Visual Informatics 9 (2025) 100241

Contents lists available at ScienceDirect

Visual Informatics

journal homepage: www.elsevier.com/locate/visinf

Research article

FlowLLM: Large language model driven flow visualization

Zilin Li, Weihan Zhang, Jun Tao *

School of Computer Science and Engineering, Sun Yat-sen University, Guangdong, China

ARTICLE INFO

Article history:
Received 22 December 2024
Received in revised form 5 March 2025
Accepted 28 April 2025
Available online 2 May 2025

Keywords:
Flow visualization
Natural language interface
Large language model
Interactive exploration

ABSTRACT

Flow visualization is an essential tool for domain experts to understand and analyze flow fields intuitively. In the past decades, various interactive techniques were developed to customize flow visualization for exploration. However, these techniques usually use specifically designed graphical interfaces, requiring considerable learning and usage effort. Recently, FlowNL Huang et al., (2023) introduces a natural language interface to reduce the effort, but it still struggles with natural language ambiguities due to the lack of domain knowledge and provides limited ability to understand the context in dialogues. To address these issues, we propose an explorative flow visualization powered by a large language model that interacts with users. Our approach leverages an extensive dataset of flow-related queries to train the model, enhancing its ability to interpret a wide range of natural language expressions and maintain context over multi-turn interactions. Additionally, we introduce an advanced dialogue management system that supports interactive continuous communication between users and the system. Our empirical evaluations demonstrate significant improvements in user engagement and accuracy of flow structure extraction. These enhancements are crucial for expanding the applicability of flow visualization systems in real-world scenarios, where effective and intuitive user interfaces are paramount.

© 2025 The Author(s). Published by Elsevier B.V. on behalf of Zhejiang University and Zhejiang University Press. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

1. Introduction

Flow visualization plays a pivotal role in the field of scientific visualization by enabling domain experts to interpret complex flow data effectively. The challenge, however, lies in accurately conveying information about specific flow structures of interest in a manner that is most meaningful to the user. Traditionally, visualization techniques have employed either fixed criteria for placing streamlines (Xu et al., 2010) or focused on predefined types of features, such as saddles (Theisel et al., 2003), vortices (Sadlo et al., 2004) or critical points (Ye et al., 2005; Peikert and Sadlo, 2009). While these methods provide a foundation for understanding flow dynamics, they often do not offer the flexibility needed to address the diverse and specific requirements of different applications and user preferences.

Recent advancements in flow visualization have introduced exploratory techniques that enable the customization of visual representations to cater to specific application needs or feature interests. These approaches facilitate user-defined specifications of streamlines through a variety of interfaces, including graph-based interfaces (Tao et al., 2018; Rossl and Theisel, 2012; Han et al., 2020), pattern queries (Wang et al., 2017), predicates (Köhler et al., 2013; Salzbrunn and Scheuermann, 2006), and tangible interfaces (Jackson et al., 2013), with graphical interfaces being the most commonly employed. Despite the flexibility and power of these graph-based interfaces, their complexity necessitates considerable learning and significant cognitive effort. In contrast, tangible interfaces facilitate more intuitive interactions through physical feedback, thereby enhancing user engagement. However, their restricted interaction freedom limits their ability to accommodate complex queries, thus posing challenges for their application in more advanced analytical scenarios.

Motivated by the successes of Natural Language Interfaces (NLIs) for visual data exploration (Yu and Silva, 2020; Narechania et al., 2021; Luo et al., 2022; Ji et al., 2021), FlowNL (Huang et al., 2023) introduced the natural language interaction to the flow visualization. FlowNL is designed to enhance the user experience with the visualization system, promoting an intuitive interaction for a wide range of analysis tasks while minimizing the learning curve and operational complexity. It uses a series of basic elements, such as vector fields, sampled streamlines, associated scalar fields, and geographic regions, to assemble complex flow structures via straightforward operations. For example, the visualization of a "typhoon" is achieved by integrating "spiral flow" extracted from streamlines, "strong wind" indicated by velocity magnitude scalars, and geographic regions.

* Corresponding author.
E-mail addresses: lizlin6@mail2.sysu.edu.cn (Z. Li), zhangwh79@mail2.sysu.edu.cn (W. Zhang), taoj23@mail.sysu.edu.cn (J. Tao).

https://doi.org/10.1016/j.visinf.2025.100241
2468-502X/© 2025 The Author(s). Published by Elsevier B.V. on behalf of Zhejiang University and Zhejiang University Press. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

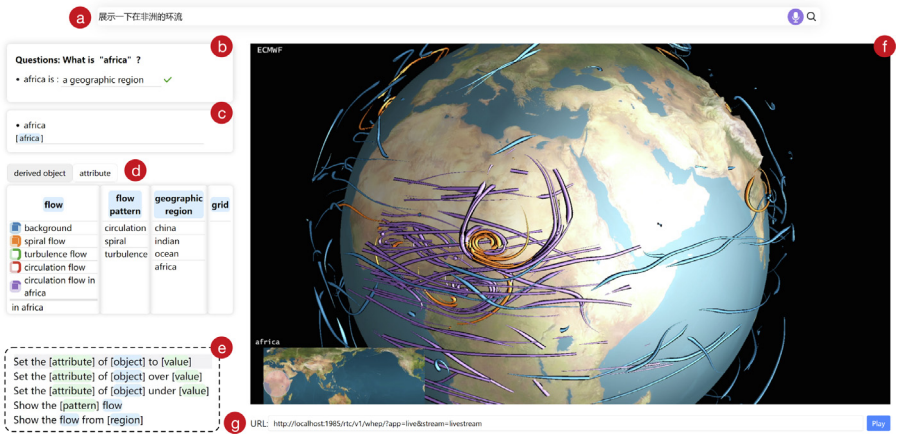


Fig. 1. The interface of FlowLLM. (a) shows the query input in natural language. FlowLLM supports multilingual input, demonstrated here with a Chinese query translated as "Show the tornado". It also facilitates voice input for queries using the microphone button on the right. (b) shows the dialog box that defines an unknown term "tornado" as a spiral flow characterized by a high velocity magnitude. (c) shows the query formula of the derived object corresponding to the definition in (b). (d) displays all primitive and derived objects. (e) demonstrates the pop-up window for specifying thresholds in defining objects that will be shown on demand. (f) visualizes the flow using streamlets, where the orange ones highlight a tornado in the African continent. (g) displays the live streaming link address for viewing FlowLLM remotely. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Despite these advancements, FlowNL faces challenges with natural language ambiguity and strict syntactical constraints that can hinder user interaction and the system's overall effectiveness. Moreover, the lack of support for multi-turn dialogues constrains the system's ability to handle complex, evolving user queries effectively.

In response to these challenges, our work introduces an enhanced version of FlowNL, named FlowLLM, which integrates a large language model to refine the system's natural language processing capabilities. By leveraging extensive sentence datasets for training, our approach not only reduces ambiguity but also introduces greater syntactical flexibility. We have significantly enhanced the system by incorporating support for multi-turn dialogues and multilingual input, allowing users to engage in more natural and conversational interactions. Furthermore, we have broadened the available flow patterns and introduced voice input capabilities. These improvements enable a more seamless exploration and analysis of flow data, thereby boosting user satisfaction and augmenting the system's adaptability across diverse scenarios.

2. Related work

Interactive flow visualization. Interactive techniques have revolutionized the analysis and visualization of flow data, enhancing their utility for a wide array of applications. Graph representations such as the flow web (Xu and Shen, 2010), streamline embedding (Rossl and Theisel, 2012), FlowGraph (Ma et al., 2014), semantic flow graph (Tao et al., 2018), and FlowNet (Han et al., 2020) are key in selecting streamlines and other flow structures. Predominantly, these approaches (Han et al., 2020; Rossl and Theisel, 2012; Ma et al., 2014; Xu and Shen, 2010) rely on a distance metric to guide the creation of graph layouts, whereas semantic data is utilized in the semantic flow graph (Tao et al., 2018) to cluster elements with similar characteristics for more in-depth analysis. In addition to these methods, other planar interactive designs aid in the comprehension and examination of flow fields. Techniques such as the glyph based on radar displays proposed by Hlawatsch et al. (2011), and the interactive comparison of flow data through the straightening of tubular flows by Angelelli and Hauser (2011), exemplify this. Furthermore, tools like streamline predicate (Köhler et al., 2013; Salzbrunn and Scheuermann, 2006), IGScript (Liu et al., 2021), and tangible interface (Jackson et al., 2013) employ similar concepts but differ in their semantic detail allocation. Pattern matching and feature detection also play a crucial role in interactive flow exploration, using the similarity of streamlines (Tao et al., 2016) and pattern matching within flow field regions (Bujack et al., 2015) to identify corresponding patterns. However, these methods may become insufficient when complex flow structures and patterns need to be examined. Thus, Su et al. (2023) presented a novel surface generation and exploration scheme for the steady flow field, guided by a user-specified importance field, which can be a scalar variable associated with the flow field, or can be derived from flow features of interests. Additionally, Zafar et al. (2024) have introduced an interactive system to explore hairpin vortices based on their geometric and physical attributes. Specialized techniques continue to be developed for specific applications and features, such as atmospheric fronts (Kern et al., 2019), PV banners (Bader et al., 2020), vortices (Günther et al., 2017), and splats (Nsonga et al., 2020a,b). Notably, innovative interactive techniques have been employed to investigate the unstable behaviors in the spatial dynamics of blood flow (Lawonn et al., 2016; van Pelt et al., 2011). For visualization of unsteady flows, Krake et al. (2021) enhanced the Dynamic Mode Decomposition (DMD) components and their representations by employing novel visualizations. Furthermore, in immersive flow visualization, Yuan et al. (2023) employed controllers and voice input for 3D exploration, while Su et al. (2021) advanced this approach by incorporating multimodal interaction through head, hand, eye and voice inputs, achieving higher accuracy and greater time efficiency.

