When, Where and How does it fail? A Spatial-temporal Visual Analytics Approach for Interpretable Object Detection in Autonomous Driving

Junhong Wang, Yun Li, Zhaoyu Zhou, Chengshun Wang, Yijie Hou, Li Zhang, Xiangyang Xue, Michael Kamp, Xiaolong (Luke) Zhang, and Siming Chen

Abstract—Arguably the most representative application of artificial intelligence, autonomous driving systems usually rely on computer vision techniques to detect the situations of the external environment. Object detection underpins the ability of scene understanding in such systems. However, existing object detection algorithms often behave as a black box, so when a model fails, no information is available on *When, Where* and *How* the failure happened. In this paper, we propose a visual analytics approach to help model developers interpret the model failures. The system includes the *micro-* and *macro*-interpreting modules to address the interpretability problem of object detection in autonomous driving. The *micro*-interpreting module extracts and visualizes the features of a convolutional neural network (CNN) algorithm with density maps, while the *macro*-interpreting module provides spatial-temporal information of an autonomous driving vehicle and its environment. With the situation awareness of the spatial, temporal and neural network information, our system facilitates the understanding of the results of object detection algorithms, and helps the model developers better understand, tune and develop the models. We use real-world autonomous driving data to perform case studies by involving domain experts in computer vision and autonomous driving to evaluate our system. The results from our interviews with them show the effectiveness of our approach.

Index Terms—Autonomous driving, spatial-temporal visual analytics, interpretability

1 INTRODUCTION

Autonomous driving has grown rapidly in recent years and has the potential to change transportation systems and even human society significantly. An autonomous driving system uses various artificial intelligence (AI) systems from a range of AI technologies. While AI algorithms make autonomous driving possible, there are still many challenges in their development.

One such challenge is that most AI algorithms are inherently black boxes that provide little information that would explain their decision-making process. Consequently, model development becomes a daunting task. For example, to develop a model for 3D object detection, an essential function in most autonomous driving systems is to understand the external environment. During the process, developers often need to manually examine a model for prediction errors and then to fine-tune the model. This is usually an inefficient process, in particular, because it is difficult for model developers to understand the exact circumstances under which a model error occurs and what caused it.

Research efforts on explainable AI (XAI), or interpretable AI, try to make the behaviors of AI algorithms more transparent to humans by deriving explanations intelligible to humans [21]. To help people better understand a neural-network-based model, for instance, information related to neuron activation in the model can be provided. Visualization has been used in XAI research: For example, to support the microscopic interpretation of convolutional neural networks (CNNs), Zeiler and Fergus [52] proposed a deconvolution and unpooling method to show the feature maps of each layer in a CNN so that the high activation value of the feature maps in the validation set can be easily observed.

However, approaches of this type focus largely on the algorithms themselves without considering the contexts including external environment, car states and so on, in which the algorithms are used. For example, the 3D object detection model in autonomous driving is sensitive to the environment and the ways a car moves. Model developers must consider both model features and the external factors in specific scenarios.

One way to address the challenge of lacking context information is to use visual analytics, which can integrate AI algorithms and visualization of various contextual information to support exploration-based analysis. Recently, research by Gou et al. [20] combined disentangled representation learning and semantic adversarial learning to help analysts understand and improve a signal light detection model. However, since autonomous driving happens in a spatial-temporal environment, it is important for **model developers in autonomous driving** to understand the spatial-temporal contexts of models. To our knowledge, visual analytics research on spatial-temporal features of AI models in autonomous driving is rare.

In this work, we design a visual analytics system to combine spatial-temporal information and features in neural networks to analyze 3D object detection models at two levels. This helps model

- *J. Wang, Y. Li, Z. Zhou, C. Wang, Y. Hou, L. Zhang, X. Xue and S. Chen are with School of Data Science, Fudan University. S. Chen is the corresponding author. E-mail: {junhongwang, yunli, zhaoyuzhou, chengshunwang, yijiehou, lizhangfd, xyxue, simingchen}@fudan.edu.cn.*
- *M. Kamp is with the Institute for AI in Medicine (IKIM) at UK Essen, Ruhr-University Bochum, and the Department of Data Science and AI, Faculty of IT, Monash University. E-mail: michael.kamp@uk-essen.de.*
- *X. Zhang is with Pennsylvania State University. E-mail: lzhang@ist.psu.edu.*

Manuscript received April 19, 2005; revised August 26, 2015.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2022.3201101

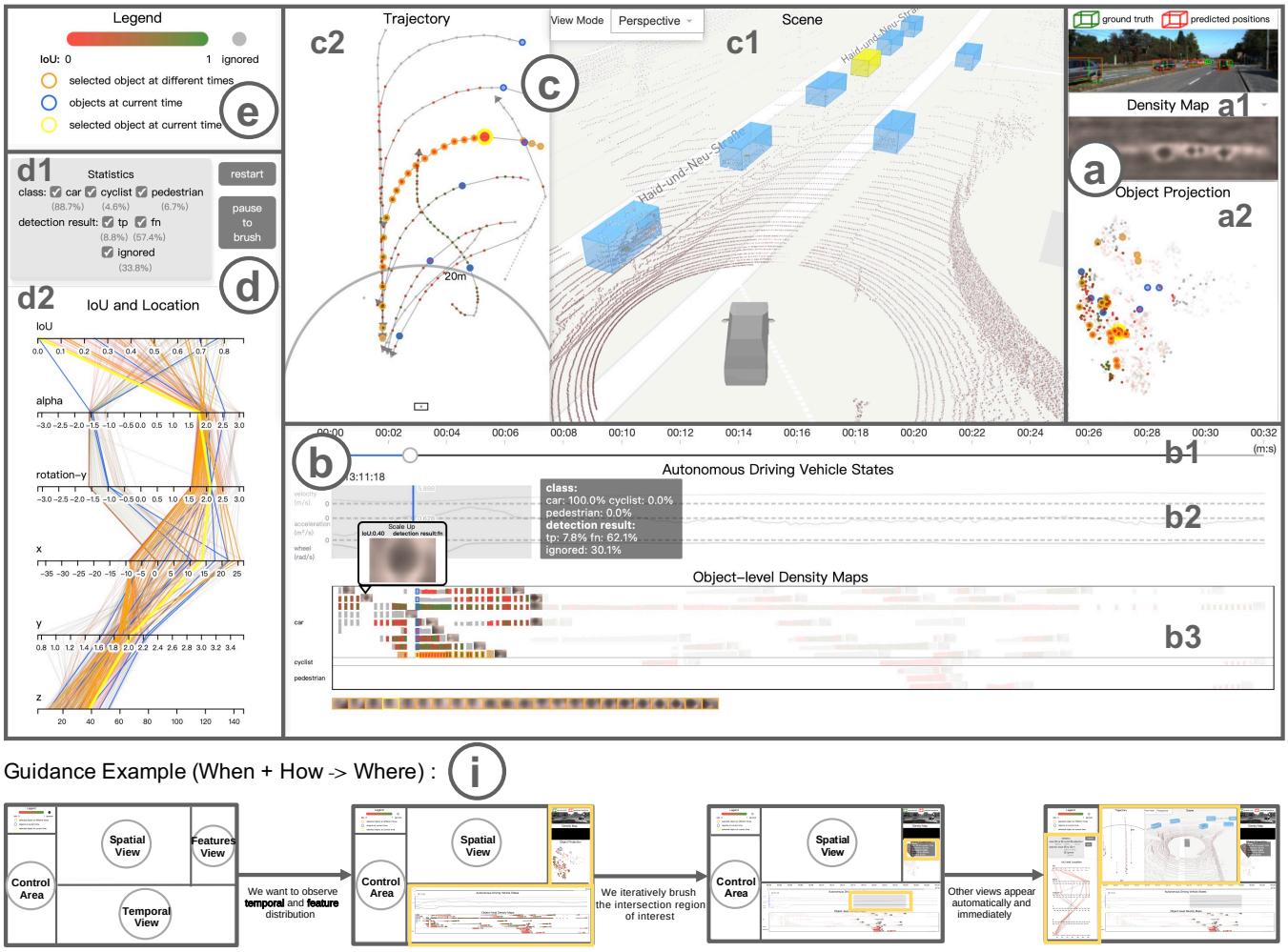


Fig. 1. System User Interface: (a) *Micro*-interpreting module: feature visualization including *Density Map* and *Object Projection*; (b) *Macro*-interpreting module: temporal visualization including *Autonomous Driving Vehicle States* and *Object-level Density Maps*; (c) *Macro*-interpreting module: spatial visualization including *Scene* and *Trajectory*; (d) Control for selecting and filtering object classes, results and locations. (e) A legend referring to all views introduces color encoding. (f) A guidance example to use our analysis workflow.

developers gain deeper understandings of the performances of AI models, in particular in failure scenarios. At the *micro* level, our system visualizes the predictions and ground truths of a trained detection model based on a sequence of input images. At the *macro* level, the system presents the time and space scenes of the autonomous driving sequence. With information from these two levels, our system can show the interaction between the spatial-temporal scene and feature visualization, reveal the correlation between them and the performances of the corresponding detection, and help model developers better interpret the model results from the three important perspectives: *When, Where* and *How*.

Our contribution can be summarized as the following:

- **An Interpretable Method Integrated with Spatial-temporal Information and CNN Features**: We designed a visual analysis method to combine *macro*-level spatial-temporal information and *micro*-level CNN features, so that they can be explored independently or jointly through interactive visualization tools.
- **An Interpretable Visual Analytics Framework for Object Detection in Autonomous Driving**: Aiming at autonomous driving, we provide a framework to analyze the results of object detection algorithms visually and interactively, with rich functionality and interpretability.
- **A Visual Analytics System Improving the Work Efficiency of Model Developers**: We provide a system prototype to interpret *When, Where* and *How* the model fails for model developers. Such insight would help them augment data or adjust the network structure to improve models more efficiently.

2 Related Work

In this section, we review literature related to object detection in autonomous driving, visual analytics for spatial-temporal information, and interpretability for machine learning.

2.1 Autonomous Driving and Object Detection

Autonomous driving has attracted more and more attention recently. Yurtsever et al. [51] outlined existing challenges in autonomous driving, and comprehensively summarized recent progress in perception, mapping, localization, planning, and human-machine interface modules. They also introduced datasets available in the development of autonomous driving systems, such as the PASCAL

VOC dataset [16], the KITTI Vision Benchmark [18] and the Oxford RobotCar [36].

As a fundamental and important task in computer vision, object detection has been widely studied and applied. CNNs have been successfully applied in object detection tasks. Often seen techniques can be categorized into two-stage and single-stage approaches. The former methods, such as Fast R-CNN by Girshick [19], produce the region proposal first, and then classify and regress the proposals to the ground-truth bounding boxes. The second group, including YOLO by Redmon et al. [43] and SSD by Liu et al. [35], directly predict the object category and location without adopting proposal learning. As an important part in the scene perception module, object detectors have been widely used in autonomous driving.

