separately on each created patch. In addition, abrupt illumination changes that result in an offset on the STD graph may occur due to, for example, switching on the pole lights at sunset or appearing darker clouds in winter seasons. Thus, this abruption also should be found and updated in the patching step. Finally, a GMM likelihood optimization and normalization between zero and one are applied on each created patch to modify the STD graph. This modification better represents the peaks and simplifies the model fitting step to count vehicles by peak detection.

1) PATCHING

Signal patching (Fig. 5-a, b) involves segmenting a continuous signal into smaller sections or frames called patches. These patches allow localized analysis and processing of specific signal segments, facilitating various signal processing tasks such as feature extraction, denoising, spectral analysis, and pattern recognition. More importantly, finding a robust and fitted equation line between each patch is much more possible than a universal curve function on a lengthy STD figure. There are primarily two approaches to signal patching. The first method divides the signal into distinct, non-overlapping segments with a fixed duration ΔT. The number of these patches is denoted as $N_{patches}$ within the signal and is determined by $N_{patches} = T/\Delta T$, ensuring equal-duration segments. The second approach involves overlapping patches, where segments of the signal share a portion, resulting in patches that have overlapping regions. In this study, we have used non-overlapping patching with 4000 frames per patch.

The offset detection method identifies signal offsets in each patch by dividing them into two segments and analyzing their statistical properties, such as mean and variance. The mean of a segment, defined in Equation 5, is calculated as the average value of all points within the range $[x_m, x_n]$, where x_m and x_n represent the start and end indices of the segment, respectively. This is mathematically expressed as:

$$\text{mean}([x_m \ldots x_n]) = \frac{1}{n-m+1}\sum_{r=m}^{n} x_r \tag{5}$$

The variance, as shown in Equation 6, measures the extent to which the values deviate from the mean. It is calculated by summing the squared differences between each data point x_r in the segment and the segment's mean. The variance is normalized by dividing the sum of squared deviations by the total number of points in the segment, $(n - m + 1)$:

$$\begin{aligned} S_{var}([x_m \ldots x_n]) &= \frac{1}{n-m+1}\sum_{r=m}^{n}(x_r - \text{mean}([x_m \ldots x_n]))^2 \\ &= \frac{S_{xx}|_m^n}{n-m+1} \end{aligned} \tag{6}$$

In this context, S_{xx} is defined in Equation 7 as the cumulative deviation for all points in the segment. It is computed by summing the product of deviations for x_r and y_r from their respective means:

$$S_{xx}|_m^n = \sum_{r=m}^{n}(x_r - \text{mean}([x_m \ldots x_n]))(y_r - \text{mean}([y_m \ldots y_n])) \tag{7}$$

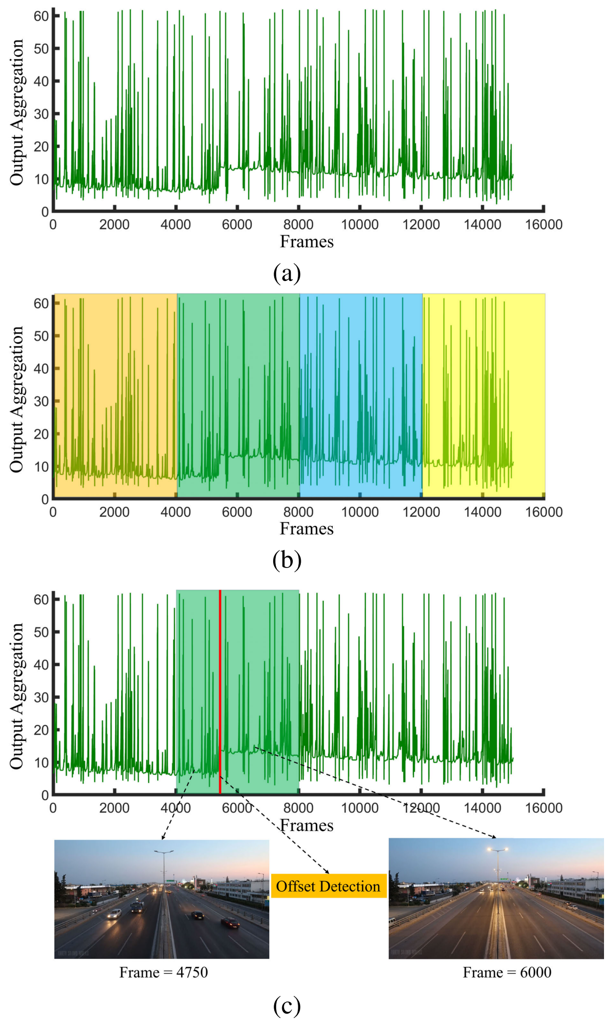

FIGURE 5. **STD graph in which a pole light caused offset; (b) displaying the patching process by colorization; (c) offset detection where the red line represents the position of the occurred offset.**

The offset detection method iteratively adjusts the division point to minimize the total residual error. The process involves finding the optimal horizontal level for each patch by evaluating changes in mean and variance and identifying the division point that results in the smallest residual error across all sections.

The patching and offset detection process is illustrated in Fig. 5. Fig. 5-c displays the red line, which indicates the detected offset's location on the STD graph. In this graph, the offset was due to switching on the pole lights. Subsequently, the patch with an identified offset is split into distinct patches. For STD graphs without offsets, the red line sits at the boundary of each patch.

2) GAUSSIAN MIXTURE MODEL (GMM) LIKELIHOOD OPTIMIZATION

This step is going to modify the STD graph on each patch. The GMM is a probabilistic model representing a probability distribution over a dataset [35]. It assumes that the data is generated from a mixture of several Gaussian distributions, each with its mean and covariance. The GMM aims to estimate the parameters of these Gaussians to fit the observed data best. Let us delve into the likelihood optimization process for a GMM, which involves maximizing the likelihood function to determine the model's parameters.

This procedure typically involves an Expectation-Maximization (EM) algorithm [36]. Consider a dataset $X = \{x_1, x_2, \ldots, x_N\}$ containing N observations in D-dimensional space. Here, X is the STD graph with one dimension. The GMM is represented as a weighted sum of K Gaussian distributions:

$$p(x|\theta) = \sum_{k=1}^{K} \pi_k N(x|\mu_k, \Sigma_k) \tag{8}$$

where θ represents the parameters of the GMM: $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$.

- π_k is the mixing coefficient for the kth Gaussian ($\sum_{k=1}^{K} \pi_k = 1$)
- μ_k is the mean vector of the kth Gaussian
- Σ_k is the covariance vector of the kth Gaussian
- $N(x|\mu_k, \Sigma_k)$ denotes the Gaussian probability density function

The goal is to maximize the log-likelihood function to estimate the optimal parameters θ:

$$\theta^* = \sum_{n=1}^{N} \log p(x_n|\theta) \tag{9}$$

The EM algorithm iteratively maximizes the likelihood function in the E-step (Expectation) and the M-step (Maximization).