Natural language interface. In the evolving landscape of information visualization, NLIs have significantly transformed how users interact with data, simplifying user interaction by transforming natural language queries into intermediate formats like explicit commands (Sun et al., 2010), SQL queries (Dhamdhere et al., 2017), VisFlow functions (Yu and Silva, 2020), and declarative specifications (Narechania et al., 2021). Luo et al. (2022) refined the translation process using a transformer-based model, enhancing systems like Articulate 2 (Kumar et al., 2016), Eviza (Setlur et al., 2016), and Evizeon (Hoque et al., 2018) with interactive dialogue capabilities. DataTone (Gao et al., 2015) employs algorithmic disambiguation with interactive widgets to resolve language ambiguities, and Orko (Srinivasan and Stasko, 2018) uses a combination of grammar and lexicon-based parsing to support diverse interaction modalities. Cui et al. (2020)

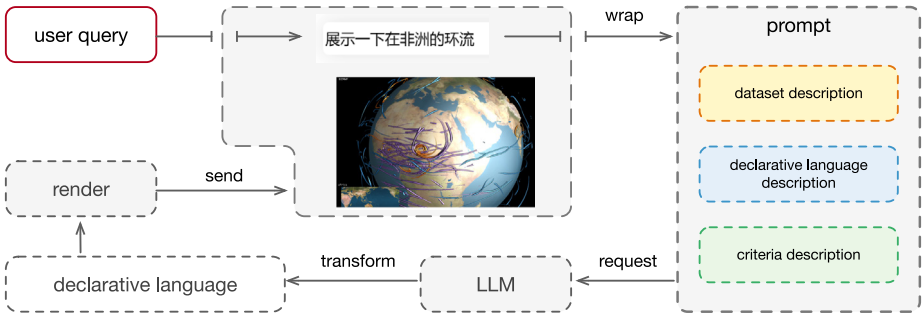


Fig. 2. The workflow of our FlowLLM: When users input a query, it is first wrapped into a specific prompt format to be processed by the LLM. The fine-tuned LLM then transformed it into a declarative language that can be understood by FlowLLM. Finally, the backend of FlowLLM provides the rendered results, which are displayed on the interface.

introduced Text-to-Viz, generating visualizations from visual elements, while Wang et al. (2023) streamlined visualization authoring with structured editing in their NLIs. Mitra et al. (2022) enhanced NL4DV for multi-turn dialogues and nuanced ambiguity resolution. Additionally, Srinivasan and Stasko (2017) and Tory and Setlur (2019) conducted studies to evaluate the effectiveness of NLIs and their impact on user experience, aiming to improve user intent interpretation and response. This categorization focuses on the conversion from natural language to data. Alternatively, the discussion could be approached from various angles by examining the application scenarios of different natural language queries (Shen et al., 2023). Visualization recommendation acts as the back-end engine of NLI to recommend visualizations (Qin et al., 2020; Shen et al., 2022). Natural language interface for databases (Affolter et al., 2019) and natural language generation for visualization (Liu et al., 2020; Tang et al., 2024) are research fields that both utilize natural language to control the presentation of content in the interface. Annotation (Ren et al., 2017; Shi et al., 2020) and narrative storytelling (Tong et al., 2018) enhance visualizations by integrating textual and visual elements together in the interface.

Instruction finetune. In recent years, the field of Large Language Models (LLMs) has witnessed remarkable progress. However, one of the major issues with LLMs is the mismatch between the training objective and the users' objective: LLMs are typically trained to minimize the contextual word prediction error on large corpora; while users want the model to "follow their instructions helpfully and safely" (Brown et al., 2020). Employing retrieval-based knowledge augmentation (Ye et al., 2024) can address certain limitations in the model's understanding of semantics; however, a more effective solution is to adopt Instruction Tuning (IT), which serves as a powerful technique to enhance both the capabilities and controllability of large language models. In the realm of instruction fine-tuning, several notable works have made significant differences across various domains. InstructDial (Gupta et al., 2022) has effectively utilized a diverse set of dialogue tasks to enhance natural language model performance in dialogue evaluation and intent detection, employing models like BART0 (Lin et al., 2022). LINGUIST (Rosenbaum et al., 2022) has shown remarkable improvements in intent classification and slot tagging by fine-tuning the AlexaTM 5B model, particularly excelling in novel intent settings and cross-lingual applications. In the field of text manipulation, CoEdIT (Raheja et al., 2023) and CoPoet (Chakrabarty et al., 2022) have both demonstrated state-of-the-art performance in text editing and poetry writing, respectively, by leveraging instruction-based fine-tuning.

3. Approach

FlowLLM is built upon the foundation of FlowNL (Huang et al., 2023). While sharing the object definition mechanism, FlowLLM improves the language parsing and flow pattern generation of FlowNL. Specifically, FlowLLM leverages a fine-tuned large language model to handle more natural, complex, and multilingual queries regarding flow exploration. Additionally, it incorporates a comprehensive set of flow patterns identified from densely sampled streamline segments. This section briefly introduces the object specification in FlowNL first and then focuses on the workflow of FlowLLM. The workflow illustrates the transformation process from user input to flow visualization as shown in Fig. 2. The rest part of this section follows the workflow's structure for detailed explanation. The details include prompt design, the usage of declarative language, and the fine-tuning of the LLM. The fine-tuning section also introduces the preparation of diverse fine-tuning data, the generation of flow patterns, and the training process.

Fig. 3. Sample declarative specifications generated by the query "Show the flow near indian ocean". The three declarative specifications correspond to the tasks to identify the flow in the specific geographic region, extend the segments of flow, and visualize the derived flow.

3.1. Objects and declarative language

FlowNL categorizes objects into two primary types: primitive and derived objects. Primitive objects are fundamental components, indivisible and inclusive of all data provided by the dataset or system, along with user-defined basic objects such as geographic regions. Derived objects, on the other hand, are formulated from these primitive objects or from other derived objects to delineate flow structures. Our system specifically incorporates two types of primitive objects: structured grids and unstructured points. Structured grids serve as hosts for the flow and scalar fields, embodying attributes of various resolutions. For simplicity,

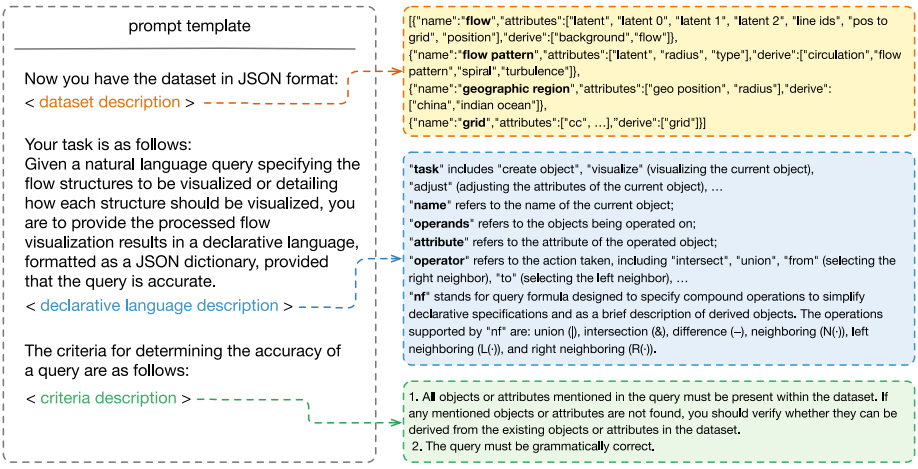


Fig. 4. The prompt template for LLM comprises four fundamental components: a description of the dataset, a description of the task, a description of the declarative language, and a description of the criteria. Each time a query is received, the prompt acts as a default inclusion, transforming the input query into a directive in declarative language to guide the system in understanding what type of visualization content should be presented, ensuring that the user's requirements are precisely met.

our work assumes all fields within a grid share the same resolution, leading to the formation of a singular "grid" primitive object that encompasses all fields. Conversely, unstructured points typically represent flow features such as critical points, sample points on streamlines, and spatial regions, with each point possessing distinct attributes like position, type, and scale.

Derived objects in the FlowNL system are created either from primitive objects or from other derived objects through two primary mechanisms. The first method involves filtering existing objects based on specific criteria. For instance, an object representing saddle points is produced by filtering critical points according to their types. The second method involves combining existing objects using a range of operations available in our system. For example, an object representing a typhoon can be defined as the intersection of a spiral flow pattern with the geographic region of the Northwest Pacific Ocean.

