Hindawi
IET Software
Volume 2024, Article ID 5802816, 11 pages
https://doi.org/10.1049/2024/5802816

Research Article

An Expository Examination of Temporally Evolving Graph-Based Approaches for the Visual Investigation of Autonomous Driving

Li Wan and Wenzhi Cheng

School of Information Engineering, Hunan University of Science and Engineering, Hunan 425199, China

Correspondence should be addressed to Li Wan; wanli@huse.edu.cn

Received 2 September 2023; Revised 9 November 2023; Accepted 29 January 2024; Published 20 March 2024

Academic Editor: Vincenzo Moscato

With the continuous advancement of autonomous driving technology, visual analysis techniques have emerged as a prominent research topic. The data generated by autonomous driving is large-scale and time-varying, yet more than existing visual analytics methods are required to deal with such complex data effectively. Time-varying diagrams can be used to model and visualize the dynamic relationships in various complex systems and can visually describe the data trends in autonomous driving systems. To this end, this paper introduces a time-varying graph-based method for visual analysis in autonomous driving. The proposed method employs a graph structure to represent the relative positional relationships between the target and obstacle interferences. By incorporating the time dimension, a time-varying graph model is constructed. The method explores the characteristic changes of nodes in the graph at different time instances, establishing feature expressions that differentiate target and obstacle motion patterns. The analysis demonstrates that the feature vector centrality in the time-varying graph effectively captures the distinctions in motion patterns between targets and obstacles. These features can be utilized for accurate target and obstacle recognition, achieving high recognition accuracy. To evaluate the proposed time-varying graph-based visual analytic autopilot method, a comparative study is conducted against traditional visual analytic methods such as the frame differencing method and advanced visual analytic methods like visual lidar odometry and mapping. Robustness, accuracy, and resource consumption experiments are performed using the publicly available KITTI dataset to analyze and compare the three methods. The experimental results show that the proposed time-varying graph-based method exhibits superior accuracy and robustness. This study offers valuable insights and solution ideas for developing deep integration between intelligent networked vehicles and intelligent transportation. It provides a reference for advancing intelligent transportation systems and their integration with autonomous driving technologies.

1. Introduction

The acceleration of urbanization has rendered automobiles indispensable for daily commuting, enhancing travel efficiency yet concurrently engendering a host of societal challenges, including escalating road accidents, urban traffic congestion, and environmental degradation [1, 2]. According to statistics, 83%–94% of traffic accidents are related to human fault [3]. The advent of autonomous vehicles and the advancement of intelligent network infrastructure have been identified as pivotal strategies to ameliorate these issues. Autonomous vehicles, emblematic of a novel transportation paradigm, amalgamate capabilities such as environmental sensing, decision-making, and control execution [4–6]. A critical hurdle in autonomous driving is the real-time extraction and analysis of intricate visual data. Vision analytics-based methodologies have gained traction as a viable resolution, given their capacity to synthesize and interpret diverse data modalities, encompassing images, videos, Lidar, and radar data [7–10]. Additionally, these approaches can model temporal dependencies in traffic scenarios, a crucial element for forecasting and strategizing the trajectory of autonomous vehicles. Visual analytics methods entail the deployment of sensors, including onboard cameras and Lidar, to capture road-related information, subsequently facilitating the processing and interpretation of data to achieve comprehensive perception of the vehicular environment. Despite the significant progress that has been made, current

visual analytics methods still have some limitations and challenges. For example, existing visual analytics methods must be more stable for target tracking and obstacle detection at different speeds. Traditional visual analytics methods fail under adverse weather conditions, such as dense fog, heavy rain, or snow. Current visual analytics methods may need to be improved in terms of real-time performance, especially in complex scenes. Therefore, there is a need to investigate an advanced visual analytics method for visualization. Among various visual analytics techniques, the time-varying graph is a prevalent visualization instrument that can depict data trends postsensor acquisition [11]. For instance, it can be employed to monitor vehicle trajectories, thereby enabling the calculation of parameters such as current vehicle position and orientation. Hence, the potential for applying time-varying graphs in vision analysis techniques is substantial. In light of this, the present study introduces a time-varying graph-based visual analytics method for autonomous driving. It conducts comparative experiments juxtaposing traditional (frame differencing method (FD)) and advanced (visual lidar odometry and mapping (V-LOAM)) visual analytics methods. The comparative analysis reveals that the proposed time-varying graph-based approach exhibits stable localization precision across varying driving distances and vehicle speeds and a consistent total feature node count in each scenario, demonstrating robustness. The proposed algorithm exhibits superior accuracy, robustness, and resource utilization relative to the alternative vision analysis algorithms.

The innovations of this paper are as follows:

(1) A method based on time-varying graphs is introduced for visual analysis in automatic driving.
(2) Using the knowledge of graph theory and the characteristics of complex transportation networks, the time-varying graphs of target and obstacle interference are established, and the edges of the graph measure the relationship between nodes.
(3) Static and time-varying graphs of targets and interference are established to analyze and compare the changes in the characteristics of various graphs, and the changes in the characteristics of such graphs provide a strong basis for the classification of targets and interference.

2. State of the Art

2.1. Study of Time-Varying Graph Concept and Its Model. Time-varying graphs, alternately termed dynamic or temporal graphs, are either directed or undirected graphs delineating each node and its corresponding edges within a specified time slice [12]. These graphs encapsulate time-dependent graphs wherein the states of nodes or edges undergo alterations over time, thereby inducing modifications in their topological relationships. In autonomous driving, time-varying graphs can encapsulate data pertinent to the traffic milieu, encompassing vehicular position, velocity, and directional alterations [13]. By processing and analyzing time-varying graphs, one can attain a more nuanced understanding of the prevailing traffic conditions, enhancing the performance and precision of autonomous driving systems.

Traditionally, the characterization of network topology models has been approached as a static graph, merely depicting the connectivity relationships between nodes and the properties of the links. This conventional approach fragments the interrelations between network topologies across distinct time intervals, thereby underutilizing network resources. Time-varying graphs address this limitation. Time-varying graph models have evolved through multiple phases, delineating several quintessential models below.