E-step (Expectation): Initialize the parameters θ randomly. Calculate the posterior probabilities (responsibilities) using Bayes' theorem:

$$\gamma(z_{nk}) = \frac{\pi_k N(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_k N(x_j|\mu_j, \Sigma_j)} \tag{10}$$

M-step (Maximization): Update the model parameters using the responsibilities:

Mean Update:

$$\mu_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) x_n}{\sum_{n=1}^{N} \gamma(z_{nk})} \tag{11}$$

Covariance Update:

$$\Sigma_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^T}{\sum_{n=1}^{N} \gamma(z_{nk})} \tag{12}$$

Mixing coefficient Update:

$$\pi_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})}{N} \tag{13}$$

The GMM likelihood optimization via the EM algorithm involves iteratively updating parameters based on the data's latent structure, allowing the model to converge to a configuration that best represents the observed data distribution. Finally, the Gaussian model, its parameters of the mean (μ) and standard deviation (σ) were obtained and applied on the STD graph patches:

$$\text{Gaussian Model} = \frac{e^{-0.5(\frac{x-\mu}{\sigma})^2}}{\sigma\sqrt{2\pi}} \tag{14}$$

Next, it is necessary to normalize the data within the range of 0 to 1:

$$\text{Normalization} = \frac{|x-\mu|}{\max(\text{Modified STD})} \tag{15}$$

Figure 6 illustrates the modified STD and normalization on a sample STD graph, where the peaks still indicate the presence of vehicles. In the original STD graph (Figure 6-a), vehicles appearing across consecutive frames caused fluctuating values, posing a challenge for accurate vehicle counting. However, in the modified STD graph, each vehicle frame value shows a consistent positive peak value.

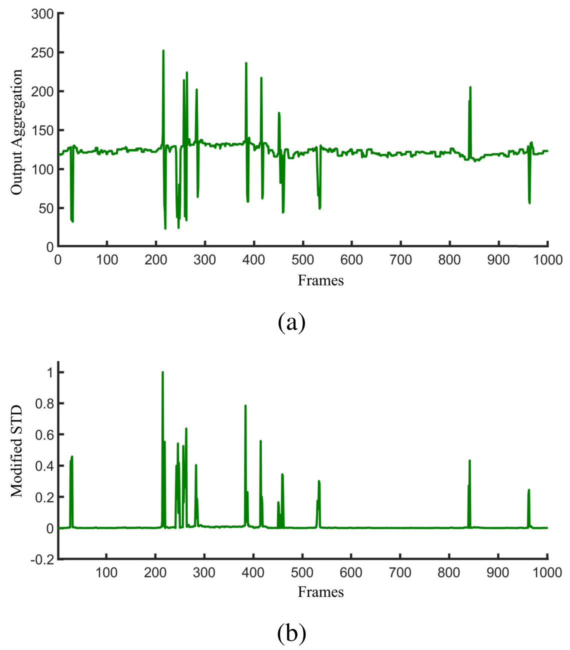

FIGURE 6. Processes of STD modification: (a) a sample STD graph; (b) a modified STD graph.

C. VEHICLE COUNTING AND SPEED ESTIMATION

On the modified STD graph, the peaks are vehicles, and others are generally called background locations. A model fitting based on Random Sample Consensus (RANSAC) is considered to model the background pattern. RANSAC can find and model a robust equation on the modified STD graph. Afterwards, we will use this model to count the vehicle and speed estimation.

1) RANSAC MODEL FITTING

RANSAC is an iterative algorithm used in computer vision and robust estimation tasks to estimate model parameters from data [37]. It starts by selecting random subsets from the data, initializing parameters like the minimum required data points, a threshold to distinguish inliers from outliers, and either a maximum iteration count or a stopping criterion. The algorithm iterates by selecting subsets, fitting a model, and evaluating the number of inliers (points consistent with the model). The model with the largest consensus set, or the highest number of inliers, is chosen. Afterward, model parameters are recalculated using all inliers, and refinement techniques like least squares may be applied to enhance accuracy. RANSAC's strength lies in its ability to mitigate the impact of outliers, ensuring robust model fitting.

In this study, polynomial fitting is employed to model signal data, where the algorithm receives modified Spatial-Temporal Diagram (STD) values (x_i, y_i) and uses the polynomial degree M specified by the user. Initially, polynomial coefficients are randomly initialized, and the algorithm performs polynomial fitting using least squares regression. The polynomial function is optimized to minimize the error between the fitted polynomial and the data points. This is done by reducing the sum of squared errors (SSE), with the residual errors of each data point calculated. Polynomial coefficients are refined using optimization techniques like Gradient Descent [38]. The iterative process continues until convergence, ensuring that the fitted polynomial best represents the relationships in the modified STD data. Tests on over 214 STD graphs identified optimal parameter ranges for the stopping criterion and random sample.

$$\text{Error} = \sum_{i=1}^{N}(y_i - (a_M x_i^M + a_{M-1}x_i^{M-1} + \cdots + a_1 x_i + a_0))^2 \tag{16}$$

$$a_j^{t+1} = a_j^t - \alpha\frac{\partial \text{Error}}{\partial a_j} \tag{17}$$

2) VEHICLE COUNTING

In Fig. 6-b, the vehicles are evident as peaks. Hence, we employ a local maximum point detection method to locate these peaks and subsequently recognize and count the vehicles between them. Finding local maxima in signal processing is like looking for the highest points in a data landscape. Imagine a graph representing your data; local maxima are those peaks with higher values than their neighboring points. Here, we consider the neighborhood points equal to the random sample (10). It works here: every point is checked as it moves along the data. A point whose value is greater than that of the points immediately preceding and following it is a candidate for a local maximum. This method helps identify significant spikes or high points within the modified STD.

In Fig. 7-b, the outcome of the local maxima process on the modified STD graph is illustrated. It reveals numerous identified local maxima points, including irrelevant points within the background values. Based on the RANSAC model fitting (depicted by the blue line in Fig. 7), a candidate peak selection stage is introduced to refine this. Local maxima points within a distance lower than the stopping criterion (0.1) are eliminated, while the rest are deemed candidate peaks (Fig. 7-c). Indeed, we used the inlier points of the RANSAC model fitting points from the candidate peaks selection to remove useless peaks. Ultimately, consecutive candidate peaks are treated as a single vehicle, given that a vehicle might span several continuous frames. Therefore, those candidate peaks in consecutive manners are selected as one class, the maximum one remains, and others are deleted. Fig. 7-d showcases the final output of the vehicle's counting. This example shows 11 vehicles passed through a point selection within 1000 recorded frames.