The declarative language meticulously controls visualization parameters and facilitates the creation of new objects, with all objects represented as subsets of elements from corresponding primitive objects. This process utilizes operations such as filtering, mapping, union, intersection, difference, and neighboring. Filtering selects elements based on attributes, from simple one-dimensional ones to complex high-dimensional ones defined by spheres' center vectors and radii. Mapping transforms a derived object's structure, remapping streamlines to grid points based on spatial correspondence. Union, intersection, and difference operations combine or segregate object elements from the same primitive, manipulating their indices to produce desired visual outputs. For objects from different primitives, like merging humid regions with spiral streamlines, the system aligns elements through mapping before applying set operations, ensuring contextual relevance. The resulting object takes on the first operand's primitive type. The neighboring operation broadens an object's spatial context, enhancing its visual impact by including adjacent elements or extending streamline segments. Fig. 3 demonstrates a specific instance of declarative language through the input: "Show the flow near the Indian Ocean".

3.2. Prompt design

While prompting can appear as easy as instructing a human, crafting effective and generalizable prompt strategies is a challenging task. To optimize the performance of large language models in understanding and executing tasks, it is crucial to formulate precise prompts that include detailed requirements and desired output formats, thereby minimizing ambiguities. This can be achieved by clearly defining query specifics, utilizing delimiters, and providing structured steps along with examples. To further enhance reliability and counteract potential inaccuracies, especially in complex topics, prompts should incorporate reference texts, directing the model to ground its responses in validated information. Simplifying complex tasks into manageable subcomponents and leveraging workflows from these simpler outputs can significantly reduce errors and improve the quality of results. Additionally, encouraging a thoughtful reasoning process in the model, akin to a "chain of thought" approach, aids in producing more accurate outcomes. Following these principles, we designed a prompt template incorporating the dataset description, declarative language description, and validation criteria description, as illustrated in Fig. 4.

Dataset description. The prompt encompasses a dataset description listing the primitive objects for building the derived ones. This description consists of the type, name, and associated attributes of each primitive. The existing derived objects are enumerated under their corresponding primitive object as well. Typical primitive objects include the flow pattern, grid, and geographic region. The flow pattern is represented by unstructured spherical regions in a latent space, and its main attributes are the center ("latent") and radius in the latent space. The object "grid" encodes the associated structured fields as attributes, such as "t" (temperature) and "w" (vertical velocity). For the language model to understand the meaning of the attributes, the explanation is included in the description together with the variable name. Note that, to ensure the prompt remains free of irrelevant information, we have omitted details such as specific file paths and file types.

Validation criteria description. The prompt also explicitly specifies criteria for evaluating the validity of user input: First, grammatical accuracy. Should there be any discrepancies, a notification of the grammatical errors will be provided. Second, the appearance of data that is not available in the existing dataset. Upon the identification of such data, we conduct an assessment to determine whether it can be generated from the existing objects in the dataset. For instance, the sentence "Eat the flow of sky". displays grammatical inaccuracies. In contrast, even if the term "Indian" is not explicitly defined, the phrase "Please show me the spiral flow in India" can be interpreted as referring to an undefined geographical location that can be generated through our derivation mechanism. Consequently, FlowLLM would prompt

the user to specify the precise geographical location.In practice, our mapping system operates exclusively on grid-based representations devoid of geographical metadata, consequently limiting the language model's capacity to establish intrinsic correlations. This necessitates manual intervention through selective annotation to establish explicit mappings between geographical terminology and corresponding grid coordinates, thereby enabling the model to comprehend spatial relationships effectively.

Improving accuracy. The performance gap between different prompt strategies can be substantial. Without specifically designed strategies, the large language model often generates incorrect responses that appear to be accurate, which is fatal in generating declarative specification. We emphasize the criteria for correctness in our entire statement. This involves determining the validity of a statement based on whether the object or attribute appears in the dataset; if an unknown object is encountered, we infer its feasibility based on the dataset and our existing operations. To examine the effectiveness of this statement in prompts, we conducted a comparative experiment. This involved using the same dataset and training both a model with and without this specified standard using GPT-3.5-turbo until convergence. The results revealed that when given the simple command "show spiral flow" the model without the directive incorrectly outputs ["task": "visualize", "name": "china spiral flow"] in four out of ten instances. In contrast, the model incorporating the directive accurately produced ["task": "create object", "name": "spiral flow", "operands": ["flow", "spiral"], "operator": "intersect", "nf": "[flow] & [spiral]", "task": "visualize", "name": "spiral flow"] in all ten trials. This experiment underscores the significance of incorporating precise standards in prompt design to enhance the model's performance and reliability.

3.3. Data preparation

Diversity. Since we aim for the large language model to produce corresponding declarative language directly from our natural language inputs, it is necessary to amass substantial data for fine-tuning. The original FlowNL system provided 26 query templates for user utilization. However, the general nouns embedded in these templates can correspond to a diverse array of terms within actual usage. For example, a simple query template like "show flow from [region]" where "[region]" could map to a variety of geographic locations such as China, India, or the Atlantic Ocean. Given that large language models require fine-tuning with user inputs as training data to enhance their understanding of concepts like "[region]", it is insufficient to train the model using a single instance such as "Show flow from China". Instead, it is essential to utilize a variety of geographic terms, such as India or the Pacific Ocean, in place of China to enrich the training dataset. This approach not only broadens the model's exposure to different geographic denominations but also improves its ability to accurately interpret and respond to diverse user queries regarding specific regions. In addition to [region], the template "Color the flow by [attribute]" features numerous possible attributes, such as "q", "vo", "w", and "ciwc" in the context of a grid, and "geo position" and "radius" for geographic regions. To ensure that the large language model accurately recognizes these attribute relationships, it is crucial to train it not only with correctly matched attribute data but also with deliberately incorrect pairings. By inputting sentences that incorrectly associate attributes, the model is trained to recognize and respond to these mismatches. When faced with erroneous attribute relationships, the model will generate error messages, thereby informing the user of the incorrect attribute application. This training approach helps to refine the model's ability to discern and correct attribute-related errors in user queries, enhancing its overall reliability and effectiveness in handling real-world input.

Accuracy. In addition to the object-related issues previously discussed, from Fig. 3 we observe that a specific declarative language for flow visualization encompasses multiple task types. "create object" pertains to generating a new entity, "visualize" refers to the visualization of an existing object. Beyond these, there are also "color" which involves applying color to an object, and "adjust" which entails modifying an object's attributes. To ensure the accuracy of the large language model after fine-tuning, our dataset encompasses all these task types. Additionally, the operations associated with different tasks vary considerably. For example, the "create object" task typically involves the "intersect" operation, which is the most common due to the limited number of our primitive objects. However, to maintain the completeness of the data, we have also designed sentences that include different operators. These operators, such as "from" and "to", correspond to distinct query formulas: right neighboring ($R(\cdot)$) and left neighboring ($L(\cdot)$). This comprehensive approach ensures that the large language model not only understands the direct actions required by various task types but also appropriately applies the complex relationships and operations that govern the interaction with and between data objects in the context of flow visualization.

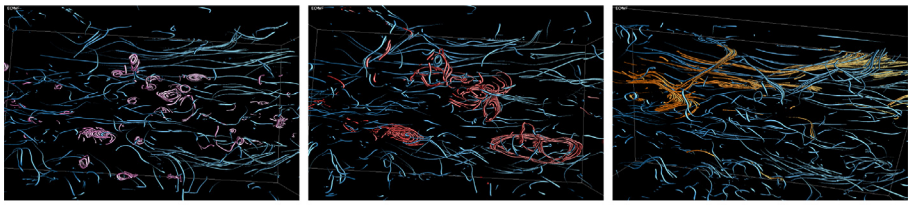
Resolving ambiguity. To effectively enhance the generalizability of large language models while addressing the structured nature of initial inputs and reducing ambiguities, our methodology involves comprehensive modifications to the training data. By rephrasing structured commands into more natural, conversational queries, such as transforming "Show the flow in China". to "Can you display the flow in China?", we train the model to process and respond to diverse linguistic styles. This transformation equips the model to handle a broader array of real-world interactions. In addition to addressing linguistic form, we actively tackle semantic ambiguities that arise from terms with multiple meanings within the dataset. For instance, the term "spiral" might refer to either a flow pattern or a type of critical point. By contextualizing inputs such as "show spiral flow" to indicate a flow pattern and "show spiral critical points" to denote critical points, we direct the model to discern and react to the intended meanings of ambiguous terms. Including such differentiated scenarios in the training set ensures that the model can accurately interpret and respond to ambiguous inputs. Our training dataset comprises approximately 500 queries, among which around 200 have been modified. Through comparative ablation experiments, the fine-tuned model trained on the modified dataset demonstrates approximately a 10% increase in accuracy for natural language inputs.

Multi-turn dialog. Moreover, to enhance the model's ability to maintain context in extended interactions, we incorporate training sequences that simulate a dialogue's memory. An example would be training the model with sequences like "Show spiral flow from China".; "Please color it in red"., teaching it to understand that "it" refers to "spiral flow from China" previously mentioned. This approach improves the model's contextual retention and application, crucial for engaging effectively in dynamic dialogues.

Through these strategies, we ensure that the model is not only adept at understanding and executing queries based on structured inputs but also capable of interpreting nuanced conversations and maintaining coherence across multiple exchanges.

3.4. Flow pattern generation

The FlowNL system supported only a limited number of patterns, specifically three, and it was difficult for users to define new flow patterns by specifying centers and radii in the deep representation space of shapes. FlowLLM randomly samples 500,000



(a) (b) (c)

Fig. 5. The visualization results of different patterns of flow. (a) shows "small-scale spiral flow", (b) shows "large-scale spiral flow", and (c) shows "a flow characterized by a gentle front followed by a sharp change at the rear".

streamline segments and generates common patterns through clustering. Specifically, for each streamline segment, we represent the segment into a 128-dimensional latent vector and cluster all vectors into 50 categories. We then manually examined the display effects of different categories to identify those with clear patterns. These patterns were then summarized. Given the large number of categories, we chose not to display all of them directly on the system interface as the previous FlowNL system did. Instead, we wrote their characteristics into meta files and input them into the large language model for training. This enables the large language model to match the natural language descriptions of patterns with our predefined flow patterns and display them.