The incipient time-varying graphs, denominated as snapshot graphs [14], are founded on the principle of discretizing the network topology across the temporal dimension to yield multiple network topologies corresponding to distinct time intervals, thereby transmuting the dynamic network model into a static one. This transformation facilitates the resolution of time-varying network issues via traditional static graph-solving techniques. The snapshot graph severs the interrelations between the dynamic topologies of each time interval, precluding the collective scheduling of static topologies across individual time slots.

The time-extended graph model [15] is predicated on correlating the topologies of each time slot in the snapshot graph with the storage resources of the nodes. Subsequently, the network topologies of each time slot are interrelated via the storage edges of the nodes, thereby converting multiple static graphs into a singular static graph for resolution. This model effectively interconnects network topologies across disparate time slots along the temporal dimension, optimizing resource utilization. Although the network necessitates segmentation to construct the time-extended graph, the storage of its topology still demands substantial resources. Nonetheless, at the algorithmic stratum, this model significantly enhances the accuracy and computational complexity of the problem solution relative to snapshot graphs.

The temporal aggregation graph model [16] is underpinned by the compression of the snapshot graph through the aggregation of link resource representations across diverse time intervals. Within a single graph, links are characterized via aggregation, with each element in the set denoting the attribute of the link for the corresponding time interval. This model, characterized by superior algorithmic performance, also addresses the issue of extensive storage resources necessitated by the time-extended graph and snapshot graph topology storage.

The storage time aggregation graph model [17] addresses the absence of constraint relationships between storage and link resources in the temporal aggregation graph. This deficiency culminates in diminished accuracy while resolving the maximum flow problem. Consequently, an enhanced version of the temporal aggregation graph termed the storage temporal aggregation graph model, is proposed. This model amalgamates the merits of the temporal aggregation graph and significantly bolsters network capacity.

2.2. Study of Autonomous Driving Vehicle Obstacle Avoidance Trajectory. Following the stages of target detection and tracking, it remains imperative to forecast the future trajectory of a

target to precisely evaluate the hazard level posed by dynamic obstacles within the traffic milieu and to formulate a judicious path. The ongoing advancements in artificial intelligence have progressively deepened its research applications in the trajectory planning domain of autonomous driving.

Liu and Shi [18] leveraged the features of a BP neural network for self-learning to construct a lane change trajectory model using actual vehicle lane change trajectory data. Gan et al. [19] employed a local planning method with a rolling window to accommodate environmental information alterations. Subsequently, a heuristic algorithm was implemented in the local planning process for critical path point processing during lane changes, followed by a transition arc to generate the lane change trajectory seamlessly. Ding et al. [20] incorporated B-samples to reprogram the curvature discontinuity phenomenon of lane change trajectories, thereby reconstructing the free lane change model. Xin et al. [21] executed efficient operations within the octree data structure to generate an online free-space flight corridor comprised of overlapping three-dimensional grids. This was followed by generating a smooth trajectory adhering to higher-order dynamic constraints. Xiong et al. [22] reformulated the path planning issue as a quintessential quadratic planning problem predicated on optimization. This method facilitated the real-time generation of 3D optimal paths adhering to input constraints, thereby efficaciously addressing the trajectory generation challenge in tightly constrained environments.

Zhang et al. [23] employed a unified approach to encapsulate environmental semantic information, thereby establishing a novel dynamic constraint composed of collision-free cubes, denoted as a spatiotemporal semantic corridor structure. Additionally, segmented Bessel curve parameterization ensured the safety, collision-free nature, and dynamic compliance of the generated trajectories. Ding et al. [24] proposed a genetic algorithm (GA) optimized long short-term memory (LSTM) prediction model. The GA algorithm optimized the LSTM model's initial parameters, expedited model convergence, and circumvented poor convergence attributable to random initial parameters, thereby enhancing the model's predictive performance. Roy et al. [25] introduced a vehicle trajectory prediction model amalgamating extreme learning machine (ELM) and deep neural networks, thereby enhancing the stochasticity of the ELM and addressing the network generalization issue. Li et al. [26] developed a spatiotemporal graph attention neural network-based vehicle trajectory prediction model capable of capturing and quantifying the interaction information of neighboring vehicles across time and space. The Uber R&D team proposed a convolutional neural networks-based vehicle trajectory prediction model [27]. Zhao et al. [28] proposed a generative adversarial networks-based model tailored for road intersection scenarios. Li et al. [29] presented a dual-learning model, and that integrates lane occupancy and risk maps for vehicle trajectory prediction. The aim is to address the challenges observed in existing deep learning-based vehicle trajectory prediction models, including computational complexity, environment-specific dependence, and neglect of vehicle interactions. Finally, based on the effectiveness of graph neural networks (GNN) [30] in node, edge, and graph classification, Jo et al. [31] presented a hierarchical GNN-based behavior prediction model known as Vector Net.

3. Methodology

In this section, a time-varying graph of target and obstacle interference is built using graph theory knowledge and complex traffic network properties to measure the relationship between nodes regarding graph edges. The time dimension is introduced to portray the appearance and disappearance of obstacle disturbances, and the changes in the characteristics of various graphs are analyzed and compared to provide a strong basis for the classification of targets and obstacles.

3.1. Graph Theory Knowledge. This subsection introduces the basic concepts of formal graph theory that support the methodology of this paper, thereby explaining how to use graph theory to construct time-varying graphs to analyze target and obstacle interference in an autonomous driving environment.

The graph $A = (Q, E)$ is composed of points $q \in Q$ and edges $e \in \mathrm{e} \subseteq Q \times Q$. The weights of the edges e_{xy} are denoted as $m(x, y)$. In constructing the graph, the targets and disturbances in the image are used as the graph nodes. The relationship between the nodes will be converted into an adjacency matrix, and the weights of the edges are obtained using the distances between the nodes.

(1) The adjacency matrix of the graph

If the nodes are connected in the graph, the corresponding position in the matrix is 1. Otherwise, it is 0.

$$G(x, y) = \begin{cases} 1 & e_{xy} \in E \\ 0 & e_{xy} \notin E \end{cases}. \tag{1}$$

(2) Weight of the graph

$$m(x, y) = \begin{cases} d(x, y) & G(x, y) = 1 \\ 0 & G(x, y) = 0 \end{cases}, \tag{2}$$

where $d(x, y) = \sqrt{(i_x - i_y)^2 + (j_x - j_y)^2}$ is the distance between two nodes x and y.