D. SPEED ESTIMATION

Speed measures how quickly something moves or the rate at which an object changes its position over a certain period. Indeed, the speed is the distance traveled per unit of time. It is typically expressed in units such as meters per second (m/s), kilometers per hour (km/h), or miles per hour (mph). Speed is a scalar quantity, representing only how fast an object moves, regardless of its direction. This step aims to estimate the speed of vehicles by the modified STD graph and vehicle counting step. The modified STD graph gives the number of vehicles that passed a point selection region and the exact timeline. This means that if we consider two points of Point A and Point B as the point selection where their distance (Distance_{AB}) is known, and the time of the Vehicle_X which passed the two points, can be obtained.

Since cameras record the time of frames, the passing time t between the two points is equal to dividing the subtraction of the number of frames in which the vehicle passed two points to the camera's FPS. Each camera has a specific and known FPS. Fig. 8 shows the modified STD graphs of Point A and Point B. The vehicle's counting step is shown in black on each graph. At Point A, for example, Vehicle_9 has passed on the frame of 508 while it goes on the frame of 534 on Point B. As the FPS of cameras was 10, the traveled time equals $t = (534 - 508)/10 = 2.6$ seconds. Since the distance of two points was 20 meters, the speed of Vehicle_9 was estimated to $\text{Speed} = 20/2.6 = 7.69$ m/s or $7.69 \times 3.6 = 27.68$ km/h. The speed of Vehicle_{10} was estimated to equal 37.89 km/h.

E. TRAFFIC PARAMETERS

Traffic parameters are essential for understanding and managing traffic flow and behavior on roadways. Traffic flow, defined as the number of vehicles passing a specific point over a set period, can be calculated by counting the vehicles that appear in a certain number of frames, $\text{Vehicles}_{\text{Count}}$, during a given period t. For instance, with a camera recording at 30 FPS over one hour (3600 seconds), the number of frames would be $3600 \times 30 = 108,000$. The traffic flow is then calculated as:

$$\text{traffic flow} = \frac{\text{Vehicles}_{\text{Count}}}{t}$$

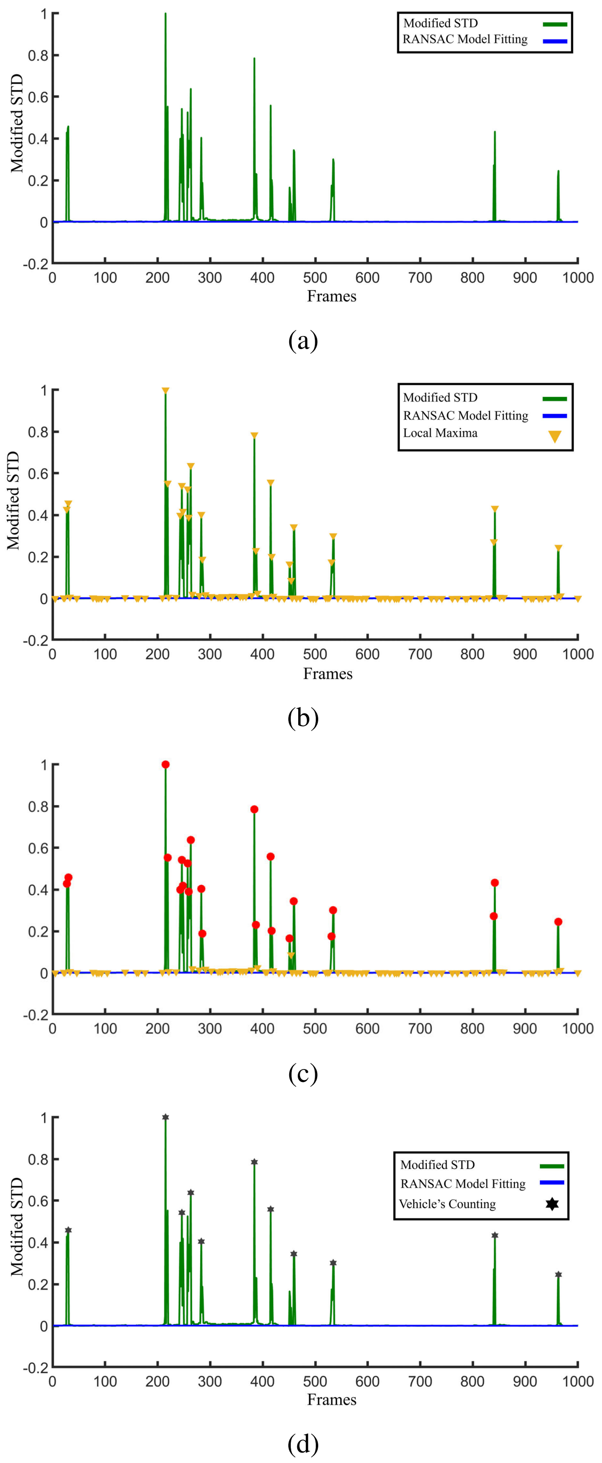

FIGURE 7. Process of vehicle counting: (a) a fitted model based on RANSAC on the modified STD; (b) local maxima detection; (c) candidate peaks detection; and (d) vehicle counting.

Volume variation, which captures fluctuations in traffic volume, can be measured through parameters like the Average Annual Daily Traffic (AADT), Average Annual Weekday Traffic (AAWT), Average Daily Traffic (ADT), and Average Weekday Traffic (AWT). These values provide insights into traffic distribution and help guide highway design decisions.

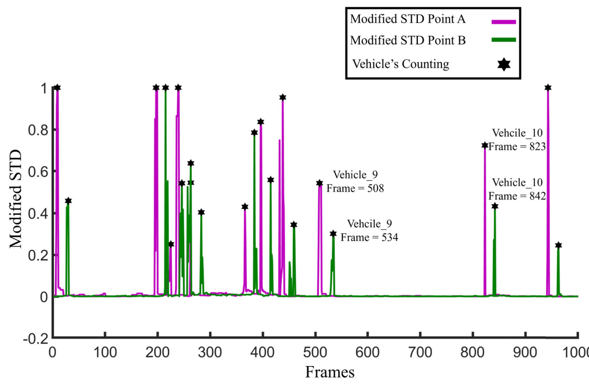

FIGURE 8. Speed estimation of vehicles by generating two modified STD graphs on two points of a lane road.

Time headway, the gap between consecutive vehicles in the same lane, is another critical parameter. It is measured in seconds and is essential for understanding traffic flow dynamics, congestion, and safety. Shorter headways often result in congestion and reduced efficiency. In comparison, longer headways improve traffic flow and safety. These parameters, including traffic flow and time headway, are fundamental for traffic analysis, allowing engineers to optimize infrastructure and improve overall traffic management and safety.

IV. DATASETS AND RESULTS

A. DATASETS

In preparing our algorithm for an industrial environment, we have prepared challenging video datasets to assess its performance thoroughly under diverse conditions. Table 2 outlines the crucial parameters of these datasets. The dataset comprises 11.438 million frames, encompassing various illumination changes and weather conditions. These challenging video datasets aim to cover almost all conceivable scenarios, ensuring a comprehensive evaluation of our algorithm's capabilities.