We initially employed the BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies) (Zhang et al., 1996) clustering algorithm to cluster the original 500,000 pieces of data into approximately 7,000 subclasses, after which we utilized the k-means algorithm (MacQueen et al., 1967) to further cluster these 7,000 subclasses into the final 50 categories. For the final 50 categories, since they rely solely on data similarity for classification, the matching of these categories to specific flow patterns requires manual verification and comparison. We manually examined the patterns of each corresponding stream and summarized their characteristics. For instance, Fig. 5 represents a small selection of patterns chosen from various flow modes, where Fig. 5(a) demonstrates "small-scale spiral flow", Fig. 5(b) exhibits "large-scale spiral flow" and Fig. 5 (c) displays a flow characterized by a gentle front followed by a sharp change at the rear.

3.5. Training

Regarding the utility of large language models, our initial choice was the Chinese-Alpaca-Pro-7B (Cui et al., 2024), a Chinese-centric model based on LLaMA2. This model expanded the original LLaMA2's vocabulary with Chinese terms and underwent secondary pre-training with Chinese data to enhance its foundational understanding of Chinese semantics. Additionally, the Chinese Alpaca model was fine-tuned with Chinese directive data, significantly improving its capability to comprehend and execute commands. We primarily valued its proficiency in processing Chinese, enabling us to use Chinese data for fine-tuning. However, after fine-tuning with approximately 400 training data entries, the model's performance on the test set was poor, achieving an accuracy of only approximately 0.6, which demonstrated poor generalization and reasoning capabilities.

Consequently, we opted to fine-tune OpenAI's GPT-3.5-turbo instead. After converting our data into the JSONL (JSON Lines) format for the fine-tuning process, both the training loss and validation loss on a set of 400 data items rapidly decreased until converging to zero. The overall training duration consists of two parts: the waiting time in the OpenAI training queue and the actual training time. Based on our multiple training sessions, the waiting time ranges from 5 to 30 min, while the actual training time, depending on our data size, ranges from 40 to 80 min. The training results from our OpenAI model, with both training loss and validation loss reaching zero, indicate that the model consistently achieves convergence after training. Fine-tuning the model enables it to align with our requirements, more effectively translating user queries into commands. Further local testing confirmed that the model could respond correctly. Therefore, we ultimately decided to use the model fine-tuned with GPT-3.5-turbo-0125 for our applications.

4. Interface

The interface of FlowLLM integrates several components crucial for flow exploration, as shown in Fig. 1. These components include the query input box, the dialog box, the object and attribute table, and the flow visualization view.

At the top of the interface lies the query input box, which captures natural language queries from users and forwards these queries to the backend LLM-driven natural language parser. This box actively monitors input as users type, identifying key terms related to objects, attributes, and colors. User input for queries supports both multilingual and voice-enabled methods Fig. 1 (a).

The dialog box plays a vital role in defining unknown terms during interactions, as seen in Fig. 1(b). The initial definitions introduce additional unknown terms, and the dialog expands accordingly, ensuring a comprehensive understanding of all terms.

Flow visualization acts as a pivotal role within the interface, where objects related to flow are depicted as streamlets, and additional entities manifest as point clouds, as illustrated in Fig. 1(f). This component not only renders the data visually but also integrates interactive widgets enabling users to adjust attribute filters across one or two dimensions. The rendering sequence is facilitated by a backend visualization engine, which transmits the visual output to the front end in the form of a video stream. Fig. 1(g) displays the URL link for live streaming. FlowLLM accommodates different rendering transmission signals to cater to various visualization needs.

The object and attribute table, as detailed in Fig. 1(d), systematically organizes derived objects and their respective attributes beneath headers that boldly denote each primitive object. Active tabs disclose comprehensive details about these derived objects, and when visualized, an accompanying glyph is displayed adjacent to the object name. This glyph is equipped with a color legend and a circular slider; the former denotes the color of the object or the applied color map, while the latter is used to modify the object's "weight" — a parameter representing particle density or point opacity.

By selecting an object name, users can view and edit its corresponding query formula, as shown in Fig. 1(c). This functionality is extended to objects defined as filters, where selection widgets appear for parameter adjustment. An example interaction includes clicking on a filter object "high velocity magnitude", prompting the display of the attribute "high velocity" histogram poped in the flow visualization's bottom left corner Fig. 1(e). Users can interact with this histogram to set the value range for "high velocity

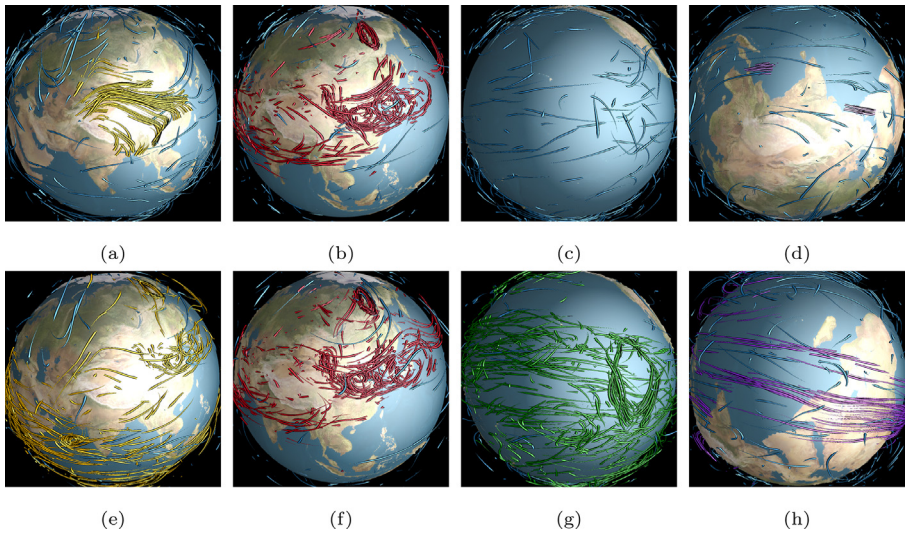


Fig. 6. The visualization results for various ambiguous statements. The top row shows the results produced by FlowNL, and the bottom row shows the outputs of FlowLLM. (a) and (e) correspond to "Show the flow from the Indian Ocean and China".; (b) and (f) correspond to "Show the flow from China with u over 0 or vo under 0".; (c) and (g) correspond to "Show the flow in the Pacific but not to China and to India".; (d) and (h) correspond to "Show the flow from China or the Indian Ocean where velocity magnitude over 20".

magnitude", illustrating a dynamic and interactive approach to data manipulation within the FlowLLM system.

Video transmission. We hope to further optimize the latency of video transmission in FlowNL. FlowNL only uses WebSockets for video transmission, which are prone to packet loss and exhibit considerable communication delays. As we test, the overall latency of FlowNL is approximately 1200 ms. Such high latency will significantly impact the user interaction experience. Additionally, during user operations, issues like stuttering and frame drops are highly likely to occur. After exploring various protocols and solutions, we discovered WebRTC, an open-source, real-time communication solution developed by Google. Essentially, WebRTC is a standard for direct communication between two web browsers, encompassing both Signaling and Media protocols. Signaling addresses the negotiation of capabilities between devices, such as supported codecs, while Media deals with encrypted and low-latency media packet transmission. Ultimately, we employed the open-source SRS project for video transmission, utilizing WHIP and WHEP protocols for signaling and UDP for media transmission, as demonstrated in Fig. 1(g) where we use the WHEP protocol to receive video. By employing this method, we successfully resolved the issues of stuttering, frame dropping, and high latency in video streaming. According to our tests, the final latency is approximately 80 ms in our internal network. Additionally, no frame drops, stuttering, or related performance issues were observed during one hour of continuous operation. Furthermore, we have applied our video streaming technology in a separate external network deployment. Our experimental validation demonstrates that the overall latency for streaming 1080p video under such external network conditions ranges roughly between 300 and 400 ms, depending primarily on the inherent bandwidth and latency characteristics of the network. Although the latency observed is higher than that within the internal network scenario, we have found the overall usability to remain satisfactory and observed no frame drops or related performance issues during the tests. Thus, we have achieved smoother video transmission through WebRTC technology, significantly enhancing the system's usability.

5. Evaluation

While the overall usability of this interaction scheme is examined in FlowNL (Huang et al., 2023), this evaluation focuses on the additional features introduced by FlowLLM. First, we conducted a user study with two visualization researchers as participants to thoroughly investigate the practical differences between FlowLLM and FlowNL, focusing specifically on several interaction aspects. Then, we studied FlowLLM's ability to resolve ambiguity. As this study focuses on logical ambiguity, we recruited nine participants to compare the outputs of FlowLLM and FLowNL, when responding to queries with logical ambiguity. In addition, we design several test cases to examine FlowLLM's usability in processing complex queries and interacting with users through multi-turn dialogues. Lastly, we also evaluated FlowLLM's ability to support multilingual queries and voice input.