(3) Degree of a node

The degree of a node is the number of nodes connected to that node, i.e.,

$$D(x, y) = \deg(x) = \sum_{y \in Q} G(x, y). \tag{3}$$

(4) Laplacian matrix of the graph

The Laplacian matrix L of the graph is calculated from the degree matrix D and the adjacency matrix G of the graph, $L = D - G$. That is,

$$L(x,y) = \begin{cases} D(x,y) & x = y \\ -1 & e_{xy} \in E \\ 0 & \text{otherwise} \end{cases}. \quad (4)$$

The Laplace matrix is non-negative symmetric, and thus L is a full rank matrix, so there will be t eigenvalues $\lambda_1 \leq \lambda_2 \leq$, …, $\leq \lambda_{|Q|}$, corresponding to $|Q|$ orthogonal eigenvectors ϕ_1, $\phi_2, \ldots, \phi_{|Q|}$, whose corresponding Laplace spectral decomposition is as follows:

$$L = \sum_{x=1}^{|Q|} \lambda_x \phi_x \phi_x^N. \quad (5)$$

The eigenvector corresponding to the smallest non-zero eigenvalue is called the Fielder vector.

(5) Eigenvector centrality

The feature vector portrays the characteristics of a node and can be used to measure the node's importance in the network. Suppose

$\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_t$ are the T eigenvalues of the adjacency matrix G, and λ is the principal eigenvalue of G, whose corresponding eigenvector is $e = [e_1, e_2, \ldots, e_t]^N$. It is easy to obtain $\lambda e_x = \sum_{y=1}^{T} g_{xy} e_y, x = 1, \ldots, t$, and the measure of the node q_x feature vector is defined as follows:

$$C_e(q_x) = \lambda^{-1} \sum_{y=1}^{T} g_{xy} e_y. \quad (6)$$

In the following subsection, we will delve into the specific procedures and methodologies used to apply these graph theory principles to create our time-varying graph.

3.2. Time-Varying Graph. In this subsection, a time-varying graph is constructed based on graph theory knowledge by introducing the time dimension. Then, the time-varying graph is used to portray the dynamic motion process of targets and obstacles.

The graph model consists of target and obstacle disturbances and is a time-varying system where the relationship of nodes changes over time. The dynamics of the system can be described by a time-varying graph with $\mathcal{G} = (Q, E, \mathcal{T}, \rho, \zeta)$, where,

(1) $\rho : E \times \mathcal{T} \to \{0, 1\}$ called the survival function, whether an edge (relation) exists at a certain moment.
(2) $\zeta : E \times \mathcal{T} \to \mathbb{T}$ called the delay function, from a given moment, the time required to cross a given edge.

The delay function can be defined for a given period, on a given edge, or ignored, and when it is ignored, the time-varying graph is defined as $\mathcal{G} = (Q, E, \mathcal{T}, \rho)$.

Both the target and the obstacle are moving, and they are imaged at different positions in each image frame, which are different but belong to the same node. We cannot reflect the correspondence of nodes at different moments by using only static maps. Establish the time matrix of nodes (F, Q, i, j). Consider it as a representation of a discrete system: F is the frame number, Q is the label of each node, and (i, j) are the image coordinates of the nodes. The node-time matrix relates the nodes in different frames of a time sequence from one to the other. The frame number is a known image sequence number, and it is easier to extract the node positions, while how to label the corresponding nodes is the difficult part of building the node-time matrix.

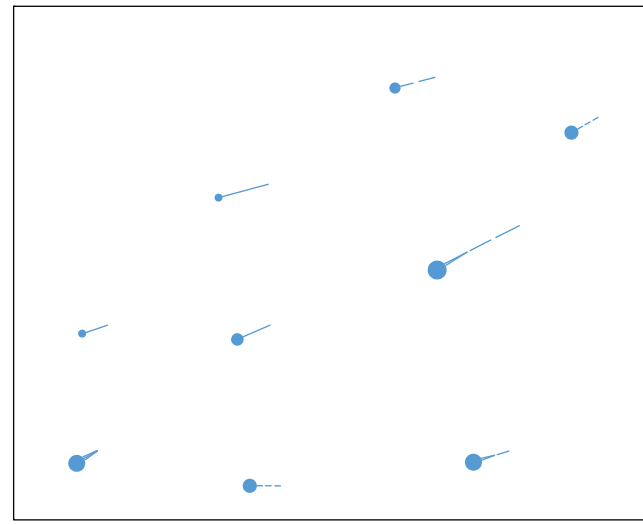


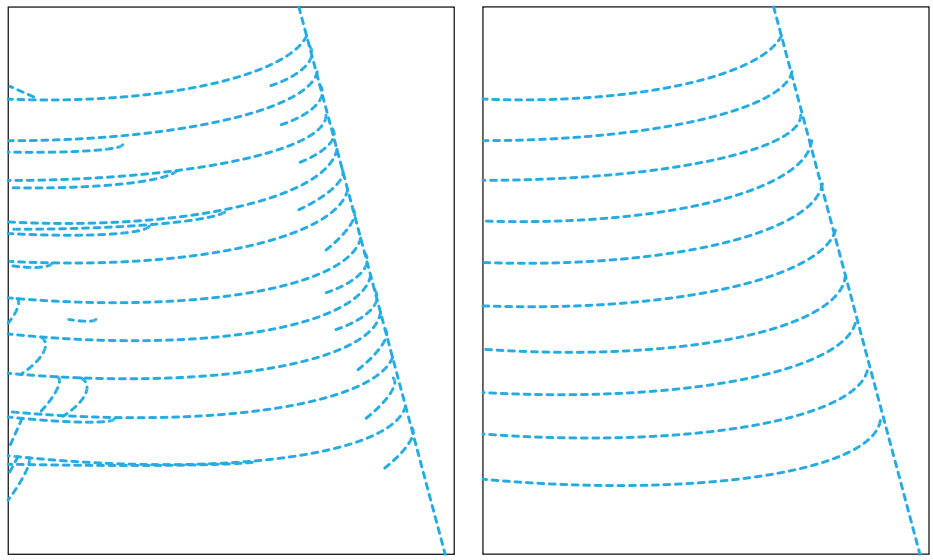
Figure 1: Schematic diagram of trailing with obstacle interference.