This study also used 511 highway cameras in Quebec City, New York, Montreal, and Edmonton, with 367,870 frames. This data was recorded between 2021 and 2023 in various weather conditions. Some of them are real-time, like the 511 highway cameras of Edmonton, Canada, and some are updated every 5 seconds, like Quebec City, which, after a while, is updating. These kinds of cameras generally have a resolution of around 352 × 240 pixels.

Also, several shared videos have been considered throughout two platforms, YouTube and Istockphoto. The keywords were used: "Traffic Jam", "Highway Road", "Night City Traffic", "Highway City Traffic Sounds", "Highway Traffic Relax", "Los Angeles Freeway", and "Highway Roadside Ground View". The other two datasets of Next Generation Simulation (NGSIM) Program I-80 Videos [39], and Nvidia AI (www.aicitychallenge.org) are publicly available for traffic monitoring. These datasets have covered more than 7.5 million frames in various weather conditions and diverse camera quality ranging from low resolution to high resolution.

B. RESULTS

1) ACCURACY ASSESSMENT

Metrics of Equ. (18) to (23) are crucial in evaluating algorithms' performance, including our vehicle counting algorithm. Precision accuracy estimates the accuracy of positive predictions, calculating the ratio of true positive predictions to all positive predictions. Recall accuracy, also known as sensitivity, measures the model's ability to correctly identify all positive instances by considering true positive predictions against the total actual positives. The False Negative Rate (FNR) indicates the proportion of actual positive instances incorrectly predicted as negative. At the same time, the False Discovery Rate (FDR) determines the ratio of false positives to the total positive predictions made by the model. The Critical Success Index (CSI), or Threat Score, assesses the agreement between predicted and observed outcomes, encompassing event and non-event detections. Lastly, the F1-Score balances precision and recall, representing the harmonic mean of the two metrics, ensuring a comprehensive evaluation of false positives and negatives in the model's predictions. Collectively, these metrics offer insights into the model's predictive accuracy, highlighting its strengths and areas for improvement in various classification scenarios. Notably, the optimal value of FNR and FDR is zero, while the others are 100%.

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \times 100 \tag{18}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \times 100 \tag{19}$$

$$\text{False Negative Rate (FNR)} = \frac{\text{FN}}{\text{FN}+\text{TP}} \times 100 \tag{20}$$

$$\text{False Discovery Rate (FDR)} = \frac{\text{FP}}{\text{FP}+\text{TP}} \times 100 \tag{21}$$

$$\text{Critical Success Index (CSI)} = \frac{\text{TP}}{\text{TP}+\text{FN}+\text{FP}} \times 100 \tag{22}$$

$$\text{F1-score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision}+\text{Recall}} \times 100 \tag{23}$$

where TP, FP and FN are True Positive, False Positive, and False Negative, respectively.

2) STD RESULTS

This section presents a comparative evaluation of multiple accuracy metrics across various datasets. It assesses the performance of our algorithm on different datasets collected from distinct sources. The precision values range from 67.2% to 99.7%, averaging 95.4%. The NGSIM dataset showcases the highest precision, while lower values are observed in the YouTube and NVIDIA AI datasets. Recall values range from 94.1% to 99.7%, averaging 96.9%, with the NGSIM dataset exhibiting the highest recall. The False Negative Rate (FNR) spans from 1.5% to 5.3%, averaging at 3.1%, showcasing the lowest rate in the NGSIM dataset. False Discovery

TABLE 2. Summary of the used datasets specifications.

Dataset	Overall Length (h)	Number of Frames (k)	Width	Height	FPS	Weather Diversity	Illumination Changes	Field of View	Video Quality
511 Cameras	∼3.5	368	1280 & 352	720 & 240	15 & 30	Snow Rain	Day, Night, Sunset	Cover All possible Views	Low, HD
NGSIM	∼13	473	640	480	10	-	Day	Vertical	Low
YouTube	∼31.5	3,413	640 & 1280 & 1920	360 & 720 & 1080	30	-	Day, Night, Sunset	Cover All possible Views	Low, HD, Full HD
Nvidia AI	∼66.5	7,184	800 & 1280 & 1920	410 & 960 & 1080	8 & 10 & 30	Snow Rain Wind	Day, Night, Sunset	Cover All possible Views	Low, HD, Full HD

Rate (FDR) ranges from 2.3% to 7.5%, with an average of 4.6%, where the YouTube dataset displays the highest FDR. Critical Success Index (CSI) varies from 88.6% to 96.6%, averaging 92.6%, with the NGSIM dataset demonstrating the highest CSI values. The F1 score fluctuates between 94.0% and 98.2%, averaging 96.1%, with NGSIM showcasing the highest F1 score among the datasets.

Across most metrics, the NGSIM dataset consistently demonstrates superior performance, while the YouTube and NVIDIA AI datasets exhibit comparatively lower accuracy in certain areas. The average values indicate reasonably good overall performance, particularly in precision, recall, and F1-Score metrics. Vehicle occlusion and shadows primarily affected the precision and recall accuracies negatively. Regarding the traffic parameters, the time frame difference between the two consecutive detected vehicles is equivalent to time headway. Also, the number of counted vehicles directly gives the volume traffic parameters, which, depending on the video times, measure the corresponding parameters. These parameters were not measured since we did not have a long time frame for a region from this study. Nevertheless, they can be obtained if these video datasets are recorded.

3) COMPUTATIONAL TIME

This study used a computer with a 12th Gen Intel(R) Core(TM) i7-4700HQ CPU running at 2.4GHz and 16.0 GB of RAM. The average time to process each frame was between 0.003 and 0.008 seconds. Each video frame was analyzed within this time range. Considering a video recording at 30 frames per second (FPS), our algorithm can analyze one-second frames in about $0.003 \times 30 = 0.09$ seconds. Our algorithm is approximately ten times faster in real-time vehicle counting and speed estimation.

V. DISCUSSION

A. CHALLENGING DATASETS

Selecting cameras for highway surveillance poses challenges based on several critical criteria that significantly impact their effectiveness in monitoring traffic flow and incidents. One primary criterion is camera resolution, where the choice between high-resolution, such as 4K, and low-resolution cameras greatly influences the quality and clarity of the captured images. The resolution choice is pivotal as high-resolution cameras offer sharper visuals, aiding in detailed object recognition. However, integrating low-resolution cameras might compromise detection accuracy due to limited image clarity. Another crucial factor is the diversity of weather conditions the cameras must withstand, including rain, snow, fog, and varying light intensities. Ensuring the cameras' functionality in diverse weather situations is essential for maintaining consistent surveillance efficacy.