In addition to 2D object detection models mentioned above, 3D object detection models have also been proposed for better perception in the 3D space. Existing prevalent approaches like VoxelNet by Zhou and Tuzel [54] for 3D object detection largely rely on LiDAR, which provides accurate 3D point clouds of the scene. Although promising, LiDAR-based approaches require expensive equipment which severely limits their scalability and applicability. Cheaper alternatives, such as the camera-based methods by Wang et al. [50], have been developed to directly predict location, dimension, and orientation in 3D space alongside the category of objects, given only RGB input images. Ding et al. [14] proposed a dynamic depth-guided local convolutional network to detect 3D objects based on depth maps extracted from monocular camera images, and promoted the development of 3D object detection in autonomous driving.

Although these methods offer promising performances for object detection, their high complexity and low interpretability posechallenges for model developing. Thus, interpreting tools are needed. In this paper, we aim to propose a spatial-temporal visual analytics approach to address the above problem.

2.2 Spatial-temporal Visual Analytics

Peuquet el al. [42] indicated that analysis of spatial-temporal data is often conducted from three perspectives: where, when and what, and spatial-temporal data analysis tasks are around them according to Andrienko et al. [7]. To support these tasks, researchers have developed various query techniques, such as P-Query by Chen et al. [12]), R-Query (regions) by Jeung et al. [28], and T-Queries (trajectories) by Lee et al. [31]. Some visual analysis methods, including space-time cube by Liu et al. [32], time-space slice by Ngo et al. [40] and animation methods by Andrienko et al. [7], have also been explored to support in-depth analysis.

Due to the complexity of spatial-temporal data structure and mapping, visualization of spatial-temporal data often focuses on issues related to overlapping and neglecting multi-dimensional and multi-records data. Researchers explored different methods to reduce the number of multi-records. For example Guo [22] used tools like filtering, and Kisilevich et al. [30] considered aggregation. In addition, sampling and spatial-temporal clustering can also be used. For the reduction of the overlapping of lines and points, approaches like bundling by Buchin et al. [9] and multiple views by Ferreira et al. [17] can help to link multiple data dimensions. In early urban visualization systems, Butkiewicz et al. [10] argued that 3D views of urban models can provide an intuitive perception of spatial information, and Chang et al. [11] suggested that information visualization designs can provide information on different dimensions of interest.

Visual analytics has been widely used in fields such as transportation and population flow to support the exploration of spatial-temporal relationships of data. For example, Chen et al. [13] built an uncertainty-aware visual analytics system for the analysis of human behaviors from heterogeneous spatial-temporal data, and the work by Andrienko et al. [6] is a system to reveal patterns and trends of mass mobility through spatial and temporal abstraction of origin-destination movement data. Sorger et al. [47] summarized the taxonomy of integrating spatial and non-spatial visualizations, which confirms the foundation of our approach. Ortner et al.'s work [41] demonstrates the effectiveness of integrated visualization for urban planning. However, none of the existing work has been integrated with the analysis of autonomous driving, and we mainly expect to apply spatial-temporal visual analysis methods to autonomous driving.

In autonomous driving, data is usually a sequence involving driving time and location information. Some open-source tools like Worldview [3], Autonomous Visualization System (AVS) [2] and Dreamview [1] can support visualization-based exploration, verification and presentation of spatial-temporal data involved in autonomous driving. However, these tools are largely strong in the presentation of relevant data, but weak in in-depth analysis. Gou et al. [20] proposed VATLD, a visual analytics system for traffic light detection, took a different approach and offered tools for in-depth data and model analysis. However, because of its focus on traffic lights, rather than diverse objects in environment, VATLD offers limited support for the analysis of data features from different levels, which are important to accurately understanding and diagnosing a model involving more diverse objects. Recently, Achberger et al. [5] proposed a visual analytics tool to help engineers analyze test drive videos annotated with navigation-specific augmented reality content. He et al. [24] proposed a visual analytics approach to diagnose and improve the accuracy of semantic segmentation on critical objects moving in various driving scenes. Jamonnak et al. [27] proposed a geo-context aware approach based on large spatial video data to study vision-based autonomous driving models focusing on the final action decisions. Hou et al. [26] proposed a visual evaluation method for the autonomous driving system. Although these work provided new insight for autonomous driving visual analytics, there is no work addressing the interpretability issue of object detection for autonomous driving, with the focus on combining the spatial-temporal information for visual analytics, which is our research contribution.

2.3 Interpretability for Machine Learning

Various visual analytics systems have been developed to support the understanding and improvement of machine learning models, especially those based on neural networks, which are often seen as a black box. Zeiler and Fergus [52] proposed a method to show the feature map of a model to help users better understand the role of each layer in a CNN and the importance of features in the original data through visualization. This method has been widely used in CNN interpretability research, such as in Simonyan et al. [46], which used saliency maps.

In addition to these map-based approaches, other methods have also been proposed. Abadi et al. [4] designed visualization tools for data flow, the change of network parameters and the model accuracy with TensorBoard, so that users can see the model structure and the parameter change process in training. Spinner et al. [48] developed explAIner to realize understanding,

diagnosis, refinement, prevention tracking and reporting of models through plug-ins in TensorBoard. Furthermore, model-specific visual analytics systems have also been developed for model interpretation. For deep generative models, the training process is critical. Liu et al. [33] developed DGMTracker to find abnormal neurons or neural layers by observing the temporal changes of data in the data flow. Systems like CNNVis by Liu et al. [34] and RNNVis by Ming et al. [38] are used to visualize and understand CNN and RNN.

Visual analytics systems for various kinds of neural networks can help model developers to further understand, diagnose and optimize models. Hohman et al. [25] summarized current visual analytics systems for machine learning.

In autonomous driving, the interpretability of a specific model has attracted more and more attention, with the aim of explaining, diagnosing and optimizing it. Bojarski et al. [8] developed VisualBackProp to visualize which part of the input images contributes most to explaining the prediction results of the model and further to help debug deep CNN-based autonomous driving models. Kim and Canny [29] visualized attention heat maps to show where a model may focus on. Zeng et al. [53] applied interpretability to the planning module of autonomous driving, and generated the output of a single module through a neural network to improve the interpretability of autonomous driving decisions. Mori et al. [39] introduced the attention branch network and used attention maps to analyze the decision-making rationales in autonomous driving. In addition to research on interpretability in computer vision, using visual analytics for object detection has also been proposed. VATLD by Gou et al. [20] combined disentangled representation learning and semantic adversarial learning to help model developers better understand and improve the accuracy and robustness of a model.

The complexity of machine-learning models demands tools to improve the interpretability of object detection. One of the bottlenecks of object detection lies in the difficulty of summarizing under what situation models may fail. Existing research largely focuses on an individual perspective, while a more comprehensive analysis is required to design autonomous driving algorithms by integrating information from different perspectives: *When, Where* and *How*.

Drawing on research on the interpretability of autonomous driving by computer vision researchers, we hope to develop a method to enhance the interpretability of object detection algorithm results by combining the high-level spatial-temporal information of the environment, or macroscopic information, with the features of data and model, or microscopic information. Our research is one of the first attempts to integrate such macroscopic and microscopic information to improve the interpretability of algorithms in autonomous driving.

3 Overview of Our Method

Following the approach of Sedlmair et al. [44], we designed our system following three stages: design requirement analysis for model development, system design and implementation, and system validation. For the first stage of design requirement analysis, we performed empirical studies involving domain experts and a literature analysis on autonomous driving (Section 4). The research outcome in this stage is a list of design requirements that the system should support. In the second stage of system design and implementation, we focused on the design of interactive visualization tools to support these requirements (Sections 5 and 6). As third stage we validated our system by using two autonomous driving scenarios as case studies to show how our system works (Section 7) and then conducted interviews with domain experts to collect their feedback on the usability and effectiveness of the system (Section 8).

4 Design Requirement Analysis

In this section, we first give an overview of our work on design requirement analysis. Then, we provide the fundamentals of object detection in autonomous driving. Finally, we present the design goal, data abstraction and the design requirements developed by involving domain experts, following and adapting the design rules developed by Miksch and Aigner [37].

4.1 Overview of Design Requirement Analysis

Research in this stage included several parts. Firstly, we collaborated with an industrial expert (P1) to learn about the work of model developers. The expert is from an autonomous driving company focusing on the perception phase of autonomous driving for over 10 years. At the same time, we conducted an analysis of literature relevant to their work. Through the interactions with P1 and literature analysis, we firstly gained knowledge about the work of model developers listed in Section 4.1. We identified a set of major concerns that model developers have had in their work listed in Section 4.3, and then inferred the goals of a system to address these concerns.

Design requirements are based on the goals and validated by domain experts. We drafted a list of design requirements and invited four domain experts, including P1, to evaluate them. The requirements were revised based on their feedback. After several iterations of the feedback and revision of the requirements, we finalized them with the consensus of all experts.

4.2 Object Detection in Autonomous Driving

Object Detection in Autonomous Driving. The basic task of object detection is object recognition and localization. In benchmarks for object detection in autonomous driving, such as KITTI Vision Benchmark [18] and Oxford RobotCar [36], objects of interest are often classified into these main categories: car, van, truck, pedestrian, and cyclist. The input to a model is a set of images captured by the camera, and the outputs include the following results of each object detected by the model: the position of an object in both 2D picture and 3D camera coordinate systems, the object class, occlusion and truncation information, 3D information, object angle, and the confidence level of detection. Inside the object detection model, the output of the convolution filter applied to the previous layer is called feature map, and each convolution kernel corresponds to one feature map.

Evaluation Metrics of Detection Results. The evaluation of the effectiveness of detection algorithms involves several concepts. The actual labeling information of an object is called ground truth, and the 2D or 3D border of an object on the 2D plane is called a bounding box. The correctness of object localization is determined by comparing the degree of overlap, or Intersection over Union (IoU), between a predicted bounding box and a ground truth bounding box with a threshold (e.g., 0.5); the correctness of object identification is determined by comparing the confidence score with a threshold (e.g., 0.7).

The measure of object detection correctness can be written as:

$$\alpha_{ij} = \frac{\text{area}\,(b_{ij} \cap b_{gt})}{\text{area}\,(b_{ij} \cup b_{gt})} \tag{1}$$

$$z_{ij} = \begin{cases} 1, & if\ \alpha_{ij} > thres \\ 0, & otherwise \end{cases} \tag{2}$$

where α_{ij} represents IoU, b_{ij} refers to the j-th detection bounding box of the object in the i-th image, and b_{gt} is the ground-truth bounding box of the object. *thres* denotes the chosen threshold. z_{ij} is the detection result: 1 means a correct detection, and 0 an erroneous one, which in this paper we define as a "case of failure".

The final evaluation metric of a model in autonomous driving is often a class-specific AP value [15], referring to the average precision corresponding to different recall values (confidence thresholds) for a specific object class. An AP value is calculated as:

$$r(t) = \frac{\sum_{ij} \mathbb{1}\,[s_{ij} \geq t]\, z_{ij}}{N} \tag{3}$$

$$p(t) = \frac{\sum_{ij} \mathbb{1}\,[s_{ij} \geq t]\, z_{ij}}{\sum_{ij} I\,[s_{ij} \geq t]} \tag{4}$$

$$AP = \frac{1}{M} \sum_{r\left\{0, \frac{1}{M-1}, \frac{2}{M-1} \cdots, 1\right\}} \max_{R \geq r} p(R) \tag{5}$$

where N is the number of ground truth on the image of a given class; s_{ij} is the confidence score of each detection; t is the threshold of confidence score; r and p are the function of the recall and precision corresponding to the confidence threshold, respectively; M is the number of equally spaced r used to calculate AP; and R is the value of r corresponding to the actual confidence level of detection.