The tail of the obstacle interference will produce sparks, and these sparks may not be connected with the interference subject at a later stage during imaging, as shown in Figure 1.

We plot the extracted nodes in one image, and we can see that the same node will form a trajectory, as shown in Figure 2(a). From the figure, we can see many invalid nodes at the end of the trajectory, which can cause great interference with the association of the trajectory. They can cause the trajectory to be disconnected and marked with wrong numbers, making the characteristics of the nodes inconsistent with the actual ones. Therefore, we need to merge these invalid nodes. From Figure 2(a), we see that the invalid nodes are mainly densely distributed at the end of the trajectory, so we consider using the distance threshold to merge the nodes that are smaller than the distance threshold. The distances of all neighboring nodes in each frame are counted, and the distances of neighboring nodes in all frames are sorted from smallest to largest. Assume that this distance sequence d has a size of T_g. Pick a distance threshold in percentage u.

$$Nbu = d\left(T_g \times u\right). \quad (7)$$

Each image frame constructs a distance matrix D with T nodes and a matrix size of T^*T. $D(x, y)$ is the distance between nodes x and y. D is a symmetric matrix whose only upper triangular part needs to be used; the rest of the positions can be set to infinity to select pairs of nodes whose node distances are less than the threshold.



(a) (b)

Figure 2: Node diagram of all frames before and after merging invalid nodes: (a) before; (b) after.

$$M\text{node} = \{(x, y) | D(x, y) < Nb\}. \tag{8}$$

We need to consider not only the node pairs that are smaller than the threshold but also the node pairs that are adjacent to these node pairs. The center of mass of these nodes is taken as the new valid node, i.e., nodes x, y, z belong to the same connected domain s, there are t_S nodes, and the center of mass of these nodes is found as follows:

$$\text{New\_node} = \left\{ \left(\frac{\sum_{k=1}^{t_s} i_k}{t_s}, \frac{\sum_{k=1}^{t_s} j_k}{t_s} \right) | \, k \leftarrow x, y, z, \ldots, \in s \right\}. \tag{9}$$

The processed nodes are plotted in the same image, as shown in Figure 2(b). Compared with Figure 2(a) without processing, it can be seen that the invalid nodes are significantly reduced.

According to the above rules, a distance threshold is needed to determine whether a node is a vanishing node.

$$D_F = \left\{ \min\left(d_{x,z}\right) | x = 2, \ldots, t; z = 1, \ldots, T_x \right\}. \tag{10}$$

If two nodes in adjacent frames match, the distance between them will not exceed the maximum value of D_F. Then, the distance threshold for determining the disappearance of a node can be set as follows:

$$Nbf = \max(D_F). \tag{11}$$

The node matching is performed between two adjacent frames, and the distance between nodes must be less than the distance threshold Nbf. The image sequence is processed sequentially, and the final node number distribution is shown in Figure 3. It can be seen that node 1 exists in every frame, and it is the target node. There are 11 remaining nodes, 10 of which are more evenly distributed and have more nodes. It coincides with the number of trajectories in Figure 2(b), which are all 10.

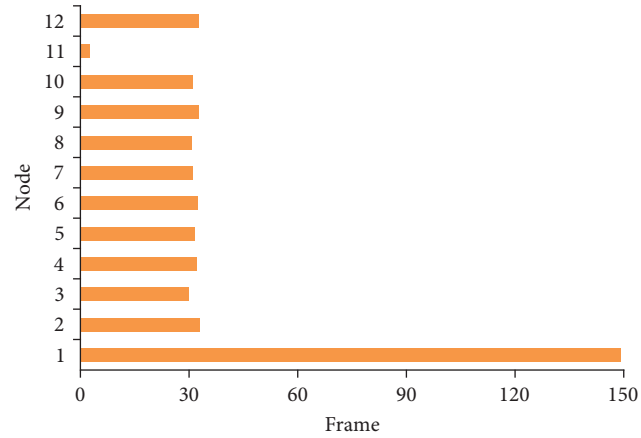


Figure 3: Node number distribution.

Counting the frame numbers of the nodes, we can learn that their frame numbers are consecutive, as shown in Table 1. After determining the number of nodes, the time matrix of nodes (F, Q, i, k) can be constructed.

In the following subsection, we will delve into using time-varying graph properties for target and obstacle recognition.

3.3. Time-Varying Graph Characteristics Analysis. Unlike the static diagram, the time-varying diagram is built based on the time matrix of nodes. From the above node time matrix, we see 11 nodes left after excluding the interfering nodes. For each frame, a graph structure consisting of 11 nodes is created. If the current frame node does not exist, then the

Table 1: Node frame number statistics.

Node number	1	2	3	4	5	6	7	8	9	10	11	12
Start frame number	1	33	43	52	61	69	79	88	97	106	111	114
End frame number	149	65	73	84	92	100	109	118	129	136	112	145

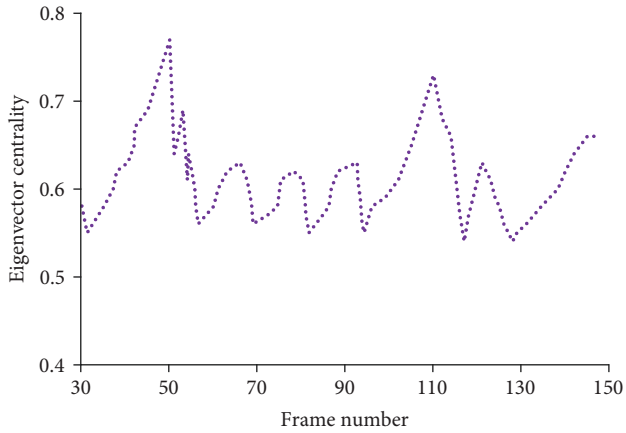


Figure 4: Target eigenvector centrality.

connection weight of this node with all other nodes is 0. Then, the adjacency matrix of the time-varying graph is as follows:

$$G(x,y) = \begin{cases} 1 & x,y \in a_F \\ 0 & \text{otherwise} \end{cases}, \tag{12}$$

where a_F is the set of nodes in the Fth frame, and the corresponding weight matrix is as follows:

$$M(x,y) = \begin{cases} \sqrt{(i_x - i_y)^2 + (j_x - j_y)^2} & x,y \in a_F \\ 0 & \text{otherwise} \end{cases}. \tag{13}$$

For each frame in the image, the adjacency matrix and the weight matrix established by the nodes are different, and the temporal changes of each node can be observed after the nodes are associated.