5.1. User study

We conducted a user study involving two researchers experienced in visualization. Both participants have extensive experience of more than three years in visualization-related research. However, the participants differed in their previous exposure to flow visualization research. Specifically, one participant had extensive domain-specific expertise and prior research experience in flow visualization, while the other participant, though having a strong foundational expertise in visualization, had no prior experience specifically related to flow visualization. Such diversity in domain-specific familiarity among participants was intentionally considered to ensure more comprehensive and balanced evaluation results.

The study process followed a structured procedure. Participants were asked to first engage with FlowNL and subsequently with FlowLLM. Initially, each participant completed a series of predefined visualization tasks based on given instructions provided by the investigator. Subsequently, participants were allowed an open exploration period without further explicit instruction, enabling them to freely interact with and comprehensively evaluate the features of each system. After completing the directed tasks and open exploration, participants rated both FlowNL and FlowLLM systems based upon standardized evaluation metrics, including usability, interaction fluency, ease of learning, and accuracy in interpreting user input. They also provided qualitative feedback regarding subjective experience, user-friendliness, and cognitive workload incurred during their interactions.

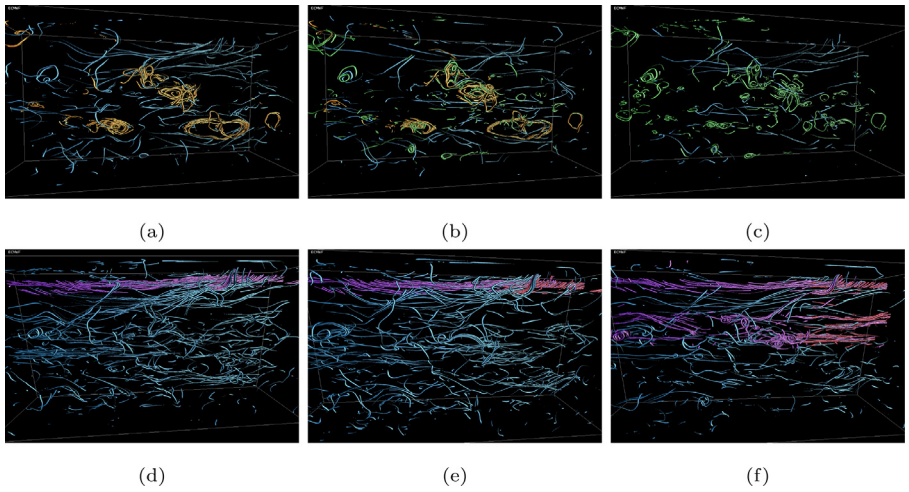


Fig. 7. The visualization results using the ECMWF dataset. The row above, labeled (a)–(c), displays the results of multiple turns of queries for different patterns of flow. The lower row, labeled (d)–(f), displays the results of operations on different objects during multiple rounds of dialogue when querying circulation flow.

The results of the evaluation indicated a clear preference toward FlowLLM among both participating researchers. According to their ratings and qualitative feedback, FlowLLM demonstrated substantial advantages in overall use fluency, usability, and ease of operation compared to FlowNL. In particular, the participant with prior flow visualization experience emphasized that FlowLLM was particularly beneficial for efficiently interpreting subtle or ambiguous inputs, allowing for expedited exploration and analysis. Meanwhile, the participant without prior domain-specific flow expertise indicated that FlowLLM noticeably lowered interaction barriers, facilitating intuitive exploration and comprehension of visualization tasks despite limited previous familiarity with the flow domain.

5.2. Ability to resolve ambiguity

In the evaluation aimed at resolving ambiguities, we first designed statements that could potentially lead to semantic ambiguities. Subsequently, through user surveys, we gathered statistics on different users' interpretations of these statements. Finally, we compared the outputs generated by our FlowLLM system and the FlowNL system for these ambiguous statements to verify the effectiveness of our system.

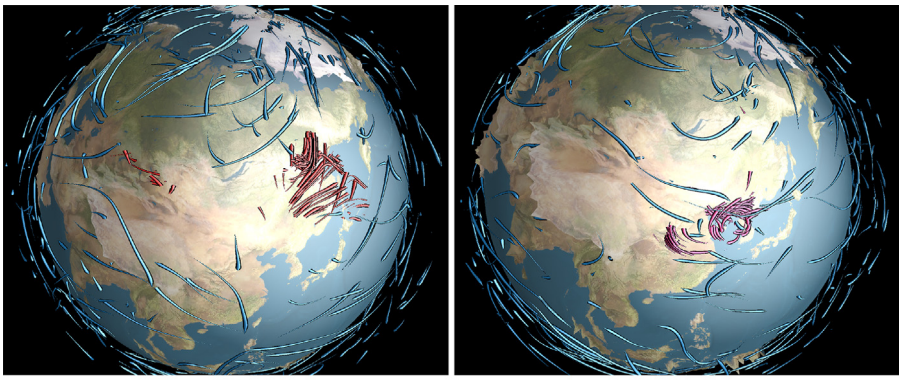
Participants. We recruited nine unpaid participants, each with a background in either computer science or a scientific domain. Two participants hold PhD degrees, five possess master's degrees, and the remaining two have bachelor's degrees. Regarding academic experience, five participants have backgrounds in mechanical engineering, physics, or atmospheric research. Four participants have research or work experience in interactive techniques, including one with atmospheric research experience. Additionally, one participant specializes in the area of databases. Four participants reported familiarity with geographic information systems, ParaView, or similar tools, although none have experience with interactive flow visualization systems or natural language interfaces.

Query design. In addition to the previously mentioned semantic ambiguity and quantitative ambiguity, we place greater emphasis on logical ambiguity. Logical ambiguity mainly involves logical conjunctions such as "and" and "or". A specific example is "Show flow in the Pacific but not to China and India". This statement can typically be interpreted in four ways. First, it may mean showing flows within the Pacific that neither flow to China nor to India. Second, it could indicate showing flows within the Pacific that do not flow to China or showing flows to India. Third, it might mean showing flows within the Pacific that do not flow to China or showing flows within the Pacific that do flow to India. Fourth, it could represent showing flows within the Pacific that do not flow to China and do flow to India. Another specific example involving "or" is "Show the flow from China where PV is under 40 or velocity magnitude is over 100". This sentence can also be understood in two ways: First, it could mean showing the flow from within China where the PV is under 40 or showing the flow from areas where the velocity magnitude is over 100. Second, it might mean showing the flow from within China where the PV is under 40, or showing the flow from within China where the velocity magnitude is over 100. We have crafted a total of twelve sentences of this nature.

Results comparison. Fig. 6 illustrates the disparate visualization outcomes generated by FlowNL and FlowLLM when confronted with queries of an ambiguous nature. Fig. 6(a) and (e) pertain to the input "Show the flow from the Indian Ocean and China". In our survey, over half of the respondents interpreted this as "Display flows originating either from the Indian Ocean or from China". a sentiment echoed by FlowLLM. However, a minority perceived it as "Display flows originating from both the Indian Ocean and China". whereas the original FlowNL erroneously parsed it as "(([flow] & [from Indian Ocean]) & [China])", meaning it displayed flows that are from the Indian Ocean and also in China, failing to grasp the true intent of "and China".

Fig. 6(b) and (f) relate to the statement "Show the flow from China with u over 0 or vo under 0". Most respondents understood this to mean "Display flows originating from within China, where 'u' is greater than 0 or 'vo' is less than 0". FlowLLM also adhered to this interpretation, whereas FlowNL misinterpreted it as "(([flow] & [from China]) & [u over 0]) & [vo under 0]", suggesting it displayed flows from China where 'u' is greater than 0 and 'v' is less than 0, indicating a misunderstanding of the logical operator "or".

Fig. 6(c) and (g) show the response to the command: "Show the flow in the Pacific but not to China and to India". This phrase, as we have discussed previously, is open to multiple interpretations due to its complexity. FlowLLM interpreted the command effectively as "Display flows within the Pacific that do not head towards China or India". This interpretation aligns with a logical understanding of the directive, effectively excluding flows directed toward these specific destinations. On the other hand, FlowNL parsed the input as towards China or India". while FlowNL parsed it as "[flow] & [in Pacific not to China] & [to India]",



(a) (b)

Fig. 8. The visualization results of queries containing complex semantic information. (a) displays the flow in the high-latitude regions of China, primarily concentrated in the northwest and northeast areas. (b) displays the result of querying China's spiral flow and highlighting it, mainly concentrated in the North China region and displayed in pink. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

which introduces a logical contradiction and does not correspond to any actual flow patterns.

Finally, Fig. 6(d) and (h) are linked to the input "Show the flow from China or the Indian Ocean where velocity magnitude over 20". FlowLLM correctly captured the intent as "Display flows originating from either China or the Indian Ocean with a velocity magnitude exceeding 20". aligning with the majority's view from our survey. Conversely, FlowNL's parsing as "[flow] & [from China] & [Indian Ocean velocity magnitude over 20]" incorrectly attributes the velocity condition directly to "Indian Ocean".

We have presented only a subset of the comparative test results, yet these are sufficient to demonstrate significant enhancements in our FlowLLM system's ability to parse complex, ambiguous statements. Additionally, we provide support for users to modify and define these ambiguous entities themselves, further enhancing the system's adaptability and user interaction capabilities. This capability not only improves the accuracy of data interpretation but also empowers users to tailor the system's responses to their specific analytical needs, thereby optimizing both the utility and the user experience.