4.3 Problem Abstraction

Drawing on interactions with P1 and literature analysis, we identified a set of concerns of model developers when they evaluate object detection models: the relationship between the times of failure, the locations of the cases of failure, the performances of the network features in the algorithm, and the synergy of those scenarios that lead to detection failures.

Based on such understandings, we summarized our **design goals** as to help developers understand *When, Where* and *How* an object detection algorithm fails.

- **When** refers to temporal analysis of failures: In a given period or timestamp, what is the state of the autonomous vehicle: accelerating, decelerating, driving speed, turning, straightening, etc.?
- **Where** concerns spatial analysis: What are the geographic locations, road situations, and surroundings of the autonomous driving vehicle and the positions of objects?
- **How** is about model analysis: How was an object detected by the model, with data and evidences from the model?

In addition, the relationships among these three questions and their joint contributions to detection results are also important.

4.4 Data Characterization

The data related to object detection tasks in autonomous driving includes two main categories: raw data and model output. Raw data includes tracklets, GPS, point clouds and images. Model output includes features and detection results. More specifically, tracklets refer to classes, sizes (length, width and height) and locations of labeled objects which are ground truth in object detection. GPS data records the latitude, longitude, speed and other driving status data of the autonomous driving vehicle. Point clouds are detected by the vehicle's LiDAR. Images are taken by the vehicle's camera. The images are input to the model, and then the model outputs the detection results. Features inside the model can also be extracted. Based on our goals, we abstracted relevant data into three categories: model data, spatial data, and temporal data. Model data refers to images inputs, the features of a model, and detection results. Spatial data includes point clouds and tracklets. Temporal data is related to vehicle state data from GPS and object sequences from tracklets.

4.5 Design Requirement Validation

To develop a visual analytics system to support model developers, We converted the goals into a set of three design requirements with three sub-bullets. These requirements focus on the support for user exploration to answer these three questions, jointly or individually.

With the drafted design requirements, we invited four domain experts to evaluate and validate them. In addition to P1, we interviewed three more experts in object detection and autonomous driving. Among them, one is a senior specialist in computer vision (P2), one is engaged in research on 2D/3D vision perception of autonomous driving scenes (P3), and one on object detection of autonomous driving (P4). They all have more 10 years of experience in their field.

During the interview, each of four experts was given our design requirements, and asked to rate them as "Acceptable", "Need Improvement", or "Unacceptable", with justifications. Based on their ratings and justifications, we revised the requirements and asked them to rate the revision. After several rounds of rating and revising, we obtained the finalized design requirements after all experts rated all requirements as "Acceptable".

4.6 Design Requirements

The finalized requirements are as the following:

R1: Answer the questions When, Where and How individually.

- R1.1: Explore the timestamps of a time series the failure cases are mainly distributed at, and the states of the autonomous driving vehicle when failure cases happen (**When**).
- R1.2: Explore the surroundings, positions of failure cases, especially the relative positions to the camera (**Where**).
- R1.3: Observe whether the feature maps of failure cases exhibit abnormalities compared to normals (**How**).

R2: With one question set, explore the other two.

- R2.1: In a whole sequence of images, for objects poorly detected over a period of interest, observe their surroundings and relative positions to the autonomous driving vehicle, their detection results, abnormalities of the feature maps, and the relationship between the detection results, the spatial location and the features (**When** $\rightarrow$ **Where and How**).
- R2.2: Similarly, for objects detected in a distance from the autonomous driving vehicle, observe the timestamps they are distributed at, whether the feature maps of failure cases exhibit abnormalities compared to the normals, the detection results of the objects, and the relationship between the spatial distribution and the other three: temporal distribution, feature distribution and detection results (**Where** $\rightarrow$ **When and How**).
- R2.3: For objects with abnormal feature maps, show their temporal information, their spatial information such as relative positions to the autonomous driving vehicle and surroundings, as

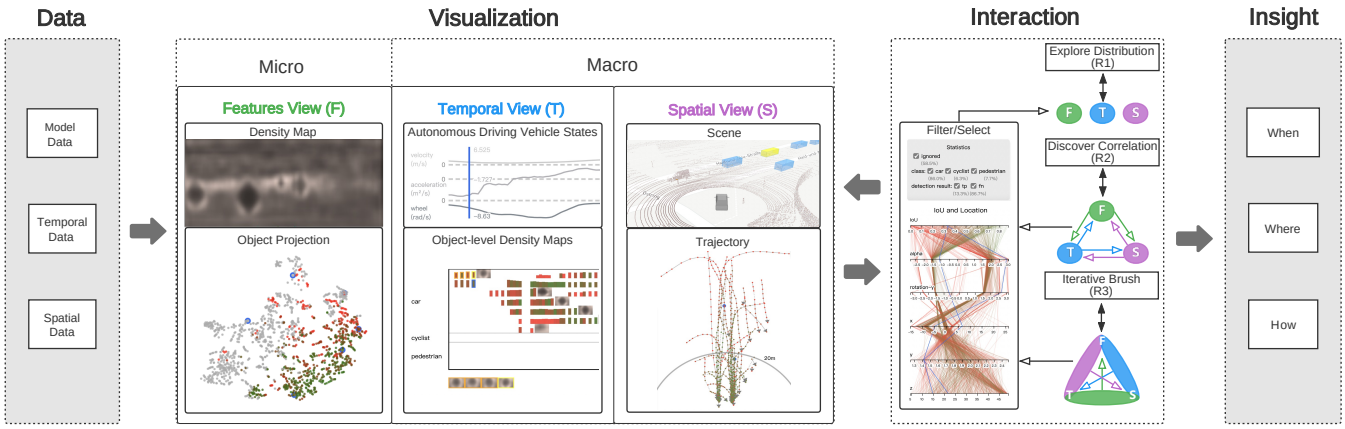


Fig. 2. Workflow of our visual analytics system. Model data, temporal data and spatial data are ported into our system through two modules – *micro* and *macro*, separately, and visualized in three views – features, temporal and spatial views. A control panel lets users conduct filtering and selecting operations. Three interaction methods are provided to support three analysis requirements and gain insights into *When, Where* and *How* questions on the cases of failure. Explore Distribution means choosing the control panel as an overview and exploring the details of the other three views by filtering. Discover Correlation means selecting one of the views as an overview and then further exploring the information of the other two views and the control panel. And Iterative Brush means selecting two, then exploring the other and the control panel.

well as their detection results. Moreover, the relationship between them (**How** → **When and Where**).

R3: With two questions set, explore the other one.

- R3.1: In a sequence of images, for objects detected in a spatial-temporal range of interest, observe whether the feature maps of failure cases exhibit abnormalities compared to the normals (**When and Where** → **How**).
- R3.2: Similarly, for objects both in a temporal range and feature distribution region or feature map characteristics of interest, observe their relative positions to the autonomous driving vehicle and surroundings, their detection results, and the relationship between the two (**When and How** → **Where**).
- R3.3: For objects in a spatial range and feature distribution region or feature map characteristics of interest, observe the timestamps they are distributed at, the states of the autonomous driving vehicle, their detection results, and the relationship between the results and temporal information (**Where and How** → **When**).

5 Visual Analytics Design

Following the requirements we designed a visual analytics system (Figure 1) with the proposed exploration workflow (Figure 2). To better help users explore the system, we provide guidance in Figure 1-f for interactive exploration. We have annotations on each view to guide users choose some view, which can be one or two according to the design requirements, as an overview. Then zoom in and filter, other views would be shown accordingly. In this section, we describe the details of our design.

5.1 *Micro*-interpreting Module: Feature Visualization

The *micro*-interpreting module focuses on the visualization of the features from the CNN model. Feature information inside the model is processed and visualized as *Density Map* (Figure 1-a1) and *Object Projection* (Figure 1-a2).

5.1.1 Feature Processing Method

The features of each layer output inside the model are abstract, and developers can hardly learn anything from massive numbers.

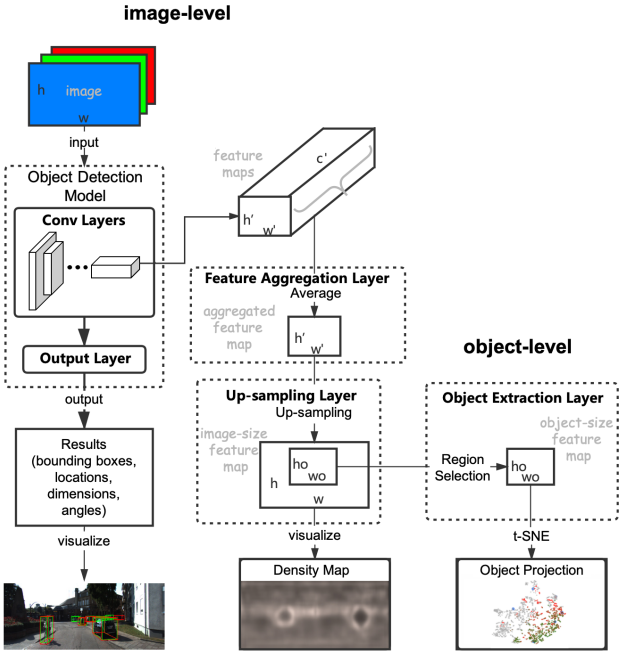


Fig. 3. The processing flow of features. Image-level and object-level processing methods to visualize features from the model.

To address this issue, we developed image-level and object-level processing methods (Figure 3) to visualize the features from the model, and to obtain the *Density Map* (Figure 1-a1) and the *Object Projection* (Figure 1-a2).

Image-level Processing: Inspired by Zeiler and Fergus [52], as well as Selvaraju et al. [45], we adopted a simple but effective method to visualize the discriminative features extracted from the network. We used the final convolutional feature maps, which contain rich spatial and semantic information [45], [52], before the output layer. We operated average pooling on feature maps along the channel dimension to generate a global discriminative density map.

The input of object detection model is an RGB image of size $h \times w$. The model processes the image with Conv Layers and Output Layer, and then outputs the results consisting of various types of information about detected objects, such as bounding boxes, locations, dimensions, and angles. We used such information to visualize the 3D bounding boxes of objects in the input image. We further extracted the final feature maps of the Conv Layers, which are in the shape of $h' \times w' \times c'$, where c' denotes the number of convolution filters. Then, the feature maps were aggregated by average pooling to obtain a 2D $h' \times w'$ aggregated feature map in the Feature Aggregation Layer. Next, the aggregated feature map was restored to the shape of the original image but with only one channel $h \times w$ using up-sampling with bi-linear interpolation in the Up-sampling Layer. Finally, the image-size feature map was normalized and mapped to a selected color space, through which we obtained the *Density Map* (Figure 1-a1).