Figures 4 and 5 show the eigenvector centrality metrics of the target and interfering nodes, respectively. It can be seen from the figures that the changes in the eigenvector centrality indexes of the interference nodes are the same and different from those of the target nodes.

Figure 5 shows that the eigenvector centrality of interference nodes 1 and 10 is significantly different from that of other nodes. This is because interference 1 is the first; there is no antecedent interference, and only new nodes will be added subsequently, so the overall trend is decreasing. Interference 10 is the last, and there is no subsequent interference, and its precursors will disappear one after another, so the first half of the curve shows a decreasing trend while the second half is an increasing trend. Through the above analysis, the feature vector centrality of this indicator can distinguish the target and obstacle interference.

4. Result Analysis and Discussion

This paper selects the Odometry scene ensemble of the KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute) dataset as the test subject. The KITTI dataset is a widely used benchmark dataset for computer vision tasks, especially autonomous driving. Researchers collected this dataset to support the development and evaluation of algorithms for tasks such as object detection, tracking, stereo, and 3D scene understanding in the context of autonomous driving. It has become a standard benchmark for evaluating the performance of algorithms in real-world, challenging driving scenarios. This dataset contains real data sequences of 11 scenes, each with and without true values. It covers typical road scenes such as suburban roads, highways, and city roads, and the visual ranging sequence covers 39.2 km of road scenes. Each scene data contains an image, Lidar point cloud, and real bit pose data sequences, a common data set for testing the localization map building algorithm.

4.1. Robustness Experiments. In real-world autonomous driving scenarios, algorithms must adapt to different environments and provide stable results to ensure the vehicle's safety. Therefore, robustness experiments in various environments are conducted in this paper.

The first experiment on robustness is to test the algorithm's ability to adapt to different environments by the number of feature nodes it extracts. Specifically, the number of visual feature nodes extracted by the time-varying graph-based vision algorithm in this paper is examined separately for different environments, and the results are compared with the V-LOAM and the FD. In order to more visually compare the differences between the three algorithms in extracting features in different scenes, the number of features extracted by each algorithm in each scene as a percentage of the total number of features extracted in all scenes is plotted, as shown in Figure 6.

The 2nd experiment is to take the average translation error and rotation error of the first 800 m in each scene with nodes at 100 m intervals (see Figures 7 and 8). As seen from Figure 7, the translation error of the V-LOAM algorithm increases with the length. This is because the algorithm does not detect loopback, leading to serious drift when the path is too long. In contrast, the translation error and rotation error of the proposed algorithm and FD decrease more smoothly with the increase in length. Compared with FD, the proposed algorithm is smoother. In addition, the

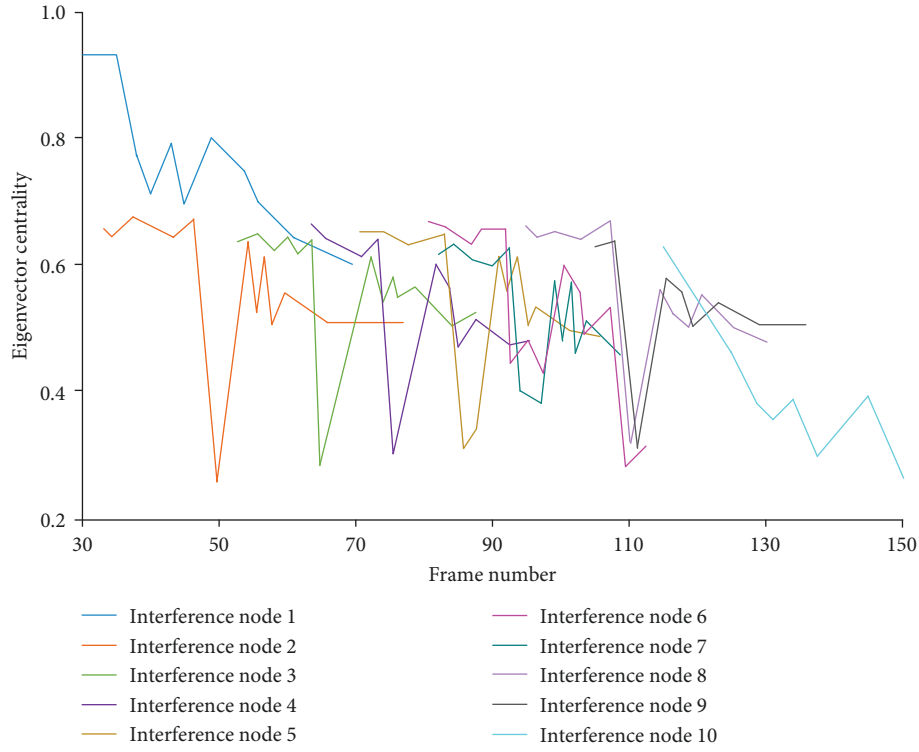


Figure 5: Interference 1–10 eigenvector centrality.