Additionally, the field of view provided by the cameras, whether oblique or vertical, impacts the angles and scopes of the road captured, each presenting its advantages and limitations regarding coverage and depth for vehicle identification and tracking. The distance between the cameras and the road infrastructure is also significant. Cameras placed at varying distances may affect the size and contextual information of vehicles captured, influencing detection and classification accuracy. Lastly, adaptability to illumination changes, such as transitioning between day, night, and sunset lighting conditions, is vital for seamless and accurate surveillance throughout the day. Each criterion plays a crucial role in the dataset selection process, aiming to address challenges and create a robust and comprehensive surveillance system tailored for efficient highway monitoring. A balanced consideration of these factors was key to ensuring optimal performance and reliability in traffic surveillance.

B. SAM AND POINT SELECTION EVALUATION

The Segment Anything Model (SAM) presents a range of advantages in computer vision and object segmentation tasks. Its versatility allows for identifying and distinguishing various objects within images or videos, not limited to specific types. SAM excels in semantic segmentation by assigning specific labels or categories to individual pixels, facilitating detailed object identification and classification. This model adapts well to diverse scenarios and environmental conditions, effectively handling variations in lighting, weather, and other challenges. Even in complex environments with multiple objects, varying shapes, and occlusions, SAM can accurately segment objects, making it highly reliable. Its efficiency contributes to accurate segmentation results, enhancing the robustness of object detection and tracking systems. SAM's ability to differentiate between different roadways and objects enables precise tracking of vehicles in urban traffic surveillance systems. This versatility finds applications across various domains, including traffic surveillance, image recognition, and object tracking, owing to its adaptability and effectiveness in segmenting diverse objects.

In the development of SAM, about 11 million images with more than 1 billion masks have been used. Fig. 9 showcases ten SAM results across diverse road infrastructures. SAM effectively detects both road segments and vehicles, offering potential for vehicle detection. However, it shows sensitivity to changes in illumination, especially at night. Also, the computational time is challenging due to processing all image pixels. At the point selection step, manually checking the selected points is required because we should ensure the selected points are located on the road lanes and cover the all-road lanes. The advantage of this process is that it is done only once per camera.

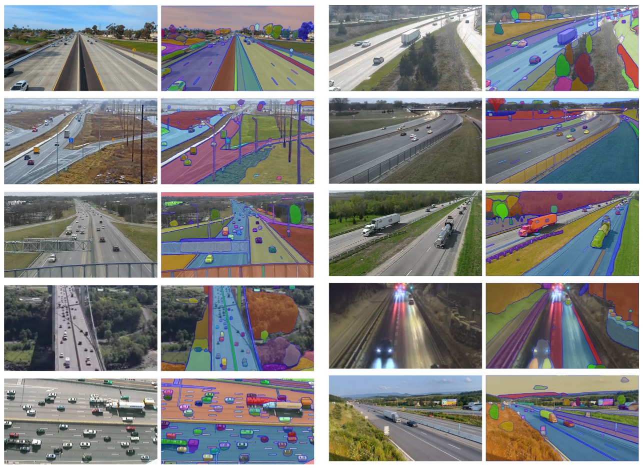

FIGURE 9. Performance of the SAM in road detection.

C. THE MODIFIED STD ANALYZING

Creating the modified STD graph is a crucial step in our algorithm; therefore, this graph should perform highly in diverse, challenging scenarios. Highlighting vehicles through peaks is the primary aim of the modified STD. We chose the most challenging videos to test our approach. For instance, the first graph in Fig. 10 illustrates the modified STD during nighttime, where peaks still represent vehicles. However, our algorithm exhibited its lowest Recall accuracy during instances when vehicle lights were on, which negatively impacted the modified STD graph. Future work might focus on implementing filters to minimize this effect.

Additionally, the second graph depicts the sunset, showing the modified STD's resilience in darker weather. In this scenario, the patching stage is key in avoiding an offset on the modified STD. Subsequent graphs illustrate snowy, rainy, and windy conditions, where the modified STD effectively reflects the vehicles as peaks.

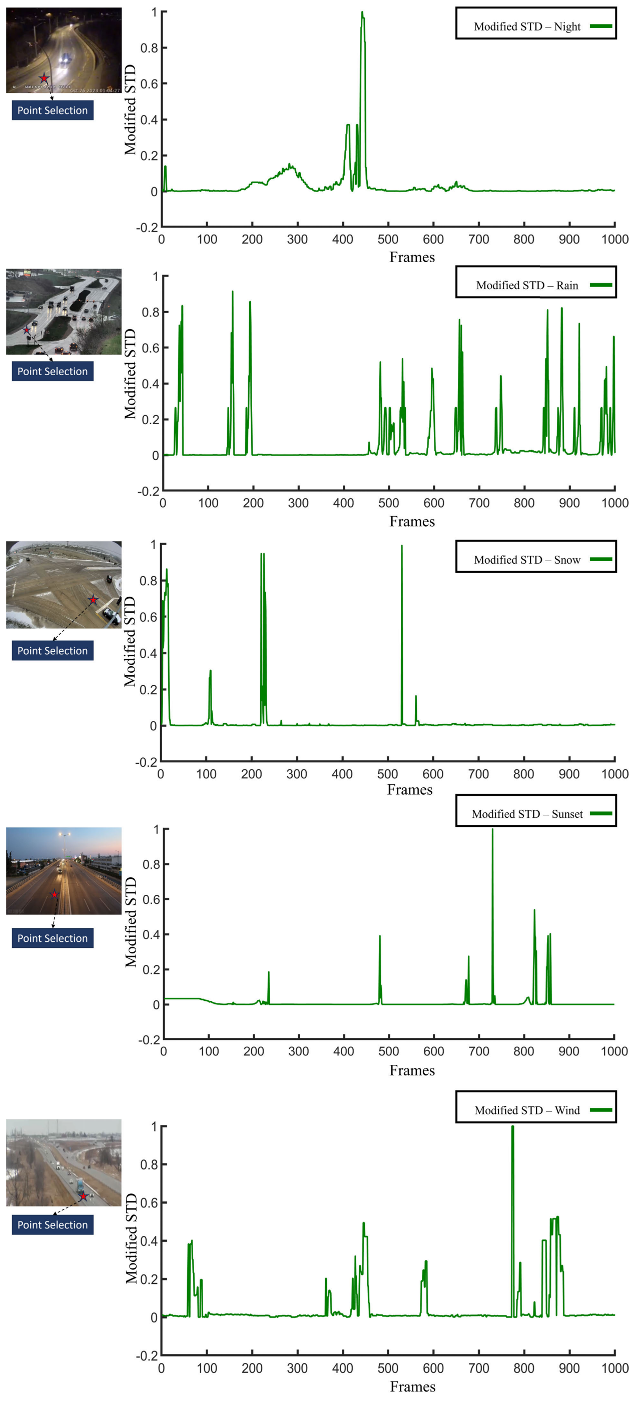

FIGURE 10. Displaying the modified STD on diverse, challenging situations like sunset and snowy days.