Object-level Processing: For object feature projection, we made a region selection from the image-size feature map with the size of 2D ground-truth boxes as seen in VATLD [20]. We scaled them into the same $h_o \times w_o$ dimensions named object-level feature map in the Object Extraction Layer, and reshaped them into one-dimension vectors for t-SNE projection in *Object Projection* (Figure 1-a2).

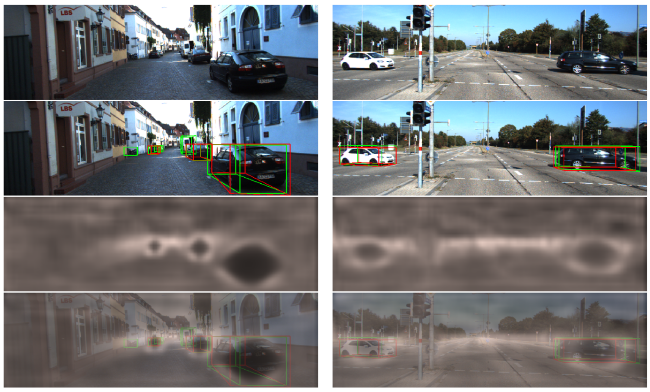


Fig. 4. Preliminary visualization of object detection results and features. Green bounding boxes indicate ground truth and red bounding boxes are predicted positions. The darker the color, the lower the activation value. Left: lose two objects in the detection; Right: detect both objects.

5.1.2 Features View

The Features View (Figure 1-a) visualizes model data to provide an insight into "*How*".

Density Map. The *Density Map* in Figure 1-a1 is obtained by the processing stages we discussed above. It provides an intuitive insight into what is captured by the model. The RGB image in Figure 1-a1 tells usenvironmental information (R1.2).

The bounding boxes for autonomous driving generally use green to indicate labeled results and red to indicate predicted results. As shown in Figure 4, the left image has five objects with green bounding boxes, but only three are detected with red bounding boxes consistent with three parts with high edge activation values and low internal activation values in the *Density Map*. In the right image, two objects are both detected consistent with two parts of activation regions. The global discriminative density map highlights the regions rich in semantic information. Through this visualization method, developers can identify those cases of failure in which the semantic information of objects is not well captured. To avoid distracting users and make the color design of the system more coherent, we choose gray-scale instead of "blue to red" in Grad-cam [45] as the color space.

This view can be replayed over time so that model developers can review how the feature maps, detection results, and surroundings change over time (R2.1). By comparing the density map with the temporal information and bounding boxes of detection results, developers can get an initial answer to the question of *How* the model detects objects (R1.3).

Object Projection. The processed features and detection results are combined in Figure 1-a2. It visualizes the processed features of each detection and corresponds to *IoU* on a two-dimensional coordinate system, providing an overview to observe the relative distance and distribution of the object-level features, and further answers the question of *How* the model fails (R1.3).

Green is generally used to indicate good performance and red to indicate poor performance in autonomous driving. To fit the habits of domain experts, we encode *IoU* with the color, mapping "0 to 1" to "red to green". There are also gray points, which represent ignored objects. The model performs extremely poorly on these objects as they are either heavily occluded, truncated, or too far away from the camera. Thus, the final assessment metrics exclude these objects. These colors are consistent with all other views (Figure 1-e).

This chart supports zooming in or out and brushing a polygon area of projection points. The selected objects will be highlighted in this chart and the update of other views will be triggered accordingly (R2.3). We chose other easily distinguishable colors referring to ColorBrewer [23] to encode interaction-related results. As shown in Figure 1-e, the objects highlighted with orange, blue, and yellow strokes are linked to the selected object, current time and intersection of them in all other views (R3.1).

5.2 *Macro*-interpreting Module: Spatial-temporal Visualization

The *macro*-interpreting module mainly focuses on model analysis from temporal and spatial perspectives. Each perspective is supported by a view. *Temporal View* (Figure 1-b) includes *Temporal Distribution* (Figure 1-b1, b2, b3). *Spatial View* (Figure 1-c) includes *Scene* (Figure 1-c1) and *Trajectory* (Figure 1-c2).

5.2.1 Temporal View

The Temporal View (Figure 1-b) visualizes the temporal data and detection results from model data including the states of an autonomous driving vehicle (Figure 1-b2) and object-level features information with detection results (Figure 1-b3) in temporal distribution (Figure 1-b1). Such information helps to provide an insight into "*When*" and "*How*".

Figure 1-b1 is a timeline that provides the horizontal axis for the following charts. It can be selected, dragged and played, and the detected objects in current time will be visualized in blue in all the views (R2.1).

Autonomous Driving Vehiche States. Figure 1-b2 shows the velocity, acceleration, and wheel steering data from the GPS monitoring of the vehicle, which provides a way to observe the states of the autonomous driving vehicle (R1.1). We integrated the line chart and two dashboards in AVS (the Autonomous Visualization System) [2] into Figure 1-b2 to share the timeline in *Temporal View* and better align the analysis between the vehicle states and the model detection results in the following chart. The

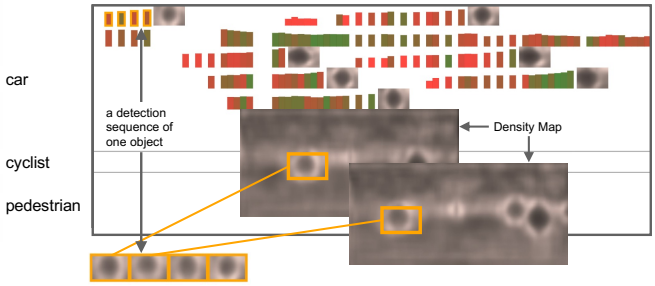


Fig. 5. Illustration of Figure 1-b3. A detection sequence of one object is put together, followed by a density map of the last moment. The object-level density map at the bottom is extracted from *density map* in the same way as *object projection*.

vertical line shows the values of line charts at the current time. The horizontal axis can be brushed. The corresponding data of the brushed time period will be highlighted in all views (R2.1).

Object-level Density Maps. Figure 1-b3 visualizes the detection sequences of objects from the tracklets, as well as the detection results from the model. This chart shows the correlation between object-level features and detection results in temporal distribution and further combines the analysis tasks of *When* and *How* (R1.1, R1.3).

As shown in Figure 5, the horizontal axis represents the timestamps of the whole sequence, and the vertical axis represents different objects. There are two design alternatives: 1) each object occupies one row, 2) merging objects shown in different periods into the same row to reduce the waste of space. In most cases detected objects only appear in a specific period, so such merging will not add too much misleading id information of objects and can save space. An object in a timestamp when the camera took the image is visualized as a combination of a rectangle and a snapshot of a density map. The object-level density maps are dimensionality reduced to one dimension by PCA and encoded to the height of the rectangles in each row. A pattern between features and detection results could be detected that the higher the height, the better the detection results tend to be in Figure 1-b3. Showing all snapshot sequences is actually not very informative and distractive. To reduce the density maps, the last density map of each object is displayed after the last rectangle. The objects are presented in classes with different boxes.

Each density map has a "Scale Up" function for easy viewing when hovering as shown in Figure 1-b3. Developers can click the detection sequence of one object or the final density map of interest to select an object (R2.3), which would also display the density maps of the whole sequence of this object at the bottom.

5.2.2 Spatial View

The Spatial View (Figure 1-c) visualizes the spatial data including Scene (Figure 1-c1) and Trajectory (Figure 1-c2) to provide insights into "*Where*".

Scene. Figure 1-c1 is a 3D spatial stereogram drawn with raw data that contains the point cloud data from the vehicle-mounted LiDAR, latitude and longitude information from GPS, and tracklets of labeled objects (e.g., relative positions, dimensions, and classes). It is a key view for spatial-temporal scene perception, which displays sufficient information of *Where* (R1.2).

We reused the 3D view in AVS [2] with customized changes in our design. In AVS [2], to show the obstacles around the objects, the LiDAR point cloud is visualized by the point set. To show the specific road and location environment, the latitude and longitude information from GPS is used to draw the map. With the 3D bounding boxes of the labeled objects drawn in the spatial scenes, developers can fully perceive the spatial information between the objects and their surroundings. We changed the colors of different classes of 3D bounding boxes in the current time to different shade levels of blue to distinguish them from other colors we used in other views. Bounding boxes for objects other than cars, pedestrians, or cyclists are removed, because they are not a concern here.

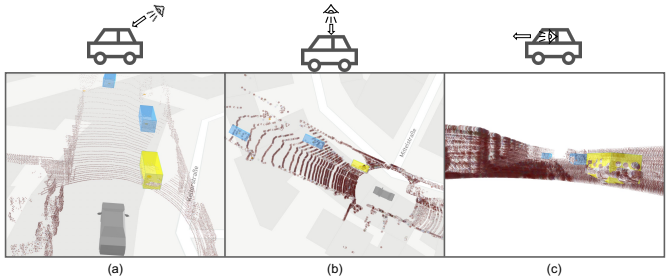


Fig. 6. Three view modes of *Scene*. (a) Perspective view at the car; (b) Orthographic view; (c) Perspective view from the car.

Developers can select a perspective through the control in the upper left corner, and three view modes are provided as shown in Figure 6. The scene can change over time, and this chart is linked to other charts through time and selected object.

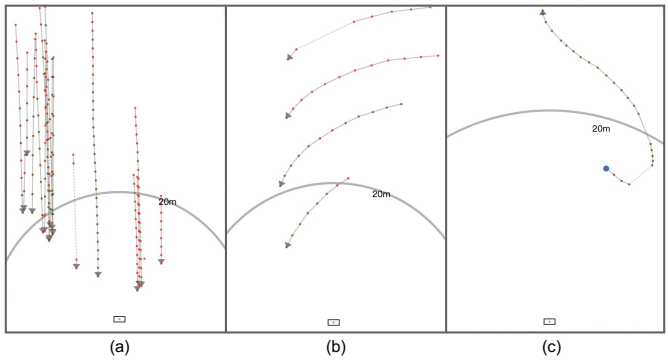


Fig. 7. Visualization of trajectory patterns. (a) facing the autonomous vehicle approaching; (b) facing the autonomous vehicle turning; (c) turning in front of the autonomous vehicle and driving away.

Trajectory. *Trajectory* (Figure 1-c2) combines the relative position in the bird's eye view from tracklets and detection results, and arranges them in the order of detection. Developers can combine such information in answering the questions of *When* and *Where* (R1.1, R1.2).

The basic trajectory visualization method such as trajectory clustering by Lee et al. [31] usually visualizes the sequences that connect the absolute positions of the object. We combined the detection results and detection field with the object trajectory to have a global sense of the distribution of detection results in space. Considering that a global view in *Scene* is provided, the ego-centric view centered on the autonomous vehicle is also important, so we adopted such an ego-centric visualization form and relative positions. The rectangle at the bottom represents the camera of the autonomous driving vehicle, which is fixed as a reference. The arc in the view marks the position of 20 meters away from the camera,

as a reference line for the relative distance perception. Each point indicates one object in one detection, and each line represents the trajectory of an object, and the direction of each arrow is the movement direction of the last two detections' positions of one object, which always combines trajectory lines to indicate the whole segment overall direction of relative movement. The dashed line indicates discontinuous detection time, *i.e.,* the object has disappeared between the beginning and end detection of the dashed line. As shown in Figure 7, three different trajectory patterns could be visualized.