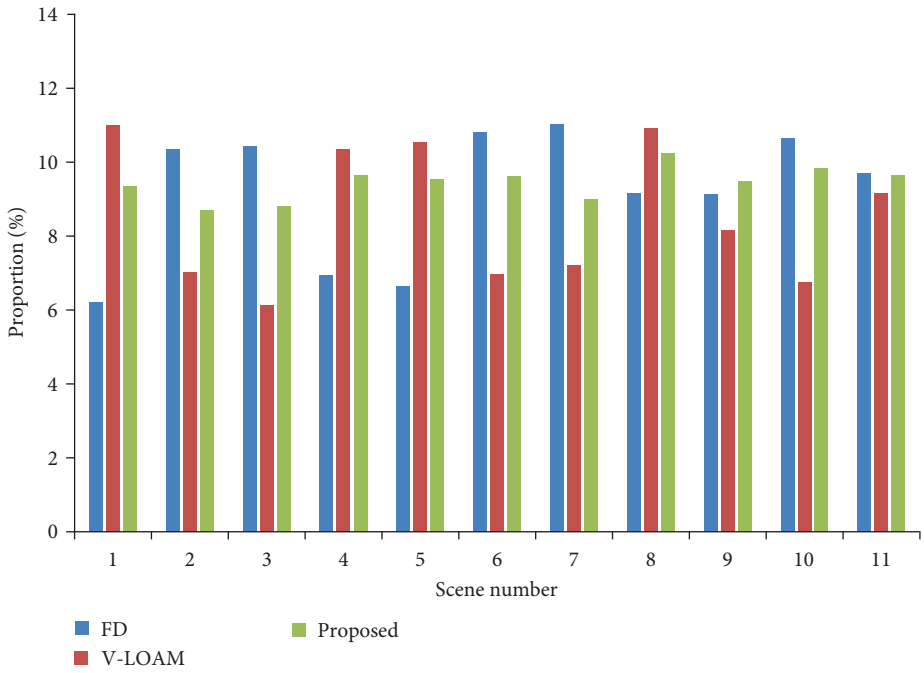


Figure 6: Number of feature nodes extracted by each algorithm as a percentage of their total number.

errors of the algorithm at different vehicle speeds were also tested.

As can be seen in Figure 8, all three algorithms change to varying degrees as the vehicle speed increases. Regarding translation error, the FD algorithm causes a large error variation, while the proposed algorithm's and V-LOAM error changes are more stable. Compared with V-LOAM, the variation of the proposed algorithm is more stable and stays within the range of 1.0–1.2. Regarding rotation error, the fluctuation amplitude of both FD and V-LOAM is

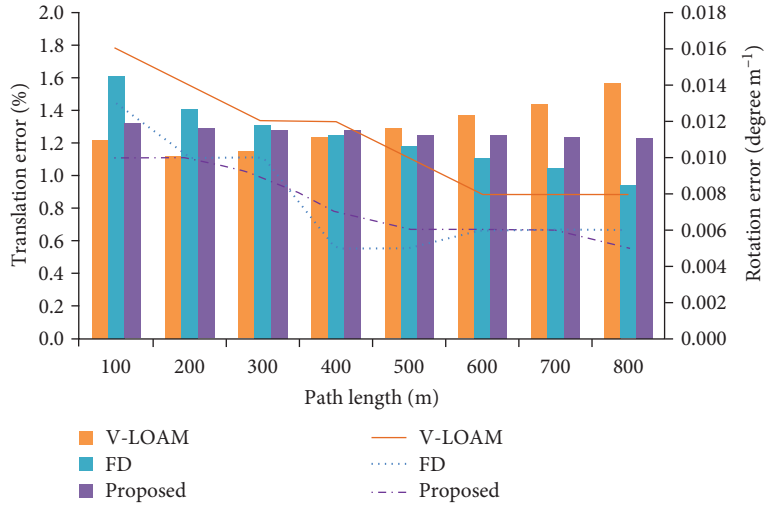


Figure 7: Average translation and rotation errors of the three algorithms at different path lengths.

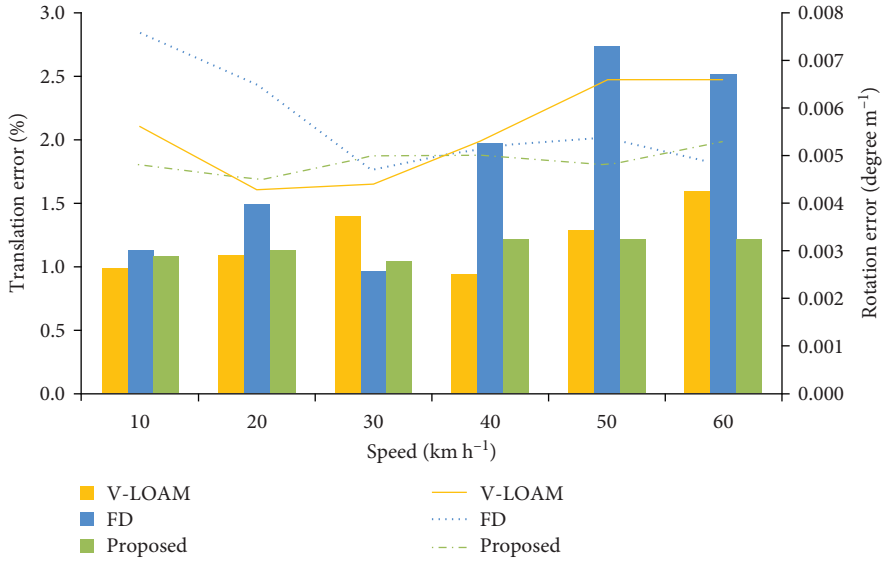


Figure 8: Average translation and rotation errors of the three algorithms at different speeds.

relatively large. In contrast, the variation amplitude of the proposed algorithm is smaller. The results prove that the algorithm in this paper has better stability performance.

In summary, the proposed algorithm has better adaptability and stability in different environments, has stronger anti-interference ability in the face of different vehicle speed changes, and higher robustness of the algorithms.

4.2. Accuracy Experiment. Precise localization and trajectory estimation are crucial for safe and reliable autonomous driving. In this paper, the accuracy experiments aim to verify the algorithm's accuracy and applicability in self-driving car applications.

The above three algorithms are run in 11 scenarios from Odometry01 to Odometry11. Then, the output poses are compared with the real trajectories to calculate absolute pose error (APE) and relative pose error (RPE), and the standard deviation, root means square error, and mean value are output. The output error values of each algorithm in the 11 scenarios are averaged (see Table 2).