5.3. Ability to answer complex queries

When handling natural language tasks, dealing with semantic context is a complex challenge. In addition to the ambiguities of language mentioned previously, addressing typical and common issues presents a significant challenge. The context and history of a conversation are critical for understanding the meaning of a particular statement. For instance, a question asked earlier in a conversation can influence the interpretation of a subsequent response. Utilizing large language models to process natural language allows for greater flexibility during query input. This approach is particularly effective in handling complex semantic contexts. FlowNL's natural language parser constructs visualization instructions based on grammatical structures, overlooking the semantic context. In this section, we evaluate the capability of FlowLLM to handle complex, lengthy queries, thereby assessing its proficiency in semantic extraction. We also employ a multi-turn dialogue test to examine its capacity to maintain a record of contextual history. In addition, we also introduce its user-friendly features.

Semantic context interpretation. We begin by constructing a complex composite query input: "Display the flow from China; illustrate its segment at high latitudes; adjust its wind speed to 10; and update its color to red". For such a complex input statement, FlowNL is incapable of parsing it because this sentence does not conform to traditional grammatical structures and cannot be transformed into declarative language commands through a natural language parser. For FlowLLM, it successfully parsed the correct declarative language commands and generated the appropriate visual results. It should also be noted that by examining the intermediate declarative statements generated, we observed that FlowLLM had accurately extracted the semantic context. The query content described regions at high latitudes, which poses a challenge for precise localization. FlowLLM identified the expression "high latitude regions" and quantitatively converted it into a parameter for filtering attributes at latitudes above 60 degrees North, demonstrating effective semantic understanding and translation. We examine the relationships between the content before and after the statement to assess the coherence and continuity of the narrative. In the query statement, the referents of the pronouns are actually different. The first pronoun refers to the flow data from China, hence when mentioning high latitudes, it specifically pertains to the flow at high latitudes in China. When it comes to the latter pronoun "its", it refers to the high-latitude flow in China, which is semantically distinct from the first "its". This differentiation highlights the importance of context in interpreting pronoun references within complex statements. This demonstrates that even when faced with complex sentences, FlowLLM is capable of understanding and converting them into a series of declarative language commands, progressively identifying the specific data to be observed. We present the visualization results as shown in Fig. 8(a), where the red flow is concentrated in the northwest and northeast regions of China. We examine another example: "Show the spiral flow in China; highlight it in a brighter color". In this statement, there is also the task of understanding what constitutes a "brighter" color. FlowLLM comprehended the semantics and colored "it" pink. We have displayed this visualization result in Fig. 8(b). The illustration highlights the occurrence of spiral flow in the North China region.

Multi-turn interactions. In the process of interacting through natural language, a typical scenario requires the system to have the capability to remember historical context. FlowLLM fulfills this requirement through the completion of a large language model. We conduct case studies on the European Centre for Medium-Range Weather Forecasts (ECMWF) dataset. We aim to check whether FlowLLM can accurately distinguish actions and objects in continuous natural language interaction. Flow exhibits different patterns based on various morphological characteristics. FlowLLM obtains different latent vectors by encoding the shapes of streamlines and clusters them in the embedding space to distinguish more patterns.

We first input the query "Show the large-scale spiral flow". As shown in Fig. 7(a), this is a newly added flow pattern, and its color

is pink. We want to compare it with the normal spiral flow, so we input the query "Show the spiral flow". We can observe that the small yellow spiral flows are dispersed throughout the space Fig. 7(b). At this point, we need to hide the previous flows to display the spiral flow alone clearly. Now, we only need to query "Hide the former". to hide the large-scale spiral flow mentioned in the previous conversation history Fig. 7(c). This demonstrates that even after multiple turns of conversational interference, FlowLLM can still understand the context referred to in the query. We provide another example to illustrate the capability of FlowLLM to handle continuous query requests. Querying "Show the circulation flow with low latitude". leads to visualization results in Fig. 7 (d). Following up, we provide a more complex query that requires the use of historical context: "Color its part with high longitude in red". Based on the results from Fig. 7(e), FlowLLM selected a portion of the turbulence flow on the right side of the space and changed its color to red. The right side corresponds to the high-longitude areas in geographic information. From these results, it is evident that the query, which includes a filter for the previously mentioned object, can be understood, and it is capable of capturing the position information corresponding to high longitude. Therefore, after these two turns of input, whether FlowLLM can remember the content of both queries and make necessary changes needs to be verified. We immediately follow up with the query "Expand them to show at high latitudes". and the results are as shown in Fig. 7(f). Not only the orange turbulence flow from the first query result but also the red flow from the high longitude part has similarly increased its presence in the high latitude areas. This indicates that FlowLLM takes into account the content mentioned in previous history when dealing with multiple turns of dialogue and can use the objects operated on in previous actions for further control.

5.4. Ability for domain-agnostic exploration

When dealing with data from diverse domains, our model can easily adapt by simply modifying the dataset description within the prompt to specify the characteristics of the new data. This approach allows us to achieve quick and cost-effective transfer across various domains. We validated this finding through a comparative experiment conducted on the vascular flow dataset, VSFS9. Initially, without any parameter fine-tuning, we directly utilized the fine-tuned model and adapted it to the new domain simply by modifying the dataset description presented in the prompt. Specifically, we explicitly defined and included critical attributes of our dataset — such as flow velocity, flow rate, and other relevant physical parameters — in the dataset description to assess the model's intrinsic capability to understand novel domain-specific inputs.

Subsequently, we conducted a fine-tuning stage to further investigate if explicit exposure to domain-specific query-output pairs would significantly improve domain adaptation performance. During fine-tuning, we continued using the modified prompt description but additionally provided training examples consisting of queries from the VSFS9 dataset along with their expected outputs. The key objective was to determine if direct training on domain-specific input–output relationships provides meaningful superiority compared with adaptation based solely on prompt modification.

To objectively evaluate the results of these two methods, we compared the output accuracy achieved by both approaches on the VSFS9 test set. Interestingly, the experimental results indicated only negligible distinctions in accuracy between the two strategies; the model exhibited strong performance in correctly interpreting new domain data solely through dataset description adjustment, achieving nearly equivalent accuracy as compared to the fully fine-tuned version. This crucial observation indicates that the primary contribution of the fine-tuning process is not merely to memorize specific domain-dependent data patterns, but rather to enable the model to understand and effectively utilize the underlying structure and semantics of our declarative language framework across diverse data domains.

5.5. Ability to facilitate user-friendly interaction

We have integrated many practical features into FlowLLM to make parts of the query more flexible and user-friendly. First, we have incorporated a voice input conversion feature. This not only enhances the convenience and efficiency of human–computer interaction but also makes the input more natural. Additionally, by leveraging the integration of the large language model, we have enabled multilingual input methods. Whether it is English, Chinese, or other languages, they can all be converted into the same information encoding through the tokenizer of the large language model. Our fine-tuned large language model can transform input content from different languages into declarative language commands to display various visualization results. If users can query in their native language, there will be significant differences in the efficiency and accuracy of their expressions.

Additionally, leveraging the representational capabilities of large language models, there is notable flexibility in handling input queries. In FlowNL's natural language parser, the grammatical structure is singular, requiring specific query formulations like "Show the flow with [attribute]". and "Color the [pattern] flow"., limiting user flexibility. We tested more casual queries, and FlowLLM handled them quite well. The extraction of semantic information from queries has been discussed previously; here, we note that FlowLLM allows more natural phrasing. For instance, the query "I want to see some flow and color it in black". resembles everyday language, enabling flexible articulation of desired visualizations, which FlowLLM manages effectively. Additionally, we tested combined features, such as inputting Chinese through the voice button, and the system accurately displayed the intended flow, as illustrated in Fig. 1, we tested the query "Show the tornado". in Chinese, where orange streamlines represent the tornado. These results indicate that FlowLLM can accurately process casual queries.

5.6. Limitations

Lack of guaranteed accuracy. While FlowLLM maintains a high accuracy in answering users' queries in general, it does not guarantee to generate valid specifications in some rare cases, due to the nature of learning-based approach. For example, even when the contents corresponding to "task", "operands", and "operator" are correct, the output for "nf" (formula) can be inconsistent, particularly when dealing with complex sentence inputs. While the formula can be fixed by the generation rules of FlowNL in this case, users may need to manually edit the formula or specification when the error is not detectable.

Visualization techniques. Our FlowLLM system is currently limited to visualizing various objects solely through streamlet animation and point clouds, which might not effectively expose complex structures. To address this limitation, it is advisable to implement more sophisticated techniques, such as integral surfaces. Moreover, enhancing the rendering of scalar features could be achieved through techniques like direct volume rendering or isosurface rendering.

6. Conclusion

We have proposed FlowLLM, which resolved the issues of ambiguity and the inability to engage in multi-turn dialogues in the original FlowNL by fine-tuning a large language model. Furthermore, FlowLLM now accommodates inputs in multiple languages, greatly extending its accessibility and improving its utility. We now also support voice input, enhancing adaptability across various interaction scenarios. Last but not least, the previously erratic, high-latency video transmission has been replaced, providing a low-latency, seamless video display that significantly smooths system interactions.

CRediT authorship contribution statement

Zilin Li: Writing – original draft, Software, Methodology, Investigation. **Weihan Zhang:** Writing – original draft, Validation, Resources. **Jun Tao:** Writing – review & editing, Supervision, Conceptualization.

