NLI4VolVis: Natural Language Interaction for Volume Visualization via LLM Multi-Agents and Editable 3D Gaussian Splatting

Kuangshi Ai, Kaiyuan Tang, and Chaoli Wang

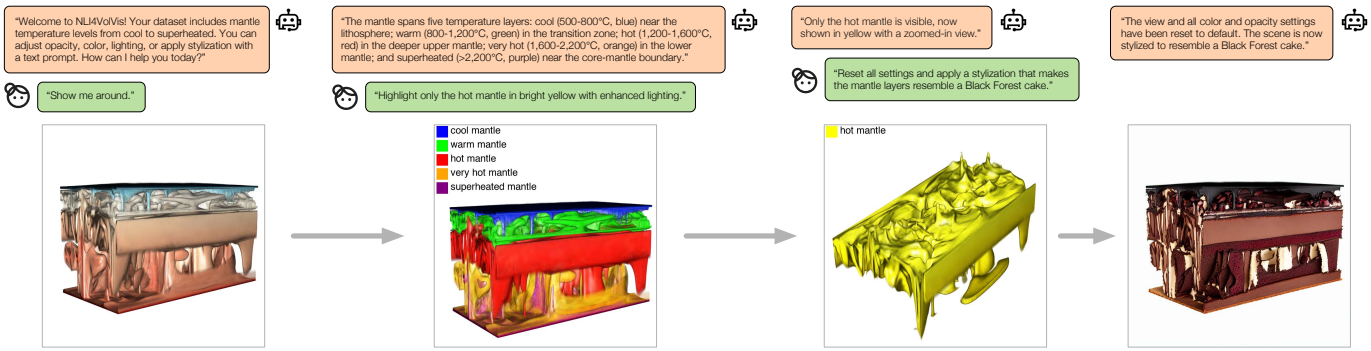

Fig. 1: An example of chatting with NLI4VolVis about the mantle temperature dataset. Our system lets users intuitively explore and edit volumetric datasets using natural language. Through open-vocabulary querying, real-time scene editing, and stylization, users can easily highlight semantic structures, adjust visualization properties, and apply creative styles.

Abstract—Traditional volume visualization (VolVis) methods, like direct volume rendering, suffer from rigid transfer function designs and high computational costs. Although novel view synthesis approaches enhance rendering efficiency, they require additional learning effort for non-experts and lack support for semantic-level interaction. To bridge this gap, we propose NLI4VolVis, an interactive system that enables users to explore, query, and edit volumetric scenes using natural language. NLI4VolVis integrates multi-view semantic segmentation and vision-language models to extract and understand semantic components in a scene. We introduce a multi-agent large language model architecture equipped with extensive function-calling tools to interpret user intents and execute visualization tasks. The agents leverage external tools and declarative VolVis commands to interact with the VolVis engine powered by 3D editable Gaussians, enabling open-vocabulary object querying, real-time scene editing, best-view selection, and 2D stylization. We validate our system through case studies and a user study, highlighting its improved accessibility and usability in volumetric data exploration. We strongly recommend readers check out our case studies, demo video, and source code at https://nli4volvis.github.io/.

Index Terms—Volume visualization, novel view synthesis, natural language interaction, open-vocabulary querying, editable 3D Gaussian splatting, multi-agent collaboration

1 Introduction

In *scientific visualization* (SciVis), *volume visualization* (VolVis) enables domain experts to explore and analyze complex 3D volumetric datasets. Traditionally, *direct volume rendering* (DVR) has been the dominant approach, valued for its ability to produce high-quality visualizations via ray marching through dense voxel grids. However, DVR suffers from significant limitations: it is computationally intensive, requires substantial storage, and scales poorly with resolution, often rendering real-time interaction impractical.

To overcome these challenges, researchers have turned to *novel view synthesis* (NVS) techniques that generate new views from sparsely sampled rendering images, eliminating the need for full volumetric integration. Among these techniques, *3D Gaussian splatting* (3DGS) [10] has emerged as a promising alternative. 3DGS directly rasterizes point-based Gaussian primitives onto the screen, offering rapid rendering and compact scene representations well suited to modern GPUs. This approach enables more efficient and interactive visualization pipelines and serves as a foundation for flexible volumetric scene editing.