Table 2 shows that in the translation error, the FD algorithm performs the worst compared to the other two algorithms. The mean values of the errors of the proposed algorithm are 26.99 and 0.09, which are lower than those of V-LOAM (32.98 and 0.25). In the rotation error, the mean values of the errors of the proposed algorithm are 0.146 and 0.002, which are also lower than those of FD and V-LOAM. The combined accuracy error of the proposed algorithm is the smallest compared with the other two algorithms.

4.3. Resource Occupancy Test Experiment. Efficient resource utilization is crucial for in-vehicle autonomous driving systems as they usually have limited computational resources. Therefore, studying CPU occupancy is crucial to evaluate the

Table 2: Average error of the three algorithms in all scenarios.

Category	Evaluation indicator	Translation error			Rotation error		
		FD	V-LOAM	Proposed	FD	V-LOAM	Proposed
APE	Standard deviation	121.14	28.76	20.99	0.005	0.008	0.006
	Root mean square	312.44	38.38	34.18	0.325	0.174	0.155
	Mean value	286.66	32.98	26.99	0.313	0.188	0.146
RPE	Standard deviation	1.12	0.33	0.14	0.005	0.004	0.002
	Root mean square	1.83	0.57	0.26	0.006	0.008	0.003
	Mean value	1.48	0.25	0.09	0.004	0.006	0.002

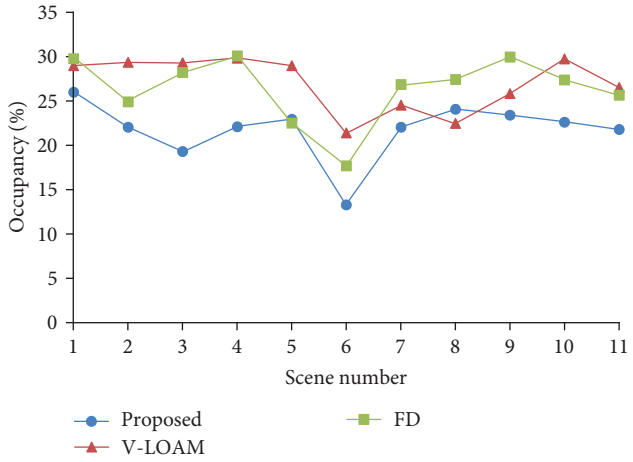


Figure 9: Comparison of CPU occupation rate of algorithms.

algorithms' computational efficiency and determine how it affects the overall performance of the self-driving car system.

In addition to the localization module, the autonomous driving system must also be equipped with target detection, path planning, and other modules. Therefore, with the limited computational resources, the computational volume of the algorithms should be reduced as much as possible, and the algorithms' performance should be guaranteed simultaneously. The average CPU (I7-7700HQ, 2.80 GHz × 4) usage of the computer is tested and compared for the three algorithms running in each environment from sequence01 to sequence11. The test method is to record the CPU occupancy rate every 5 s while running and then take the average value. The algorithm CPU occupancy results are shown in Figure 9.

According to Figure 9, the average CPU occupancy of the proposed algorithm is 21.8%, while the average CPU occupancy of FD is 26.4% and V-LOAM is 27%. It shows that the proposed algorithm is much lower than the other two. This is because the total number of feature nodes extracted by the proposed algorithm is smaller, and the optimization strategy at the back end reduces the computation. Therefore, the time-varying graph-based algorithm proposed in this paper has a lower resource occupation rate while having higher accuracy and robustness, and it is more suitable for vehicle autonomous driving systems.

5. Conclusion

Autonomous driving represents a burgeoning technology that has recently garnered considerable focus. Vision-centric analytical methodologies are prevalently employed in autonomous driving because they assimilate and scrutinize diverse data types in real time. As a visual analytical instrument of vision technology, time-varying graphs can be utilized in the autonomous driving domain to analyze vehicle motion trajectories, identify obstacles, etc. Consequently, this research introduces a time-varying graph-centered approach for the visual analysis of autonomous driving. The proposed technique is juxtaposed against the conventional vision analysis method (FD) and the advanced visual analysis method (V-LOAM) and is examined utilizing the KITTI dataset. The research findings demonstrate that the time-varying graph-centered method delineated in this paper exhibits superior accuracy and robustness. This investigation can facilitate the steering of the evolution of more efficacious and precise autonomous driving systems. It can serve as a benchmark for other application methodologies predicated on vision analysis in the transportation sector.

Data Availability

The labeled dataset used to support the findings of this study is available from the corresponding author upon request.

Disclosure

To complete the research work, we have received academic guidance and assistance from our unit and a small amount of financial support to help complete the research project.

Conflicts of Interest

The author of this article states that there are no conflicts of interest with the unit academically.

Acknowledgments

This work was supported in part by the construct program of applied characteristic discipline in Hunan Province and the Project of Hunan Provincial Natural Science Foundation of China (Grant No. 2023JJ50421). The author acknowledges that some content in this article was translated and polished by ChatGPT (https://chat.openai.com/) to improve the

quality of English writing. We sincerely thank ChatGPT for its contribution.

References

[1] F. Zhao, Z. Mu, H. Hao et al., "Hydrogen fuel cell vehicle development in China: an industry chain perspective," *Energy Technology*, vol. 8, no. 11, Article ID 2000179, 2020.

[2] F. O. Okeke, A. E. Okosun, C. A. Udeh, and C. J. Okekeogbu, "Cities for people: the dependency & impact of automobile in the life of city dwellers," *European Journal of Sustainable Development*, vol. 9, no. 3, Article ID 157, 2020.

[3] D. Xiao, M. Dianati, W. G. Geiger, and R. Woodman, "Review of graph-based hazardous event detection methods for autonomous driving systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 5, pp. 4697–4715, 2023.

[4] S. Manoharan, "An improved safety algorithm for artificial intelligence enabled processors in self-driving cars," *Journal of Artificial Intelligence and Capsule Networks*, vol. 1, no. 2, pp. 95–104, 2019.