Ethical approval

This study does not contain any studies with human or animal subjects performed by any of the authors.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported in part by the National Natural Science Foundation of China through grants 62172456 and 62372484.

Appendix A. Supplementary data

For a demonstration of the dynamic interactions of FlowLLM system, please refer to the video in the supplementary material at https://doi.org/10.1016/j.visinf.2025.100241.

References

Affolter, K., Stockinger, K., Bernstein, A., 2019. A comparative survey of recent natural language interfaces for databases. VLDB J. 28 (5), 793–819.

Angelelli, P., Hauser, H., 2011. Straightening tubular flow for side-by-side visualization. IEEE Trans. Vis. Comput. Graphics 17 (12), 2063–2070. http://dx.doi.org/10.1109/TVCG.2011.235.

Bader, R., Sprenger, M., Ban, N., Rüdisühli, S., Schär, C., Günther, T., 2020. Extraction and visual analysis of potential vorticity banners around the alps. IEEE Trans. Vis. Comput. Graphics 26 (1), 259–269. http://dx.doi.org/10.1109/TVCG.2019.2934310.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al., 2020. Language models are few-shot learners. Adv. Neural Inf. Process. Syst. 33, 1877–1901.

Bujack, R., Kasten, J., Hotz, I., Scheuermann, G., Hitzer, E., 2015. Moment invariants for 3D flow fields via normalization. In: 2015 IEEE Pacific Visualization Symposium (PacificVis). pp. 9–16. http://dx.doi.org/10.1109/PACIFICVIS.2015.7156350.

Chakrabarty, T., Padmakumar, V., He, H., 2022. Help me write a poem: Instruction tuning as a vehicle for collaborative poetry writing. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, pp. 6848–6863. http://dx.doi.org/10.18653/v1/2022.emnlp-main.460.

Cui, Y., Yang, Z., Yao, X., 2024. Efficient and effective text encoding for Chinese LLaMA and Alpaca. arXiv:2304.08177.

Cui, W., Zhang, X., Wang, Y., Huang, H., Chen, B., Fang, L., Zhang, H., Lou, J.-G., Zhang, D., 2020. Text-to-Viz: Automatic generation of infographics from proportion-related natural language statements. IEEE Trans. Vis. Comput. Graphics 26 (1), 906–916. http://dx.doi.org/10.1109/TVCG.2019.2934785.

Dhamdhere, K., McCurley, K.S., Nahmias, R., Sundararajan, M., Yan, Q., 2017. Analyza: Exploring data with conversation. In: Proceedings of the 22nd International Conference on Intelligent User Interfaces. IUI '17, Association for Computing Machinery, New York, NY, USA, pp. 493–504. http://dx.doi.org/10.1145/3025171.3025227.

Gao, T., Dontcheva, M., Adar, E., Liu, Z., Karahalios, K.G., 2015. DataTone: Managing ambiguity in natural language interfaces for data visualization. In: Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology. UIST '15, Association for Computing Machinery, New York, NY, USA, pp. 489–500. http://dx.doi.org/10.1145/2807442.2807478.

Günther, T., Gross, M., Theisel, H., 2017. Generic objective vortices for flow visualization. ACM Trans. Graph. 36 (4), http://dx.doi.org/10.1145/3072959.3073684.

Gupta, P., Jiao, C., Yeh, Y.-T., Mehri, S., Eskenazi, M., Bigham, J., 2022. InstructDial: Improving zero and few-shot generalization in dialogue through instruction tuning. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, pp. 505–525. http://dx.doi.org/10.18653/v1/2022.emnlp-main.33.

Han, J., Tao, J., Wang, C., 2020. FlowNet: A deep learning framework for clustering and selection of streamlines and stream surfaces. IEEE Trans. Vis. Comput. Graphics 26 (4), 1732–1744. http://dx.doi.org/10.1109/TVCG.2018.2880207.

Hlawatsch, M., Leube, P., Nowak, W., Weiskopf, D., 2011. Flow radar glyphs—Static visualization of unsteady flow with uncertainty. IEEE Trans. Vis. Comput. Graphics 17 (12), 1949–1958. http://dx.doi.org/10.1109/TVCG.2011.203.

Hoque, E., Setlur, V., Tory, M., Dykeman, I., 2018. Applying pragmatics principles for interaction with visual analytics. IEEE Trans. Vis. Comput. Graphics 24 (1), 309–318. http://dx.doi.org/10.1109/TVCG.2017.2744684.

Huang, J., Xi, Y., Hu, J., Tao, J., 2023. FlowNL: Asking the flow data in natural languages. IEEE Trans. Vis. Comput. Graphics 29 (1), 1200–1210. http://dx.doi.org/10.1109/TVCG.2022.3209453.

Jackson, B., Lau, T.Y., Schroeder, D., Toussaint, K.C., Keefe, D.F., 2013. A lightweight tangible 3D interface for interactive visualization of thin fiber structures. IEEE Trans. Vis. Comput. Graphics 19 (12), 2802–2809. http://dx.doi.org/10.1109/TVCG.2013.121.

Ji, X., Tu, Y., He, W., Wang, J., Shen, H.-W., Yen, P.-Y., 2021. USEVis: Visual analytics of attention-based neural embedding in information retrieval. Vis. Informatics 5 (2), 1–12. http://dx.doi.org/10.1016/j.visinf.2021.03.003.

Kern, M., Hewson, T., Schätler, A., Westermann, R., Rautenhaus, M., 2019. Interactive 3D visual analysis of atmospheric fronts. IEEE Trans. Vis. Comput. Graphics 25 (1), 1080–1090. http://dx.doi.org/10.1109/TVCG.2018.2864806.

Köhler, B., Gasteiger, R., Preim, U., Theisel, H., Gutberlet, M., Preim, B., 2013. Semi-automatic vortex extraction in 4D PC-MRI cardiac blood flow data using line predicates. IEEE Trans. Vis. Comput. Graphics 19 (12), 2773–2782. http://dx.doi.org/10.1109/TVCG.2013.189.

Krake, T., Reinhardt, S., Hlawatsch, M., Eberhardt, B., Weiskopf, D., 2021. Visualization and selection of dynamic mode decomposition components for unsteady flow. Vis. Informatics 5 (3), 15–27. http://dx.doi.org/10.1016/j.visinf.2021.06.003.

Kumar, A., Aurisano, J., Di Eugenio, B., Johnson, A., Gonzalez, A., Leigh, J., 2016. Towards a dialogue system that supports rich visualizations of data. In: Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue. pp. 304–309.

Lawonn, K., Glaßer, S., Vilanova, A., Preim, B., Isenberg, T., 2016. Occlusion-free blood flow animation with wall thickness visualization. IEEE Trans. Vis. Comput. Graphics 22 (1), 728–737. http://dx.doi.org/10.1109/TVCG.2015.2467961.

Lin, B.Y., Tan, K., Miller, C., Tian, B., Ren, X., 2022. Unsupervised cross-task generalization via retrieval augmentation. In: Advances in Neural Information Processing Systems. vol. 35, Curran Associates, Inc., pp. 22003–22017.

Liu, R., Gao, M., Ye, S., Zhang, J., 2021. Igscript: An interaction grammar for scientific data presentation. In: Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems. pp. 1–13.

Liu, C., Xie, L., Han, Y., Wei, D., Yuan, X., 2020. AutoCaption: An approach to generate natural language description from visualization automatically. In: 2020 IEEE Pacific Visualization Symposium (PacificVis). IEEE, pp. 191–195.

Luo, Y., Tang, N., Li, G., Tang, J., Chai, C., Qin, X., 2022. Natural language to visualization by neural machine translation. IEEE Trans. Vis. Comput. Graphics 28 (1), 217–226. http://dx.doi.org/10.1109/TVCG.2021.3114848.

Ma, J., Wang, C., Shene, C.-K., Jiang, J., 2014. A graph-based interface for VisualAnalytics of 3D streamlines and pathlines. IEEE Trans. Vis. Comput. Graphics 20 (8), 1127–1140. http://dx.doi.org/10.1109/TVCG.2013.236.

MacQueen, J., et al., 1967. Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. vol. 1, (14), Oakland, CA, USA, pp. 281–297.

Mitra, R., Narechania, A., Endert, A., Stasko, J., 2022. Facilitating conversational interaction in natural language interfaces for visualization. In: 2022 IEEE Visualization and Visual Analytics. VIS, pp. 6–10. http://dx.doi.org/10.1109/VIS54862.2022.00010.

Narechania, A., Srinivasan, A., Stasko, J., 2021. NL4DV: A toolkit for generating analytic specifications for data visualization from natural language queries. IEEE Trans. Vis. Comput. Graphics 27 (2), 369–379. http://dx.doi.org/10.1109/TVCG.2020.3030378.

Nsonga, B., Niemann, M., Fröhlich, J., Staib, J., Gumhold, S., Scheuermann, G., 2020a. Detection and visualization of splat and antisplat events in turbulent flows. IEEE Trans. Vis. Comput. Graphics 26 (11), 3147–3162. http://dx.doi.org/10.1109/TVCG.2019.2920157.

Nsonga, B., Scheuermann, G., Gumhold, S., Ventosa-Molina, J., Koschichow, D., Fröhlich, J., 2020b. Analysis of the near-wall flow in a turbine cascade by splat visualization. IEEE Trans. Vis. Comput. Graphics 26 (1), 719–728. http://dx.doi.org/10.1109/TVCG.2019.2934367.