D. MODEL FITTING ASSESSMENT

The model fitting for the modified STD (Spatial-Temporal Data) graph aims to optimize a mathematical model, ensuring it effectively captures the inherent patterns within the modified STD graph. This designed model is a valuable tool, facilitating comprehensive analysis, aiding in insightful interpretation, and enabling accurate counting of vehicles within the modified STD graph. Through iterative adjustments and fine-tuning, the model aligns itself more closely with the background values in the STD data, enhancing its capability to recognize and clarify patterns. Ultimately, this refined model acts as a robust framework for understanding the complexities and dynamics embedded within the modified STD graph, thereby empowering more informed decision-making and deeper insights into spatial-temporal data trends.

In this section, we have chosen a particularly challenging modified STD graph showing vehicles passing a point selection region at close intervals, with roughly one vehicle passing per second. This scenario leads to fewer recorded values for the background section, posing a challenge during the model fitting stage. Fig. 11 illustrates the polynomial fitting process on a modified STD graph using degrees (M) ranging from one to five. Two scenarios were evaluated: one incorporated RANSAC for coefficient calculation, and the other excluded RANSAC. In Fig. 11-a, the polynomials fitted via RANSAC demonstrate that the model fitting for the background values becomes inaccurate as M increases. Among the polynomial models calculated using RANSAC, the one with a degree of one ($M = 1$), representing a linear equation, exhibits the best performance. Fig. 11-b shows the polynomial fitting stage without RANSAC, indicating that none of the calculated polynomials are suitable for accurate fitting. This highlights the critical role of RANSAC, even when the modified STD graph contains numerous vehicles. For future works, various model fitting algorithms ranging from machine learning algorithms such as Support Vector Machine (SVM), Random Forest (RF), and Long short-term memory (LSTM) to mathematical methods like Autoregressive Integrated Moving Average (ARIMA) can be suggested.

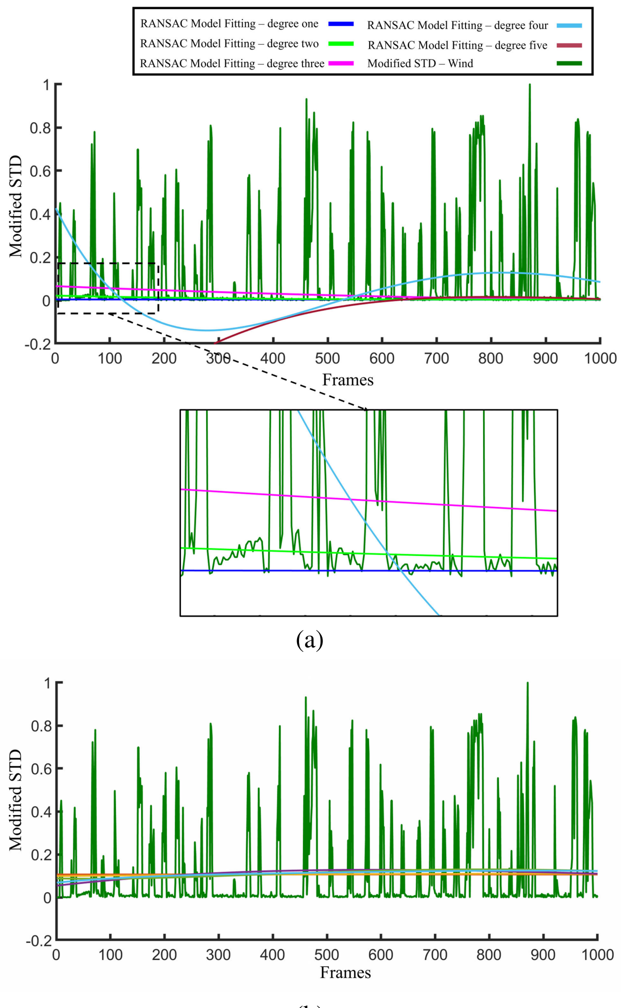

FIGURE 11. The model fitting assessment by various polynomials calculates the coefficients with (a) RANSAC and (b) not RANSAC.

E. COMPARISON WITH THE STATE-OF-THE-ART METHODOLOGIES

Previous work in this domain typically falls into two categories: Deep Learning algorithms and image/video processing techniques. Our evaluation compares our algorithm against these groups based on cost-effectiveness, real-time processing, and generality. As summarized in Table 2, our algorithm was tested on approximately 283 different road infrastructures, mainly in Canada and the USA, achieving over 96.1% accuracy across 11 million frames. This is significantly higher than previous methods evaluated on much smaller datasets (Table 1). Notably, our method operates without needing high-performance GPUs, a major advantage over deep learning-based methods. While our algorithm does not include vehicle classification, it has been tested under diverse weather and lighting conditions, unlike previous methods focused on clear weather scenarios.

Previous studies, such as [12], have compared deep learning models, including YOLOv7 and FasterRCNN, and concluded that YOLOv7 exhibits the best performance for vehicle detection in highway camera footage. Based on this study, we evaluated our algorithm using the provided datasets and methods. The results showed that our algorithm and YOLOv7 achieved good accuracy when vehicles were recorded with high-quality, high-resolution cameras. However, YOLO models struggled to maintain performance for remote areas of the road, which are typically covered by cameras due to their wide field of view. In contrast, our algorithm achieved 67% to 85% accuracy in vehicle counting under these challenging conditions.

Real-time processing is a key advantage of our approach, allowing for immediate analysis and decision-making. Our algorithm processes data with minimal latency, running between 0.003s and 0.008s per frame, making it suitable for applications like vehicle counting and surveillance. In comparison, deep learning models such as YOLOv8, YOLOv7, and FasterRCNN needs GPUs processors to handle data in a real time situations.

F. APPLICATIONS IN ADVANCED TRANSPORTATION SYSTEMS

Our algorithm demonstrates significant potential for integration with emerging Intelligent Transportation Systems (ITS) technologies. In the context of unmanned vehicles and intelligent traffic control, our approach offers several key advantages:

First, regarding autonomous vehicle systems, our algorithm's ability to accurately detect and track vehicles in real-time (0.003-0.008 seconds per frame) provides crucial input for navigation and path planning. The spatial-temporal data generated by our system can help autonomous vehicles better understand traffic flow patterns and make more informed decisions. Since our algorithm performs effectively under various weather and lighting conditions, it offers reliable environmental perception that complements autonomous vehicles' existing sensor systems.

Our algorithm's capabilities align well with modern reinforcement learning approaches regarding intelligent traffic signal control. The real-time vehicle counting and speed estimation data can be direct input for reinforcement learning models managing traffic signals. For instance, our system's traffic flow measurements and speed estimations can help optimize signal timing patterns. The algorithm's ability to process multiple lanes simultaneously makes it particularly valuable for heterogeneous traffic conditions, where different vehicle types and varying traffic densities must be considered for optimal signal control.