Developers can zoom in and brush later (R1.2). The brushed objects and corresponding paths will be reserved and redrawn. The update of other views will be triggered accordingly as well (R1.2, R2.1).

5.3 Interactive Visualization to Interpret

We designed the following interaction methods to link all views to explore.

Filter/Select. Figure 1-d is composed of two controls. Figure 1-d1 supports developers to select the data by checking what categories (e.g., car, cyclist, or pedestrian) detected objects belong to, and what detection results (e.g., tp–true positive, fn–false negative, or ignored) may look like. The types of objects and the percentage of detection results are shown below. When developers explore the other views, the statistics of the filtered data are also calculated in real-time as shown in Figure 1-b2, which can be compared with the overall statistics. Clicking the restart button in the upper right corner can restore the data to the initial state for a new round of exploration. Clicking the play button will update the data with orange and blue consistent with the *Scene* and timeline, but developers need to re-click "pause to brush" to stabilize the brush.

The control in Figure 1-d2 takes the form of parallel coordinates to show the high-dimensional labeled data, which is plotted with *IoU* and locations of labeled objects, including detailed relative orientation (alpha and rotation-y) and position (x, y, z in the camera coordinate system: rightward, upward, and forward distances). Ignored objects are not drawn on the first axis, *i.e., IoU*, since they are not included in the AP calculation. Each axis can be brushed, and the intersection will be taken to filter the data (R1, R2.3).

Explore Distribution. The control panel in Figure 1-d is used as an overview. After filtering low *IoU*, tp or fn, developers are able to observe the selected objects in other views. In addition, they can choose to observe the angular information and relative position information through other axes. With the *Temporal Distribution*, developers can examine the corresponding time position on the timeline. More intuitive information about spatial-temporal scenes and the states of the autonomous driving vehicle of the detection can be obtained in the *Scene* and the *Density Map*. This interaction is used to explore the problem of *When, Where* and *How* the objects are poorly detected (R1).

Discover Correlation. One of the three views is used as an overview to further explore the other two and the control panel. For example, after observing the *Temporal Distribution* and the corresponding detection results, developers can brush a time range of interest. Then, they can investigate the object-level density maps in the *Temporal View*, the *Object Projection*, the *Trajectory* and the parallel coordinate diagram with *IoU* and relative location information. They can drag the timeline to the corresponding position to play, perceive the actual spatial-temporal scenes through the *Scene* and images in the *Density Map*, and explore the relationship between the features and detection results through Figure 1-a1. Through this interactive exploration, for objects detected over time that developers are interested in, they can observe the spatial-temporal environment, the features, the detection results, and the relationships of these results to spatial locations and features (R2.1).

Iterative Brush. Two of the three views are used as an overview together to explore the other one and the control panel. For example, combining the spatial-temporal information and detection results from the *Temporal View*, parallel coordinate diagram and *Spatial View*, developers can select the objects by the iterative brush of their interest and playing the timeline. Then, they are capable of observing the selected objects in other views. This interaction helps to explore how objects in some spatial-temporal regions are detected, their feature distribution, detection results, and the relationship between results and features (R3.1).

6 Implementation

We used an open-source system, AVS, as the underlying framework for system development. AVS [2], is a web-based 3D autonomous driving data visualization tool. It contains two parts, XVIZ and StreetscapeGL. XVIZ provides a stream-oriented view of the scene over time, and a declarative user interface system that constructs XVIZ streams based on the data to be transmitted to StreetscapeGL. StreetscapeGL is a web toolkit for building XVIZ protocol data, which provides components to visualize XVIZ streams in 3D views and various charts.

With AVS, we built the playable point cloud maps and line charts of the states of an autonomous driving vehicle. We added to StreetscapeGL various designs, including the features view for visualizing *micro*-level features including density map and object projection, and the temporal distribution and trajectory for visualizing *macro*-level spatial-temporal scenes. We also modified the scene and line charts to optimize the coordination among different views, and added controls to enable the interactions between our own implemented views and the original stream data visualization views of AVS.

7 Case Study

To demonstrate how our system can help developers understand and diagnose models, we present two case studies with two different autonomous driving scenarios. We first introduce how we processed and prepared data for case studies, and then describe each case study individually.

7.1 Data Processing

We chose the KITTI [18] Dataset, a popular and well-known dataset in autonomous driving. There are many sequences that last about 10 - 120 seconds for investigation and benchmarking. Our system can load different pieces of data for investigation and we only show the selected cases for demonstration. After extracting the sequence data, we first trained a 3D object detection model D4LCN by Ding et al. [14] using one-half of the images in our dataset. We then selected two segments of autonomous sequence data for case studies. The images in these two segments have labels but were not used for training. By putting them into the previously trained model, we recorded the model outputs on object detection. We collected feature maps from the layer before output layer and

the final results, including bounding boxes, classes, occlusion and truncation information, locations, dimensions, angles and scores. We used the output results of the model and ground truth to calculate the *IoU* as the measuring metric of the detection results, pre-processed the feature maps, and ported them into our system together with the tracklets, LiDAR point clouds, GPS, and the images of the sequence data.

7.2 Case 1: Autonomous Driving on a High Way with the Presence of Other Objects

This case study concerns the movement of an autonomous driving vehicle on a highway. At first, the vehicle makes a right turn, and then continues on the highway where other cars, cyclists and pedestrians are presented. The whole driving process lasts about 32 seconds, with 314 timestamps. The model identified 233 timestamps of images and detected 1314 objects. Here, we follow the three interaction methods described in Section 5.3 to explore the sequence data of this case and analyze the model performance.

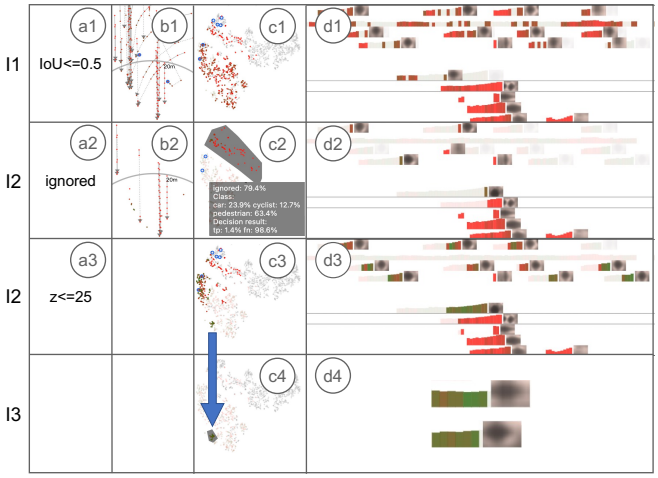


Fig. 8. Interaction methods in Case 1. I1: Exploring Distribution; I2: Discovering Correlation; I3: Iterative Brush. Bad detection results in c2 are mostly related to cyclists and pedestrians, as shown in d2, which indicates the effectiveness of aggregation in *Object Projection*. Most cyclists and pedestrians are close to the vehicle, as showed in a3 and d3. The effectiveness of aggregation in *Object Projection* is also visualized in c4, with only two objects in d4.

Exploring Distribution. As shown in Figure 8-a1, b1, c1, d1, we can only find that the cases of failure happened at both distant and adjacent locations to the camera. The time and feature distribution of failure cases are scattered (R1) with the comparison of Figure 1, which is not informative enough. This indicates that further interactive exploration is needed to obtain insights into *When, Where* and *How* the cases failed.

Discovering Correlation. Figure 8-c1 and Figure 1-a2 show that most of the points clustered around the ignored points are poorly detected. We are interested in such points, so explore their spatial-temporal distribution by selecting and brushing in Figure 8-a2, c2 (R2.3). Figure 8-c2, d2 shows that more of these points are in the classes of cyclist and pedestrian (cyclist: 12.7%, pedestrian: 63.4%, car: 23.9%) compared with the overall (cyclist: 6.3%, pedestrian: 7.7%, car: 86.0%) in Figure 8-d1, and they are in proximity to the camera as indicated by Figure 8-b2. The model did not achieve high accuracy in detecting these two classes of objects even when they were at close distances. One explanation for this failure is that these two classes make up a small percentage

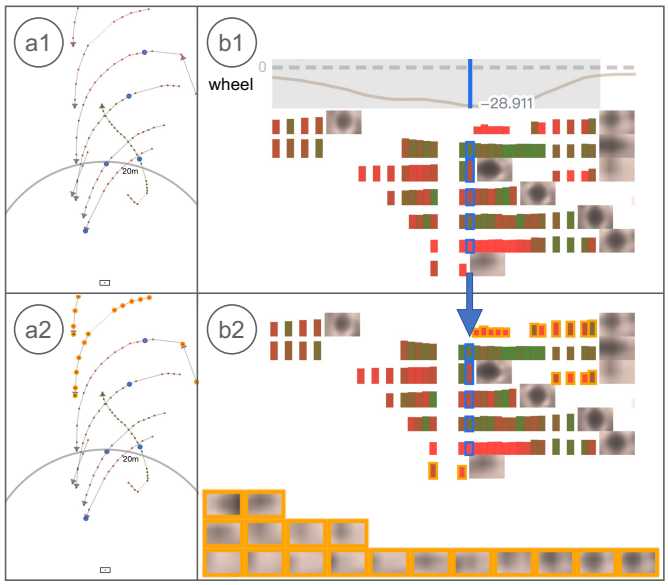


Fig. 9. Discovering Correlation example. Objects with bad detection results indicated by the end density map during the turning process in b1 are mostly with obscure activation boundaries in b2 and farther than others in a2.

of the training dataset, compared to the class of car (cyclist: 8.8%, pedestrian: 2.9%, car: 57.2%). This small percentage also explains why the relationship between the relative position and the *IoU* (z-axis in Figure 1-d2) is inconsistent with the intuitive belief that closer distance means more accurate detection.

Brushing the objects closer to the camera in Figure 8-a3, we observe that their distribution is more concentrated in Figure 8-c3 and this is consistent with the previous analysis that cyclists and pedestrians mostly appear closer to the camera in Figure 8-d3 (R2.2). We are interested in a small set of points that are closer together in Figure 8-c3, which will be explored in subsequent interactions. We are interested in the turning detection results, so we brush the consistent period, and observe the trajectory and detection sequences of objects in Figure 9-a1, b1 (R2.1). As shown in Figure 9-a2, the sequences with bad detection results indicated by the last density map (which mostly have obscure activation boundaries in Figure 9-b2) are far away from the camera (R2.1).

Iterative Brush. For the cluster and the object-level density maps we just mentioned, we continue to explore and find that this cluster represents only two objects, as shown in Figure 8-d4, reflecting the fact that the feature distribution in Figure 8-c3 knows not only the aggregation of the detection results (Figure 8-c2), but alsoaggregations of the same object (R3.3).