[5] J. Ni, Y. Chen, Y. Chen, J. Zhu, D. Ali, and W. Cao, "A survey on theories and applications for self-driving cars based on deep learning methods," *Applied Sciences*, vol. 10, no. 8, Article ID 2749, 2020.

[6] S. Jain and I. Malhotra, "A review on obstacle avoidance techniques for autonomous driving vehicle," *International Journal of Advanced Science and Technology*, vol. 29, no. 6, pp. 5159–5167, 2020.

[7] S. Jamonnak, Y. Zhao, X. Huang, and M. Amiruzzaman, "Geo-context aware study of vision-based autonomous driving models and spatial video data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 1, pp. 1019–1029, 2022.

[8] M. Lei, D. Yang, and X. Weng, "Integrated sensor fusion based on 4D MIMO radar and camera: a solution for connected vehicle applications," *IEEE Vehicular Technology Magazine*, vol. 17, no. 4, pp. 38–46, 2022.

[9] H. Fujiyoshi, T. Hirakawa, and T. Yamashita, "Deep learning-based image recognition for autonomous driving," *IATSS Research*, vol. 43, no. 4, pp. 244–252, 2019.

[10] T. Zhou, M. Yang, K. Jiang, H. Wong, and D. Yang, "MMW radar-based technologies in autonomous driving: a review," *Sensors*, vol. 20, no. 24, Article ID 7283, 2020.

[11] Y. Zhao, L. Ge, H. Xie et al., "ASTF: visual abstractions of time-varying patterns in radio signals," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 1, pp. 214–224, 2023.

[12] D. B. Tay and J. Jiang, "Time-varying graph signal denoising via median filters," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 3, pp. 1053–1057, 2021.

[13] H. Pang, N. Liu, C. Hu, and Z. Xu, "A practical trajectory tracking control of autonomous vehicles using linear time-varying MPC method," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 236, no. 4, pp. 709–723, 2022.

[14] F. Bauzá Mingueza, M. Floría, J. Gómez-Gardeñes, A. Arenas, and A. Cardillo, "Characterization of interactions' persistence in time-varying networks," *Scientific Reports*, vol. 13, Article ID 765, 2023.

[15] D. Hu, S. Zhang, and A. M. Zou, "Velocity-free fixed-time attitude cooperative control for spacecraft formations under directed graphs," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 8, pp. 2905–2927, 2021.

[16] I. Maduako and M. Wachowicz, "A space-time varying graph for modelling places and events in a network," *International Journal of Geographical Information Science*, vol. 33, no. 10, pp. 1915–1935, 2019.

[17] T. Zhang, J. Li, H. Li, S. Zhang, P. Wang, and H. Shen, "Application of time-varying graph theory over the space information networks," *IEEE Network*, vol. 34, no. 2, pp. 179–185, 2020.

[18] M. Liu and J. Shi, "A cellular automata traffic flow model combined with a BP neural network based microscopic lane changing decision model," *Journal of Intelligent Transportation Systems*, vol. 23, no. 4, pp. 309–318, 2019.

[19] N. Gan, M. Zhang, B. Zhou, T. Chai, X. Wu, and Y. Bian, "Spatio–temporal heuristic method: a trajectory planning for automatic parking considering obstacle behavior," *Journal of Intelligent and Connected Vehicles*, vol. 5, no. 3, pp. 177–187, 2022.

[20] Y. Ding, W. Zhuang, L. Wang, J. Liu, L. Guvenc, and Z. Li, "Safe and optimal lane-change path planning for automated driving," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 235, no. 4, pp. 1070–1083, 2021.

[21] L. Xin, Y. Kong, S. E. Li et al., "Enable faster and smoother spatio–temporal trajectory planning for autonomous vehicles in constrained dynamic environment," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 235, no. 4, pp. 1101–1112, 2021.

[22] L. Xiong, Z. Fu, D. Zeng, and B. Leng, "An optimized trajectory planner and motion controller framework for autonomous driving in unstructured environments," *Sensors*, vol. 21, no. 13, Article ID 4409, 2021.

[23] T. Zhang, W. Song, M. Fu, Y. Yang, X. Tian, and M. Wang, "A unified framework integrating decision making and trajectory planning based on spatio–temporal voxels for highway autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 10365–10379, 2022.

[24] W. Ding, J. Huang, G. Shang et al., "Short-term trajectory prediction based on hyperparametric optimisation and a dual attention mechanism," *Aerospace*, vol. 9, no. 8, Article ID 464, 2022.

[25] B. Roy, M. P. Singh, M. R. Kaloop et al., "Data-driven approach for rainfall-runoff modelling using equilibrium optimizer coupled extreme learning machine and deep neural network," *Applied Sciences*, vol. 11, no. 13, Article ID 6238, 2021.

[26] J. Li, H. Ma, Z. Zhang, J. Li, and M. Tomizuka, "Spatio-temporal graph dual-attention network for multi-agent prediction and tracking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 10556–10569, 2022.

[27] B. Yang, X. Cao, K. Xiong et al., "Edge intelligence for autonomous driving in 6G wireless system: design challenges and solutions," *IEEE Wireless Communications*, vol. 28, no. 2, pp. 40–47, 2021.

[28] C. Zhao, Y. Zhu, Y. Du, F. Liao, and C.-Y. Chan, "A novel direct trajectory planning approach based on generative adversarial networks and rapidly-exploring random tree," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 17910–17921, 2022.

[29] Z. Li, P. Zhao, C. Jiang, W. Huang, and H. Liang, "A learning-based model predictive trajectory planning controller for automated driving in unstructured dynamic environments," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 6, pp. 5944–5959, 2022.

[30] V. La Gatta, V. Moscato, M. Postiglione, and G. Sperli, "An epidemiological neural network exploiting dynamic graph structured data applied to the COVID-19 outbreak," *IEEE Transactions on Big Data*, vol. 7, no. 1, pp. 45–55, 2020.

[31] E. Jo, M. Sunwoo, and M. Lee, "Vehicle trajectory prediction using hierarchical graph neural network for considering interaction among multimodal maneuvers," *Sensors*, vol. 21, no. 16, Article ID 5354, 2021.