Furthermore, our algorithm's cost-effectiveness and compatibility with existing camera infrastructure make it particularly valuable for large-scale ITS deployment. Unlike methods requiring specialized sensors or high-performance computing resources, our approach can be implemented using existing traffic cameras and standard computing hardware. This makes it an economically viable solution for widespread adoption in smart city initiatives.

VI. CONCLUSION

This research introduces a novel Spatial-Temporal Diagram (STD) algorithm for real-time vehicle counting and speed estimation in traffic surveillance. Our algorithm demonstrated robust performance across varying conditions by evaluating over 11 million frames from diverse sources, including 511 highway cameras in the USA and Canada. Key methodological contributions include:

1. A cost-effective approach that performs exceptionally well with existing low-resolution cameras, eliminating the need for expensive equipment upgrades

2. Superior accuracy compared to state-of-the-art deep learning methods, achieving over 96% accuracy while YOLOv8 and Faster RCNN achieved only 60% on identical datasets

3. Real-time processing capabilities of 0.003-0.008 seconds per frame without requiring GPU support

Our experimental results validate the algorithm's effectiveness across diverse weather conditions and illumination changes while maintaining high precision and recall accuracies. Future work will address current limitations in vehicle occlusion handling and complex scenario management.

ACKNOWLEDGMENT

The authors would like to thank all the individuals and organizations who made these datasets and algorithms available. They want to express their sincere appreciation and gratitude to Mitacs for their funding support in making this project possible.

REFERENCES

[1] Q. Ren, C. He, Q. Huang, D. Zhang, P. Shi, and W. Lu, "Impacts of global urban expansion on natural habitats undermine the 2050 vision for biodiversity," *Resour., Conservation Recycling*, vol. 190, Mar. 2023, Art. no. 106834.

[2] B. Verma, R. Snodgrass, B. Henry, B. Smith, and T. Daim, "Smart cities—An analysis of smart transportation management," in *Managing Innovation in a Global and Digital World: Meeting Societal Challenges and Enhancing Competitiveness*, R. Tiwari and S. Buse, Eds., Wiesbaden, Germany: Springer, 2020, pp. 367–388.

[3] B. P. L. Lau, S. H. Marakkalage, Y. Zhou, N. U. Hassan, C. Yuen, M. Zhang, and U.-X. Tan, "A survey of data fusion in smart city applications," *Inf. Fusion*, vol. 52, pp. 357–374, Dec. 2019.

[4] D. Shokri, H. Rastiveis, S. M. Sheikholeslami, R. Shahhoseini, and J. Li, "Fast extraction of power lines from mobile LiDAR point clouds based on SVM classification in non-urban area," *Earth Obs. Geomat. Engin*, vol. 5, no. 2, pp. 63–73, 2021.

[5] D. Shokri, M. Zaboli, F. Dolati, and S. Homayouni, "POINTNET++ transfer learning for tree extraction from mobile LiDAR point clouds," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 10, pp. 721–727, Jan. 2023.

[6] N. K. Jain, R. K. Saini, and P. Mittal, "A review on traffic monitoring system techniques," in *Soft Computing: Theories and Applications*, K. Ray, T. K. Sharma, S. Rawat, R. K. Saini, and A. Bandyopadhyay, Eds., Singapore: Springer, 2019, pp. 569–577.

[7] Y. Chen, W. Wang, and X. M. Chen, "Bibliometric methods in traffic flow prediction based on artificial intelligence," *Expert Syst. Appl.*, vol. 228, Oct. 2023, Art. no. 120421.

[8] T. Zhang, P. J. Jin, Y. Ge, R. Moghe, and X. Jiang, "Vehicle detection and tracking for 511 traffic cameras with U-shaped dual attention inception neural networks and spatial–temporal map," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2676, no. 5, pp. 613–629, May 2022.

[9] R. Kaur and S. Singh, "A comprehensive review of object detection with deep learning," *Digit. Signal Process.*, vol. 132, Jan. 2023, Art. no. 103812.

[10] V. Mandal and Y. Adu-Gyamfi, "Object detection and tracking algorithms for vehicle counting: A comparative analysis," *J. Big Data Analytics Transp.*, vol. 2, no. 3, pp. 251–261, Dec. 2020.

[11] R. Ghosh, "On-road vehicle detection in varying weather conditions using faster R-CNN with several region proposal networks," *Multimedia Tools Appl.*, vol. 80, no. 17, pp. 25985–25999, Apr. 2021.

[12] D. Shokri, C. Larouche, and S. Homayouni, "A comparative analysis of multi-label deep learning classifiers for real-time vehicle detection to support intelligent transportation systems," *Smart Cities*, vol. 6, no. 5, pp. 2982–3004, Oct. 2023.

[13] J. Sang, Z. Wu, P. Guo, H. Hu, H. Xiang, Q. Zhang, and B. Cai, "An improved YOLOv2 for vehicle detection," *Sensors*, vol. 18, no. 12, p. 4272, Dec. 2018.

[14] U. Mittal, P. Chawla, and R. Tiwari, "EnsembleNet: A hybrid approach for vehicle detection and estimation of traffic density based on faster R-CNN and YOLO models," *Neural Comput. Appl.*, vol. 35, no. 6, pp. 4755–4774, Feb. 2023.

[15] A. Bell, T. Mantecón, C. Díaz, C. R. Del-Blanco, F. Jaureguizar, and N. García, "A novel system for nighttime vehicle detection based on foveal classifiers with real-time performance," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 5421–5433, Jun. 2022.

[16] G. Wieczorek, S. B. U. D. Tahir, I. Akhter, and J. Kurek, "Vehicle detection and recognition approach in multi-scale traffic monitoring system via graph-based data optimization," *Sensors*, vol. 23, no. 3, p. 1731, Feb. 2023.

[17] J. S. Chang, J. Hwang, and M. Choi, "Vehicle detection approach adjusting road curves to estimate local traffic density under real driving conditions," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2677, no. 3, pp. 1382–1396, Mar. 2023.