We try to find some patterns through our interaction methods. Firstly, we select a period with more objects appeared in Figure 10-e (①). In Figure 10-f, we observe two end density maps that are clearly inconsistent with others, so we click them to get their distributions in other views (②). We move the timeline in Figure 10-d to the corresponding position (③) and find the actual location of the objects in Figure 10-b (⑤) by looking at the selected position in Figure 10-a (④). Combining Figure 10-c (⑥) with the dynamic play function of the views, we find that the sharper activation boundaries are consistent with better detection results. The fact that the selected car is parked provides macro interpretability to better detection results. The sequence with uneven activation boundaries is associated with bad detection

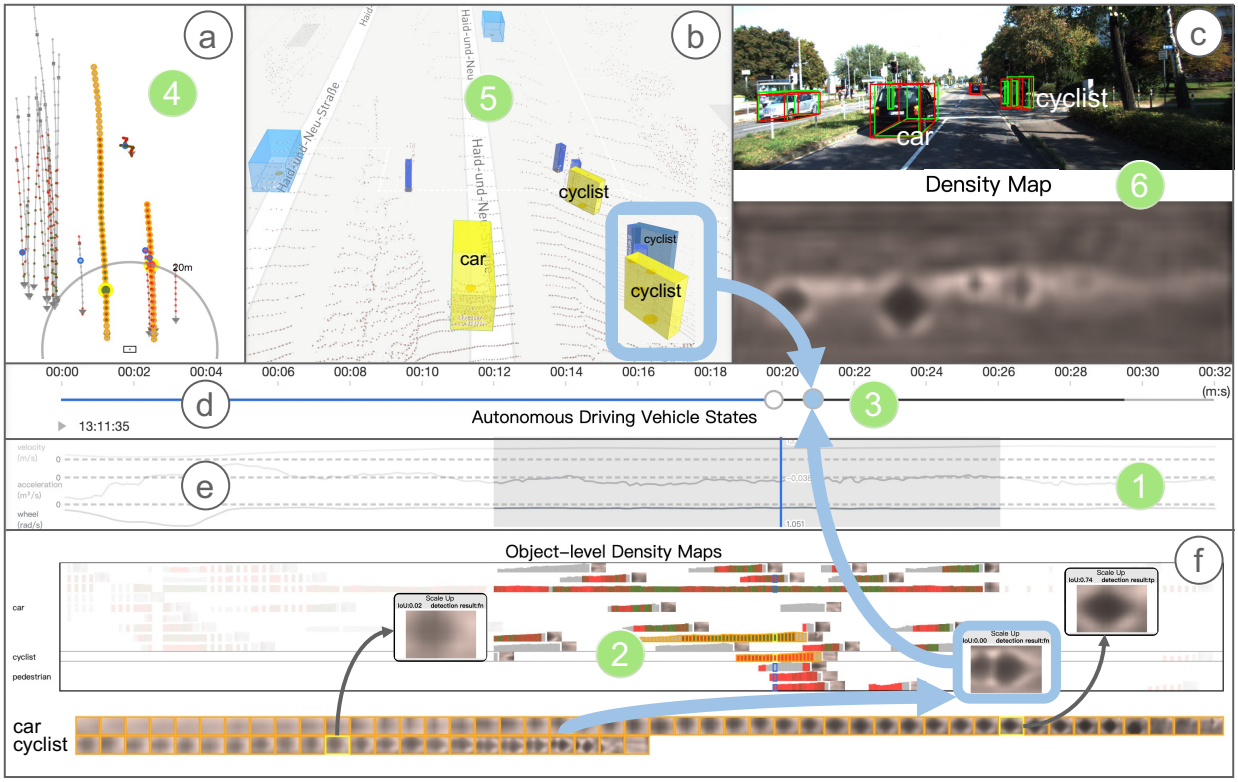


Fig. 10. The whole exploration process from the case. The numbers of ①-⑥ indicate interaction step sequence. (f) The first sequence with sharper activation boundaries and good detection results corresponds to the parked car in (a, b, c); and the second sequence with uneven activation boundaries and bad detection results corresponds to a cyclist in (a, b, c) close to the pedestrians and the other cyclist.

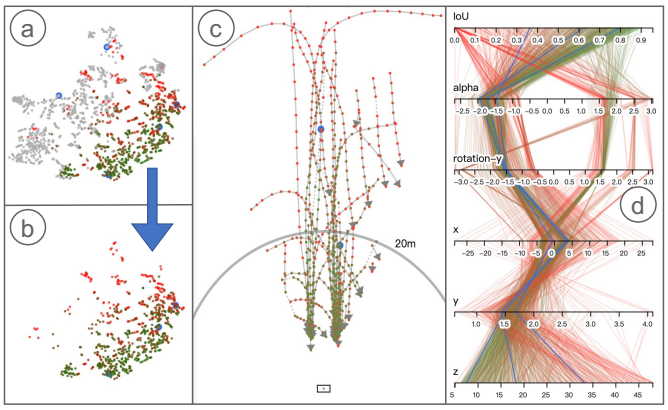


Fig. 11. Overview distribution of Case 2. (a, b) The objects closer to the ignored are mostly detected with poorer results. (c, d) The closer the distance, the better the detection.

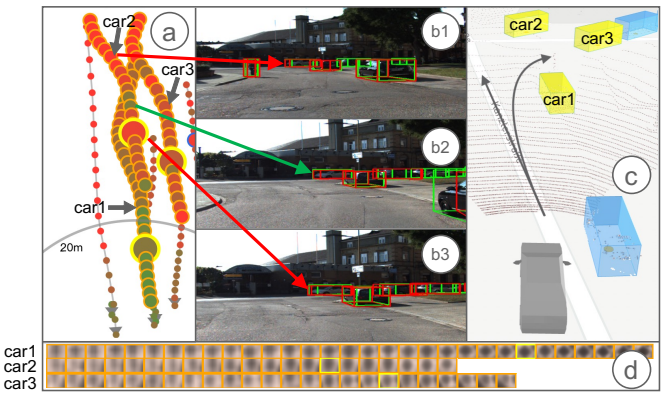


Fig. 12. Occlusion pattern. (b1) Car2 is not badly occluded; (b2) The detection results of car2 gradually become better as the autonomous driving vehicle approaches; (b3) As car2 approaches car1, the detection model might more focus on car1.

results and proximity to the pedestrians and the other cyclist, which may provide micro- and macro-level interpretability to the cases of failure. Thus, using interactive visualization, we are able to explain the failed predictions from micro and macro perspectives. (R3.2)

7.3 Case 2: Autonomous Driving with the Presence of Other Objects Parked on Both Sides of Road

This case study is built on a scenario in which an autonomous driving vehicle is driving when there are cars, cyclists and pedestrians parking around. We try to find some detection patterns in this case and summarize them as occlusion, turning and deviation. The parking process lasts about 45 seconds, and includes 435 timestamps. In total, 194 timestamps of images and 1596 objects are detected.

From the projection shown in Figure 11, there are two distinct clusters in Figure 11-a: one mostly consists of objects with poor detection, including ignored objects and objects close to them, and the other has objects with better detection results. The trajectory of objects (Figure 11-c) shows that there are several turnings in this sequence. There is an obvious trend in Figure 11-c and z-axis in Figure 11-d that the closer the distance of an object is to the vehicle, the better the detection result is. Most of the lines are distributed in the x-axis [-10,20], indicating that objects appear on both sides of the autonomous vehicle.

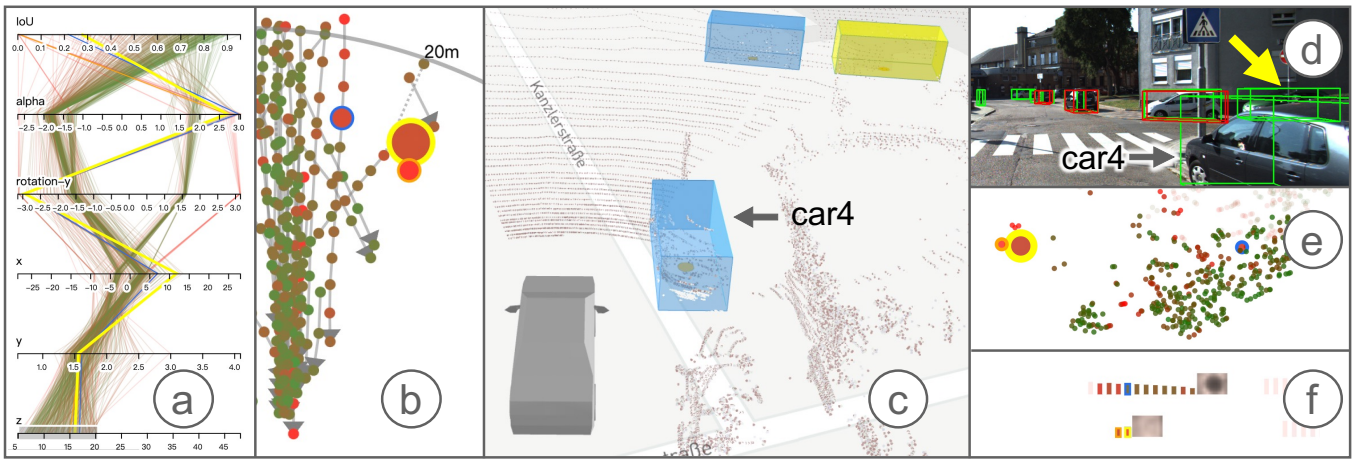


Fig. 13. Deviation pattern. (b, c, d, f) The car is close but still poorly detected due to the relative position deflection and occluded by car4; (a) The angle-dependent alpha and rotation-y of the deflection angle information differ from the other objects; (e) The feature distribution is away from good detection concentration.

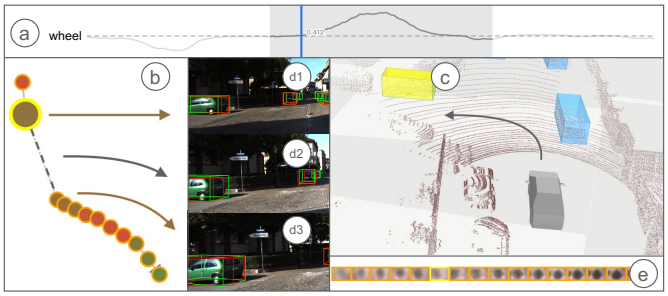


Fig. 14. Turning pattern. (d1) Detection before the dashed line; (d2) The detection on the dashed line. Although the bounding box is drawn, the object is ignored in the AP calculation because it is truncated to a greater extent; (d3) Detection after the dashed line.