Peikert, R., Sadlo, F., 2009. Topologically relevant stream surfaces for flow visualization. In: Proceedings of the 25th Spring Conference on Computer Graphics. SCCG '09, Association for Computing Machinery, New York, NY, USA, pp. 35–42. http://dx.doi.org/10.1145/1980462.1980472.

van Pelt, R., Olivan Bescos, J., Breeuwer, M., Clough, R.E., Groller, M.E., ter Haar Romenij, B., Vilanova, A., 2011. Interactive virtual probing of 4D MRI blood-flow. IEEE Trans. Vis. Comput. Graphics 17 (12), 2153–2162. http://dx.doi.org/10.1109/TVCG.2011.215.

Qin, X., Luo, Y., Tang, N., Li, G., 2020. Making data visualization more efficient and effective: a survey. VLDB J. 29 (1), 93–117.

Raheja, V., Kumar, D., Koo, R., Kang, D., 2023. CoedIT: Text editing by task-specific instruction tuning. In: Findings of the Association for Computational Linguistics: EMNLP 2023. Association for Computational Linguistics, Singapore, pp. 5274–5291. http://dx.doi.org/10.18653/v1/2023.findings-emnlp.350.

Ren, D., Brehmer, M., Lee, B., Höllerer, T., Choe, E.K., 2017. Chartaccent: Annotation for data-driven storytelling. In: 2017 IEEE Pacific Visualization Symposium (PacificVis). Ieee, pp. 230–239.

Rosenbaum, A., Soltan, S., Hamza, W., Versley, Y., Boese, M., 2022. LINGUIST: Language model instruction tuning to generate annotated utterances for intent classification and slot tagging. In: Proceedings of the 29th International Conference on Computational Linguistics. International Committee on Computational Linguistics, Gyeongju, Republic of Korea, pp. 218–241.

Rossl, C., Theisel, H., 2012. Streamline embedding for 3D vector field exploration. IEEE Trans. Vis. Comput. Graphics 18 (3), 407–420. http://dx.doi.org/10.1109/TVCG.2011.78.

Sadlo, F., Peikert, R., Parkinson, E., 2004. Vorticity based flow analysis and visualization for pelton turbine design optimization. In: IEEE Visualization 2004. pp. 179–186. http://dx.doi.org/10.1109/VISUAL.2004.128.

Salzbrunn, T., Scheuermann, G., 2006. Streamline predicates. IEEE Trans. Vis. Comput. Graphics 12 (6), 1601–1612. http://dx.doi.org/10.1109/TVCG.2006.104.

Setlur, V., Battersby, S.E., Tory, M., Gossweiler, R., Chang, A.X., 2016. Eviza: A natural language interface for visual analysis. In: Proceedings of the 29th Annual Symposium on User Interface Software and Technology. UIST '16, Association for Computing Machinery, New York, NY, USA, pp. 365–377. http://dx.doi.org/10.1145/2984511.2984588.

Shen, L., Shen, E., Luo, Y., Yang, X., Hu, X., Zhang, X., Tai, Z., Wang, J., 2022. Towards natural language interfaces for data visualization: A survey. IEEE Trans. Vis. Comput. Graphics 29 (6), 3121–3144.

Shen, L., Shen, E., Luo, Y., Yang, X., Hu, X., Zhang, X., Tai, Z., Wang, J., 2023. Towards natural language interfaces for data visualization: A survey. IEEE Trans. Vis. Comput. Graphics 29 (6), 3121–3144. http://dx.doi.org/10.1109/TVCG.2022.3148007.

Shi, D., Xu, X., Sun, F., Shi, Y., Cao, N., 2020. Calliope: Automatic visual data story generation from a spreadsheet. IEEE Trans. Vis. Comput. Graphics 27 (2), 453–463.

Srinivasan, A., Stasko, J., 2017. Natural language interfaces for data analysis with visualization: considering what has and could be asked. In: Proceedings of the Eurographics/IEEE VGTC Conference on Visualization: Short Papers. EuroVis '17, Eurographics Association, Goslar, DEU, pp. 55–59. http://dx.doi.org/10.2312/eurovisshort.20171133.

Srinivasan, A., Stasko, J., 2018. Orko: Facilitating multimodal interaction for visual exploration and analysis of networks. IEEE Trans. Vis. Comput. Graphics 24 (1), 511–521. http://dx.doi.org/10.1109/TVCG.2017.2745219.

Su, C., Yang, C., Chen, Y., Wang, F., Wang, F., Wu, Y., Zhang, X., 2021. Natural multimodal interaction in immersive flow visualization. Vis. Informatics 5 (4), 56–66. http://dx.doi.org/10.1016/j.visinf.2021.12.005.

Su, K., Zhang, J., Xie, D., Tao, J., 2023. Importance guided stream surface generation and feature exploration. Vis. Informatics 7 (2), 54–63. http://dx.doi.org/10.1016/j.visinf.2023.05.002.

Sun, Y., Leigh, J., Johnson, A., Lee, S., 2010. Articulate: A semi-automated model for translating natural language queries into meaningful visualizations. In: Smart Graphics. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 184–195.

Tang, T., Wu, Y., Gao, J., Ruan, K., Zhang, Y., Ye, S., Wu, Y., Chen, X., 2024. ArtEyer: Enriching GPT-based agents with contextual data visualizations for fine art authentication. Vis. Informatics 8 (4), 48–59. http://dx.doi.org/10.1016/j.visinf.2024.11.001.

Tao, J., Wang, C., Chawla, N.V., Shi, L., Kim, S.H., 2018. Semantic flow graph: A framework for discovering object relationships in flow fields. IEEE Trans. Vis. Comput. Graphics 24 (12), 3200–3213. http://dx.doi.org/10.1109/TVCG.2017.2773071.

Tao, J., Wang, C., Shene, C.-K., Shaw, R.A., 2016. A vocabulary approach to partial streamline matching and exploratory flow visualization. IEEE Trans. Vis. Comput. Graphics 22 (5), 1503–1516. http://dx.doi.org/10.1109/TVCG.2015.2440252.

Theisel, H., Weinkauf, T., Hege, H.-C., Seidel, H.-P., 2003. Saddle connectors - an approach to visualizing the topological skeleton of complex 3D vector fields. In: IEEE Visualization, 2003. VIS 2003.. pp. 225–232. http://dx.doi.org/10.1109/VISUAL.2003.1250376.

Tong, C., Roberts, R., Borgo, R., Walton, S., Laramee, R.S., Wegba, K., Lu, A., Wang, Y., Qu, H., Luo, Q., et al., 2018. Storytelling and visualization: An extended survey. Information 9 (3), 65.

Tory, M., Setlur, V., 2019. Do what I mean, not what I say! design considerations for supporting intent and context in analytical conversation. In: 2019 IEEE Conference on Visual Analytics Science and Technology. VAST, pp. 93–103. http://dx.doi.org/10.1109/VAST47406.2019.8986918.

Wang, Z., Esturo, J.M., Seidel, H.-P., Weinkauf, T., 2017. Stream line–based pattern search in flows. In: Computer Graphics Forum. vol. 36, (8), Wiley Online Library, pp. 7–18.

Wang, Y., Hou, Z., Shen, L., Wu, T., Wang, J., Huang, H., Zhang, H., Zhang, D., 2023. Towards natural language-based visualization authoring. IEEE Trans. Vis. Comput. Graphics 29 (1), 1222–1232. http://dx.doi.org/10.1109/TVCG.2022.3209357.

Xu, L., Lee, T.-Y., Shen, H.-W., 2010. An information-theoretic framework for flow visualization. IEEE Trans. Vis. Comput. Graphics 16 (6), 1216–1224. http://dx.doi.org/10.1109/TVCG.2010.131.

Xu, L., Shen, H.-W., 2010. Flow web: a graph based user interface for 3D flow field exploration. In: Visualization and Data Analysis 2010. vol. 7530, SPIE, pp. 152–163.

Ye, Y., Hao, J., Hou, Y., Wang, Z., Xiao, S., Luo, Y., Zeng, W., 2024. Generative AI for visualization: State of the art and future directions. Vis. Informatics 8 (2), 43–66. http://dx.doi.org/10.1016/j.visinf.2024.04.003.

Ye, X., Kao, D., Pang, A., 2005. Strategy for seeding 3D streamlines. In: VIS 05. IEEE Visualization, 2005.. pp. 471–478. http://dx.doi.org/10.1109/VISUAL.2005.1532831.

Yu, B., Silva, C.T., 2020. FlowSense: A natural language interface for visual data exploration within a dataflow system. IEEE Trans. Vis. Comput. Graphics 26 (1), 1–11. http://dx.doi.org/10.1109/TVCG.2019.2934668.

Yuan, Z., He, S., Liu, Y., Yu, L., 2023. MEinVR: Multimodal interaction techniques in immersive exploration. Vis. Informatics 7 (3), 37–48. http://dx.doi.org/10.1016/j.visinf.2023.06.001.

Zafar, A., Yang, D., Chen, G., 2024. Extract and characterize hairpin vortices in turbulent flows. IEEE Trans. Vis. Comput. Graphics 30 (1), 716–726. http://dx.doi.org/10.1109/TVCG.2023.3326603.

Zhang, T., Ramakrishnan, R., Livny, M., 1996. BIRCH: an efficient data clustering method for very large databases. In: Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data. SIGMOD '96, Association for Computing Machinery, New York, NY, USA, pp. 103–114. http://dx.doi.org/10.1145/233269.233324.