[18] W. Liu, S. Liao, and W. Hu, "Towards accurate tiny vehicle detection in complex scenes," *Neurocomputing*, vol. 347, pp. 24–33, Jun. 2019.
[19] Y. Li, B. Li, B. Tian, and Q. Yao, "Vehicle detection based on the and-or graph for congested traffic conditions," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 984–993, Jun. 2013.
[20] S. Messelodi, C. M. Modena, and M. Zanin, "A computer vision system for the detection and classification of vehicles at urban road intersections," *Pattern Anal. Appl.*, vol. 8, nos. 1–2, pp. 17–31, Sep. 2005.
[21] M. S. Regio, G. V. Souza, R. Rosa, S. R. Musse, I. H. Manssour, and R. H. Bordini, "Truckfier: A multiclass vehicle detection and counting tool for real-world highway scenarios," in *Proc. 20th Int. Conf. Inf. Technol.-New Gener. (ITNG)*. Cham, Switzerland: Springer, 2023, pp. 341–350.
[22] Z. Dai, H. Song, X. Wang, Y. Fang, X. Yun, Z. Zhang, and H. Li, "Video-based vehicle counting framework," *IEEE Access*, vol. 7, pp. 64460–64470, 2019.
[23] M. S. Chauhan, A. Singh, M. Khemka, A. Prateek, and R. Sen, "Embedded CNN based vehicle classification and counting in non-laned road traffic," in *Proc. 10th Int. Conf. Inf. Commun. Technol. Develop.*, Jan. 2019, pp. 1–11.
[24] M. Anandhalli and V. P. Baligar, "A novel approach in real-time vehicle detection and tracking using raspberry pi," *Alexandria Eng. J.*, vol. 57, no. 3, pp. 1597–1607, Sep. 2018.
[25] R. Rios-Cabrera, T. Tuytelaars, and L. Van Gool, "Efficient multi-camera vehicle detection, tracking, and identification in a tunnel surveillance application," *Comput. Vis. Image Understand.*, vol. 116, no. 6, pp. 742–753, Jun. 2012.
[26] H. Song, H. Liang, H. Li, Z. Dai, and X. Yun, "Vision-based vehicle detection and counting system using deep learning in highway scenes," *Eur. Transp. Res. Rev.*, vol. 11, no. 1, p. 51, Dec. 2019.
[27] N. Singh, P. Saini, O. Shubham, R. Awasthi, A. Bharti, and N. Kumar, "Improved YOLOv5l for vehicle detection: An application to estimating traffic density and identifying over speeding vehicles on highway scenes," *Multimedia Tools Appl.*, vol. 83, no. 2, pp. 5277–5307, Jan. 2024.
[28] S. Hua, M. Kapoor, and D. C. Anastasiu, "Vehicle tracking and speed estimation from traffic videos," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 153–1537.
[29] A. Kumar, P. Khorramshahi, W.-A. Lin, P. Dhar, J.-C. Chen, and R. Chellappa, "A semi-automatic 2D solution for vehicle speed estimation from monocular videos," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 137–1377.
[30] T. Huang, "Traffic speed estimation from surveillance video data: For the 2nd NVIDIA AI city challenge track 1," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 161–1614.
[31] Z. Yang and L. S. C. Pun-Cheng, "Vehicle detection in intelligent transportation systems and its applications under varying environments: A review," *Image Vis. Comput.*, vol. 69, pp. 143–154, Jan. 2018.
[32] D. Meimetis, I. Daramouskas, I. Perikos, and I. Hatzilygeroudis, "Real-time multiple object tracking using deep learning methods," *Neural Comput. Appl.*, vol. 35, no. 1, pp. 89–118, Jan. 2023.
[33] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, Z. Yang, Y. Zhang, and D. Tao, "A survey on vision transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 87–110, Jan. 2023.
[34] A. Wang, H. Chen, Z. Lin, J. Han, and G. Ding, "RepViT: Revisiting mobile CNN from ViT perspective," 2023, *arXiv:2307.09283*.
[35] M. Gogebakan, "A novel approach for Gaussian mixture model clustering based on soft computing method," *IEEE Access*, vol. 9, pp. 159987–160003, 2021.
[36] M. Hao, W. Shi, H. Zhang, and C. Li, "Unsupervised change detection with expectation-maximization-based level set," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 1, pp. 210–214, Jan. 2014.
[37] J. Ke, A. J. Watras, J.-J. Kim, H. Liu, H. Jiang, and Y. H. Hu, "Efficient online real-time video stabilization with a novel least squares formulation and parallel AC-RANSAC," *J. Vis. Commun. Image Represent.*, vol. 96, Oct. 2023, Art. no. 103922.
[38] Y. Rao, Y. Yi, O. T. Nartey, and S. U. Jan, "Relevance gradient descent for parameter optimization of image enhancement," *Comput. Graph.*, vol. 117, pp. 124–133, Dec. 2023.
[39] B. Coifman and L. Li, "A critical evaluation of the next generation simulation (NGSIM) vehicle trajectory dataset," *Transp. Res. B, Methodol.*, vol. 105, pp. 362–377, Nov. 2017.

DANESH SHOKRI is currently pursuing the Ph.D. degree with Laval University, Québec City, Canada. During his academic journey, he has developed expertise in independent research, algorithm evaluation, and presenting results effectively. His work aims to bridge the gap between cutting-edge research and real-world applications in geomatics and AI. e geospatial datasets. In addition to his academic pursuits, he is currently a Scientific Researcher with the Centre de Géomatique du Québec (CGQ). He is responsible for developing cutting-edge methodologies, primarily based on artificial intelligence, to analyzing mobile LiDAR point clouds and designing innovative AI-driven solutions for geospatial applications. His research interests include advanced geospatial data analysis, computer vision, and artificial intelligence applications, particularly in object detection.

CHRISTIAN LAROUCHE received the Ph.D. degree in photogrammetry and remote sensing from the University of Calgary, Calgary, AB, Canada, in 1995.

He has worked for nearly 20 years in private companies, including Trimble Corporation. He has been a Professor with the Department of Geomatics Sciences and a Researcher with the Geospatial Data and Intelligence Research Center (CRDIG), Université Laval, since May 2015. His main research interests include mobile LiDAR and particularly on the development of various techniques for calibrating mobile LiDAR systems and for estimating the uncertainties related to the data acquired with these systems. He is currently the Director of the Bachelor of Science in Geomatics Program and the Geomatics Certificate Program, where he is also the Director of the Metrology Laboratory.

SAEID HOMAYOUNI (Senior Member, IEEE) received the B.Sc. degree in surveying and geomatics engineering from the University of Isfahan, Isfahan, Iran, in 1996, the M.Sc. degree in remote sensing and geographic information systems from Tarbiat Modares University, Tehran, Iran, in 1999, and the Ph.D. degree in signal and image from Télécom Paris Tech, Paris, France, in 2005.

From 2006 to 2007, he was a Postdoctoral Fellow with the Signal and Image Laboratory, University of Bordeaux Agro-Science, Bordeaux, France. From 2008 to 2011, he was an Assistant Professor with the Department of Surveying and Geomatics, College of Engineering, University of Tehran, Tehran. From 2011 to 2013, he was with the Earth Observation Group of the Agriculture and Agri-Food Canada, Ottawa Center of Research and Development, Ottawa, ON, Canada, through the Natural Sciences and Engineering Research Council of Canada Visitor Fellowship Program. In 2013, he was a Replacing Assistant Professor of remote sensing and geographic information systems with the Department of Geography, Environment, and Geomatics, University of Ottawa, Ottawa. Since April 2019, he has been an Associate Professor of environmental remote sensing and geomatics with the Centre Eau Terre Environment, Institut National de la Recherche Scientifique, Québec City, QC, Canada. He is currently leading a research group on earth observation analytics by artificial intelligence with interests in optical and radar Earth observations analytics for urban and agroenvironmental applications.