Occlusion Pattern. We select a period at the end of the sequence in the *Autonomous Driving Vehicle States*. Figure 12-a shows one interesting phenomenon: there are two trajectories that do not conform to the pattern that the closer the distance is, the better the detection is. In the region where they intersect, two trajectories have a process of getting better and then worse. We interpret this observation as follows. After dragging the timeline and picking the car at the appropriate position in Figure 12-c, the correspondence of the car in Figure 12-c and Figure 12-a shows that car2 and car3 may be obscured by car1. We play the timeline and combine the real scenes and object-level density maps to understand the detection results' change of car2. We find that car2 is not badly occluded when the autonomous vehicle is far away in Figure 12-b1, and the detection results of car2 gradually become better in Figure 12-b2 as the autonomous driving vehicle approaches. However, as car2 approaches car1, it is obscured by car1 and the detection results become worse again in Figure 12-b3 (although the difference between Figure 12-b2 and Figure 12-b3 is not so obvious, density maps of car2 in Figure 12-d could help to interpret the trend of getting better or worse. We think that in Figure 12-b3, the detection model would focus more on car1 because it is closer compared to Figure 12-b2). Then as the distance is drawn closer, the detection results become gradually better again. (R3.1)

Turning Pattern. In Figure 14-a, we see a large change in the wheel angle, select the corresponding period, and observe a trajectory in Figure 14-b where the detection result first becomes better, then worse and then better. The dashed line indicates that the car appears and disappears. Dragging the timeline and selecting the car at the appropriate location in Figure 14-c, we learn that the selected car disappears from the detector's field of view due to truncation when the autonomous driving vehicle turns (Figure 14-d2). Moreover, the detection results are poor at large angle change of relative position with analysis of the real scene in Figure 14-d1, d2, d3. The density maps of the car in Figure 14-e could assist in interpreting the detection results. (R3.1)

Deviation Pattern. In Figure 13-a, by brushing objects within 20m of the forward distance from the camera, we spot some lines with lower *IoU*. Observing that there is an object in Figure 13-f with lower *IoU*, we select it and drag the timeline, and find that in Figure 13-b, the orange object appears in the right front position. Thus, we select the object at the corresponding position in Figure 13-c. With the adjustment of the timeline combined with the real scene in Figure 13-d, we find that the car is close but still poorly detected due to the relative position deflection and occluded by car4. The angle-dependent alpha and rotation-y of the deflection angle information in Figure 13-a also differ from those of other objects, which could help to explain the deviation. The feature distribution in Figure 13-e can verify the pattern of good detection concentration found in the overview distribution. (R3.3)

8 Expert Study

We conducted an expert study to evaluate the usability and effectiveness of our system.

8.1 Study Process

We recruited seven experts, P1 - P7, to conduct an interview study to evaluate the usability and effectiveness of our system. Four of them, P1, P2, P3, and P4, were involved in the study of validating design requirements mentioned in Section 3. Three new experts, P5, P6 and P7, have been in the area of computer vision for autonomous driving for more than 5 years, and had no knowledge about our research before being recruited. We interviewed each of them separately.

Each interview session includes a 20-minute introduction, a 7-minute video presentation, 30-minute exploration using the think-aloud method by Van et al. [49], and a semi-structured interview lasting between 20 to 30 minutes. The video presentation introduces the user interface and interaction methods of our system, and demonstrates our exploration processes to discover the patterns similar to those mentioned in Section 7. In the exploration phase, we instructed all experts to watch the interaction methods and Case 1 in the video and asked them about their understanding of the video. Two experts, P3 and P5, volunteered to use our system to reproduce the analysis result of Case 1 and freely explore the system.

In interviews, we asked all the experts questions in four categories: their understanding of all our views and the three types of interaction methods, their thoughts about our analytical methodology, their opinions on the findings of case studies and exploration results, and their judgment on the values of the system to model improvement.

8.2 Results from the Exploration Phase

In the exploration session, the two experts, P3 and P5, correctly performed three interaction methods (Section 5.3. They successfully found the patterns in Case 1, and their successes imply their full understanding of our system. During the exploration process, P5 discovered how the detection of a cyclist around might be affected by light in Case 1. P3 found a failed detected car that was obscured by a vehicle when the autonomous driving vehicle was turning, a phenomenon that fits the findings of Case 2. They also noticed several car movement trajectories that lead to better and worse detection results and identified some occlusion patterns.

We also recorded those views used by two experts in their exploration processes. Our data showed that all views in our system had been used. The experts made full use of *Object-level Density Maps* and *Scene*, and said that the image in the upper right corner aided their perception of the real scene. P5 frequently used the *Statistics* in Figure 1-d1 to filter and parallel coordinate plot on the left to see the distribution, and indicated that acceleration and velocity were possibly used for other situations, such as when the autonomous driving vehicle is in special road conditions or emergency braking.

8.3 Results from the Semi-structured Interviews.

Data from interviews showed that all experts confirmed that they understood the views and interaction tools of our system.

- **Analytical Methodology.** We asked if our interface design and interaction are easy to use and understand, and they were all positive about our design. P1, P2, P3 and P4 all felt that our visualization design fully met their needs. P5, P6 and P7 confirmed that these visualizations were exactly what they wanted to know and the analysis workflows could improve their daily work efficiency. We also asked experts about what other information they would like to know. P2 suggested that we add some attribute information of objects, such as color. P4 and P5 suggested more visualization designs on model features, such as multi-scale layers. Adding light and weather conditions was also mentioned (P2, P5, P6, P7).
- **Cases.** We interviewed the experts about their understanding of the cases we showed them and whether we should consider other scenarios of interest. They confirmed that they understood the findings to the *When, Where* and *How* questions demonstrated in our cases, and believed they are useful for model understanding. P5 indicated that occlusion and light were the main causes in terms of results, consistent with his past experience. He concluded that the algorithm has poor detection results for small objects. P4 would like to explore more patterns, such as those of acceleration, deceleration and path planning.
- **Insight and Inspiration.** We asked whether those patterns that we explored could help the understanding of failure detection in these scenarios and whether the insights could help to improve the model. They all agreed that these patterns deepened their understanding and helped to improve the model in the future. P2 believed that *"These explored patterns not only make the original understanding verified, but they also allow us to quickly find some problems and summarize some classifications, which can improve efficiency."* P5 said *"The understanding of the spatial-temporal scenario is sufficient and I expect the system to further integrate more information within the model."* P6 said *"This is a powerful tool to verify the results of the model."* For improving and enhancing the model, P7 said *"It is very useful for extracting special scene data, and increasing its training ratio can improve the performance of the model in special scenes."* P2 indicated that *"It is helpful, but the improvement of the model itself is more important."* P4 stated that *"Previously we had only trained and tested the whole dataset to get the accuracy, or visualized the bounding boxes, without having an overall view of the results"*, and he believed that *"The system helps to analyze the scenarios for failed samples, to augment data, and to improve the model performance"*. P3 believed that our method is very useful for model improvement, and said that *"Analyzing the failure detection cases of small objects at long distances can help us design methods such as data augmentation, multi-scale feature fusion and information interaction between multiple objectives. By analyzing the feature maps of obscured objects, we can design loss functions, optimize the design idea of anchor and change the matching strategy."*

Overall, all seven experts fully understood our system and were positive about its usability, effectiveness, and implications for future model enhancement.

9 Discussion and Conclusion

Comparison and Generalization. Previous interpretable methods largely focused on feature representations in models, or micro-level information as discussed in our research. Our approach explains model failure from a new perspective—by combining macro-level temporal and spatial context information with micro-level feature interpretations. Model developers can use our approach to improve the efficiency of error analysis, and discover spatial-temporal patterns in specific scenarios when the model fails. These findings could be used to perform data augmentation to improve algorithm performance.

Our approach can be extended to other interpretable work with algorithmic errors by considering contextual factors. For analysis involving spatial-temporal data, our method can be directly applied. For analysis involving other factors, such as interactions among models in the same system, our method suggests the consideration of the integration of the visualization of macro-level relationships among these models, such as hierarchy view or network view, with traditional views on model features. Model developers can use our approach to explain model errors and improve model effects from a spatial-temporal perspective by combining information about the features inside the model.

Limitations and Future Work. There are three straightforward directions of improvements of our work: First, although we provided comprehensive interpretability of spatial-temporal scenarios, our interpretability at the *micro* level can go deeper. Our system only visualizes the outputs of the network layer in front of the output layer by up-sampling, and may not provide sufficient insight into the *How* question. More work is needed to meet the needs at the level of feature visualization, such as multi-scale network features as suggested by P4 and P5 in Section 8. Second, additional requests from domain experts for additional information visualization in Section 8 can be considered in the future, e.g., adding macro information such as light, weather and object color. Third, the detection results are not integrated into Figure 1-c1, and the inclusion of real-time detection results in the visualization scene could be considered in the future to make the system more intuitive.

Then, our current system largely focuses on the analysis of failure scenarios with a time length at the level of minutes, and we have not tested the scalability of the system with longer cases. This is largely due to the fact that the available public data sets only provide scenarios with such time lengths. To analyze longer scenarios, our system needs to consider the challenge in computational powers demanded by such scenarios. While data can be processed offline, real-time interaction with a large data volume could be a bottleneck. To improve the scalability of our system, measures like better system architecture to accommodate longer scenarios or effective data aggregation methods across different scales can be considered.

Furthermore, to fully understand the effectiveness of our system on the work of model developers, more user studies are needed. Considering the complexity of the task in the analysis of model failures, we believe that a longitudinal study involving the deployment of our system to model developers and the collection of behavior data through their real actions will provide reliable evidence of the system's effectiveness.

Lessons Learned. Autonomous driving technologies have been growing rapidly recently, and visualization in the field has just started. There is a great demand for visualization in autonomous driving. Therefore, visualization is very important in this field of research. The exploration of interpretability is challenging. Our approach validates that combining spatial-temporal relationships with information from the neural network can provide new insights for model developers.

Due to the complexity of the knowledge in the field of autonomous driving, it is challenging to fully understand the breadth and depth of the problems model developers encounter. To support their work, visual analytics researchers need to gain in-depth knowledge of this field and the nature of the work of model developers. Doing so will allow visual analytics researchers to approach the problems from the perspective of users, and then build better systems.

Conclusion. In this research, we propose a visual analytics approach and develop a system to combine *micro*-level model features and *macro*-level contextual factors, spatial-temporal information, to provide in-depth visual analysis of object detection models in autonomous driving. Our case studies demonstrate the rich functions of the system in analyzing different autonomous driving scenarios. Our interviews with domain experts show that our approach can improve the interpretability of object detection models and help model improvement through the analysis of model failures. We expect our research will offer some new ideas for the interpretability of object detection in autonomous driving by combining information from multiple perspectives and supporting rich exploration-oriented user interactions with models.

ACKNOWLEDGMENTS

The authors want to thank reviewers for their suggestions. This work is supported by Shanghai Municipal Science and Technology Major Project (No. 2018SHZDZX01, 2021SHZDZX0103), General Program (No. 21ZR1403300), Sailing Program (No.21YF1402900) and ZJLab.

REFERENCES

[1] Apollo. https://apollo.auto/index_cn.html.
[2] AVS. https://avs.auto/#/.
[3] Worldview. https://webviz.io/worldview/#/.
[4] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.
[5] A. Achberger, R. Cutura, O. Türksoy, and M. Sedlmair. Caarvida: Visual analytics for test drive videos. In *International Conference on Advanced Visual Interfaces*, pages 1–9, 2020.
[6] G. Andrienko, N. Andrienko, G. Fuchs, and J. Wood. Revealing patterns and trends of mass mobility through spatial and temporal abstraction of origin-destination movement data. *IEEE Transactions on Visualization and Computer Graphics*, 23(9):2120–2136, 2016.
[7] N. Andrienko, G. Andrienko, and P. Gatalsky. Exploratory spatio-temporal visualization: An analytical review. *Journal of Visual Languages & Computing*, 14(6):503–541, 2003.
[8] M. Bojarski, A. Choromanska, K. Choromanski, B. Firner, L. Jackel, U. Muller, and K. Zieba. VisualBackProp: Visualizing cnns for autonomous driving. *arXiv preprint arXiv:1611.05418*, 2, 2016.
[9] K. Buchin, B. Speckmann, and K. Verbeek. Flow map layout via spiral trees. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2536–2544, 2011.
[10] T. Butkiewicz, W. Dou, Z. Wartell, W. Ribarsky, and R. Chang. Multi-focused geospatial analysis using probes. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1165–1172, 2008.
[11] R. Chang, G. Wessel, R. Kosara, E. Sauda, and W. Ribarsky. Legible cities: Focus-dependent multi-resolution visualization of urban relationships. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1169–1175, 2007.
[12] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *ACM SIGMOD International Conference on Management of Data*, pages 491–502, 2005.
[13] S. Chen, Z. Wang, J. Liang, and X. Yuan. Uncertainty-aware visual analytics for exploring human behaviors from heterogeneous spatial temporal data. *Journal of Visual Languages & Computing*, 48:187–198, 2018.
[14] M. Ding, Y. Huo, H. Yi, Z. Wang, J. Shi, Z. Lu, and P. Luo. Learning depth-guided convolutions for monocular 3D object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 11672–11681, 2020.
[15] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes challenge 2012 (VOC2012) results. In *URL http://www. pascal-network. org/challenges/VOC/voc2011/workshop/index. html*, 2011.
[16] M. Everingham, A. Zisserman, C. K. Williams, L. Van Gool, M. Allan, C. M. Bishop, O. Chapelle, N. Dalal, T. Deselaers, G. Dorkó, et al. The 2005 PASCAL visual object classes challenge. In *Machine Learning Challenges Workshop*, pages 117–176. Springer, 2005.
[17] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva. Visual exploration of big spatio-temporal urban data: A study of New York city taxi trips. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2149–2158, 2013.
[18] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.
[19] R. Girshick. Fast r-cnn. In *IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[20] L. Gou, L. Zou, N. Li, M. Hofmann, A. K. Shekar, A. Wendt, and L. Ren. VATLD: A visual analytics system to assess, understand and improve traffic light detection. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2020.

[21] D. Gunning, M. Stefik, J. Choi, T. Miller, S. Stumpf, and G.-Z. Yang. Xai—explainable artificial intelligence. *Science Robotics*, 4(37):eaay7120, 2019.

[22] D. Guo. Flow mapping and multivariate visualization of large spatial interaction data. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1041–1048, 2009.

[23] M. Harrower and C. A. Brewer. ColorBrewer. org: An online tool for selecting colour schemes for maps. *The Cartographic Journal*, 40(1):27–37, 2003.

[24] W. He, L. Zou, A. K. Shekar, L. Gou, and L. Ren. Where can we help? A visual analytics approach to diagnosing and improving semantic segmentation of movable objects. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1040–1050, 2021.

[25] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau. Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE Transactions on Visualization and Computer Graphics*, 25(8):2674–2693, 2019.

[26] Y. Hou, C. Wang, J. Wang, X. Xue, X. Zhang, J. Zhu, D. Wang, and S. Chen. Visual evaluation for autonomous driving. *IEEE Transactions on Visualization and Computer Graphics*, 28(01):1030–1039, 2022.

[27] 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*, 28(01):1019–1029, 2022.

[28] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. T. Shen. Discovery of convoys in trajectory databases. *Computer Science*, 1(1):1068–1080, 2009.

[29] J. Kim and J. Canny. Interpretable learning for self-driving cars by visualizing causal attention. In *IEEE International Conference on Computer Vision*, pages 2942–2950, 2017.

[30] S. Kisilevich, F. Mansmann, M. Nanni, and S. Rinzivillo. Spatio-temporal clustering. In *Data Mining and Knowledge Discovery Handbook*, pages 855–874. Springer, 2009.

[31] J. G. Lee, J. Han, and K. Y. Whang. Trajectory clustering: A partition-and-group framework. In *ACM SIGMOD International Conference on Management of Data*, pages 593–604, 2007.

[32] L. Liu, H. Zhan, J. Liu, and J. Man. Visual analysis of traffic data via spatio-temporal graphs and interactive topic modeling. *Journal of Visualization*, 22(1):141–160, 2019.

[33] M. Liu, J. Shi, K. Cao, J. Zhu, and S. Liu. Analyzing the training processes of deep generative models. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):77–87, 2018.

[34] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu. Towards better analysis of deep convolutional neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):91–100, 2017.

[35] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *European Conference on Computer Vision*, pages 21–37. Springer, 2016.

[36] W. Maddern, G. Pascoe, C. Linegar, and P. Newman. 1 year, 1000 km: The oxford robotcar dataset. *The International Journal of Robotics Research*, 36(1):3–15, 2017.

[37] S. Miksch and W. Aigner. A matter of time: Applying a data–users–tasks design triangle to visual analytics of time-oriented data. *Computers & Graphics*, 38:286–290, 2014.

[38] Y. Ming, S. Cao, R. Zhang, Z. Li, Y. Chen, Y. Song, and H. Qu. Understanding hidden memories of recurrent neural networks. In *IEEE Conference on Visual Analytics Science and Technology*, pages 13–24, 2017.

[39] K. Mori, H. Fukui, T. Murase, T. Hirakawa, and H. Fujiyoshi. Visual explanation by attention branch network for end-to-end learning-based self-driving. In *IEEE Intelligent Vehicles Symposium*, pages 1577–1582, 2019.

[40] C.-W. Ngo, T.-C. Pong, and H.-J. Zhang. Motion analysis and segmentation through spatio-temporal slices processing. *IEEE Transactions on Image Processing*, 12(3):341–355, 2003.

[41] T. Ortner, J. Sorger, H. Steinlechner, G. Hesina, H. Piringer, and E. Gröller. Vis-A-Ware: Integrating spatial and non-spatial visualization for visibility-aware urban planning. *IEEE Transactions on Visualization and Computer Graphics*, 23(2):1139–1151, 2016.

[42] Peuquet, Donna, and J. It's about time: A conceptual framework for the representation of temporal dynamics in geographic information systems. *Annals of the Association of American Geographers*, 84(3):441–441, 1994.

[43] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[44] M. Sedlmair, M. Meyer, and T. Munzner. Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2431–2440, 2012.

[45] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-Cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision*, pages 618–626, 2017.

[46] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[47] J. Sorger, T. Ortner, H. Piringer, G. Hesina, and M. E. Gröller. A taxonomy of integration techniques for spatial and non-spatial visualizations. In *Vision, Modeling, and Visualization*, pages 57–64, 2015.

[48] T. Spinner, U. Schlegel, H. Schäfer, and M. El-Assady. explAIner: A visual analytics framework for interactive and explainable machine learning. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1064–1074, 2020.

[49] S. Van, Y. Barnard, and J. Sandberg. The think aloud method: A practical guide to modelling cognitive processes. In *Academic Press*. 1994.

[50] L. Wang, L. Du, X. Ye, Y. Fu, G. Guo, X. Xue, J. Feng, and L. Zhang. Depth-conditioned dynamic message propagation for monocular 3D object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 454–463, 2021.

[51] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda. A survey of autonomous driving: Common practices and emerging technologies. *IEEE Access*, PP(99):1–1, 2020.

[52] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014.

[53] W. Zeng, W. Luo, S. Suo, A. Sadat, and R. Urtasun. End-to-end interpretable neural motion planner. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8660–8669, 2019.

[54] Y. Zhou and O. Tuzel. VoxelNet: End-to-end learning for point cloud based 3D object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.

Junhong Wang is a postgraduate at School of Data Science, Fudan University. She got her bachelor's degree in Information Management and Information Systems from Shanghai University of Finance and Economics. Her main research interests are visual analytics for interpretable machine learning and storytelling in data video.

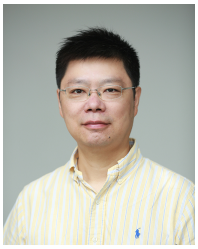
Yun Li is an undergraduate at College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. She is going to further her study at School of Data Science, Fudan University. Her main research interests are visualization and visual analytics for autonomous driving.

Zhaoyu Zhou is a postgraduate at School of Data Science, Fudan University. She got her bachelor's degree in Data Science and Big Data Technology from Fudan University. Her main research interests are visualization and visual analytics for Explainable AI.

Chengshun Wang is a postgraduate at School of Data Science, Fudan University. He got his bachelor's degree in Information Management and Information System from North China Electric Power University. His main research interests are visual analytics for autonomous driving, visualization in finance, and GAN.

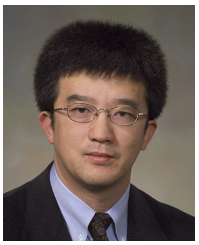
Yijie Hou is a postgraduate at School of Data Science, Fudan University. She got her bachelor's degree in Statistics from Zhejiang University. Her main research interests are visual analytics for autonomous driving.

Li Zhang is a tenure-track Professor at School of Data Science, Fudan University. He was a Research Scientist at Samsung AI Center Cambridge, and a Postdoctoral Research Fellow at the University of Oxford. Prior to joining Oxford, he read his PhD in computer science at Queen Mary University of London. His main research interests are computer vision and deep learning. His research has been published in top computer vision conferences and has topped international computer vision task rankings several times.



Xiangyang Xue is an associate dean at School of Data Science, Fudan University. He got his PhD at School of Telecommunications Engineering, Xidan University. Currently, he is an academic editor of IEEE TCDS, Journal of Computer Research and Development, and Journal of Frontiers of Computer Science & Technology. His research interests include data analysis of multimedia information, deep learning, and artificial intelligence.

Michael Kamp leads the research group *Trustworthy Machine Learning* at the institute for AI in medicine (IKIM) at University Medicine Essen, located at Ruhr-University Bochum. At the same time, he is an affiliated researcher at the Department of Data Science and AI, Faculty of IT, Monash University. He's interested in theoretically sound and practically useful learning approaches in critical applications, such as healthcare and autonomous driving. More information can be found at https://michaelkamp.org.



Xiaolong(Luke) Zhang is an associate professor at College of Information Sciences and Technology, Pennsylvania State University. He got his PhD at School of Information, University of Michigan. His research interests include using advanced interactive technologies like multiscale, virtual environments, and collaboration to improve the understanding of large information structures and the design of interactive tools to support visually-guided navigation in large information space.

Siming Chen is an associate professor at School of Data Science, Fudan University. He was a research scientist at Fraunhofer Institute IAIS and a PostDoc researcher of University of Bonn in Germany. He got his PhD in Peking University. His research interests include visual analytics of social media, cyber security and spatial temporal data. He published several papers in IEEE VIS, IEEE TVCG, EuroVis, etc. More information can be found in http://simingchen.me.