• *The authors are with the Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556, USA. E-mail: {kai, ktang2, chaoli.wang}@nd.edu.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

Despite such rendering improvements, the creation of meaningful visualizations still hinges on the *transfer function* (TF), which maps voxel intensities to color and opacity. While many advanced TF techniques have been proposed [4, 5, 9, 32, 34, 39] to enhance expressiveness, TF design remains a complex and often unintuitive task. Conventional TFs, in particular, lack semantic awareness, making object-level exploration and manipulation challenging in intricate datasets.

Recent advances in vision-language models have introduced automatic, text-driven TF generation [27]. However, these methods still fall short of enabling interactive, real-time scene editing through natural language, underscoring a persistent disconnect between *natural language interaction* (NLI) and intuitive, semantic-level control in VolVis.

The rise of *large language models* (LLMs) presents a transformative opportunity to bridge this gap. With strong reasoning and planning capabilities, LLMs can act as intelligent agents—interpreting user intent, orchestrating multi-step tasks, and generating context-aware responses. When combined with vision-language models, they facilitate semantic comprehension of visual data, enabling open-vocabulary interaction and editing. Moreover, *multi-agent systems*—comprising collaborative LLMs—can support distributed task execution, enhancing system robustness and flexibility.

Yet, no existing system has unified editable volumetric representations, open-vocabulary scene understanding, and collaborative multi-agent LLMs to support intuitive, natural language-based VolVis. Prior efforts [27, 29] either lack object-level semantic granularity or fall short of enabling real-time interaction and flexible scene manipulation.

To address these gaps, we present NLI4VolVis (Natural Language

Interaction for Volume Visualization), a unified system that seamlessly integrates rendering, perception, and interaction. NLI4VolVis combines: (1) editable 3DGS [58] for real-time volumetric rendering, (2) vision-language models for semantic scene understanding, and (3) a multi-agent LLM-based framework to interpret user commands and manipulate volumetric scenes interactively. With NLI4VolVis, users across all levels of expertise can explore, query, and edit scenes using natural language. The system supports semantic-level operations such as adjusting color and opacity, modifying lighting, selecting optimal viewpoints, and applying 2D stylization effects. An example is illustrated in Figure 1.

At the core of NLI4VolVis is a robust multi-agent architecture where specialized agents collaboratively interpret user intents, generate declarative commands, and iteratively refine visualizations to achieve desired outcomes. Building on the *visualization-perception-action* (VPA) loop introduced by AVA [38], LLM agents iteratively refine visualizations by perceiving semantic details, evaluating progress toward user-defined goals, and taking subsequent actions until the target is achieved. The system also employs a flexible semantic segmentation pipeline, accommodating multiple segmentation strategies to ensure robust performance across diverse datasets. By integrating these components into an intuitive, interactive interface, NLI4VolVis provides a comprehensive solution for natural language-driven VolVis. The main contributions of NLI4VolVis are as follows:

- **Semantic-aware real-time rendering and editing for VolVis**: We leverage multi-view semantic segmentation to identify scene components and train editable 3DGS models, enabling efficient real-time rendering, editing, and manipulation of volumetric scenes through natural language querying.
- **Robust scene understanding pipeline**: We develop a scene understanding pipeline that combines vision-language embeddings with entropy-guided multi-view image selection, effectively aligning user queries with target volumetric structures.
- **Hierarchical multi-agent system powered by LLMs**: We design a multi-agent architecture where individual agents follow the VPA loop and collaborate in interpreting user intentions and executing the corresponding visualization tasks.
- **Comprehensive NLI interface**: We provide a real-time, interactive interface that integrates visualization and scene editing, voice-enabled communication, transparent agent action logs, and 2D stylization, ensuring an intuitive and seamless user experience.

2 Related Work

Traditional VolVis methods, such as DVR, offer precise results but are computationally intensive. They often require expert knowledge to fine-tune TFs and lack intuitive, semantic-level interaction. With the advent of LLMs and vision-language models, NLI has emerged as a promising approach for exploring and manipulating volumetric data more intuitively. Building on these advancements, our work integrates a multi-agent LLM framework with editable 3DGS. As summarized in Table 1, NLI4VolVis unifies five key capabilities: (1) NLI for visualization, (2) open-vocabulary querying, (3) real-time scene editing, (4) semantic scene understanding, and (5) user-controlled modifications. Together, these capabilities significantly improve the flexibility, accessibility, and usability of VolVis systems.

Natural language interaction for visualization. NLI for visualization has gained significant attention, enabling users to create and interact with visual representations of data using plain language. Incorporating well-structured dialogues and contextual understanding greatly improves user interaction in visualization systems [61]. An analysis of visualization-oriented NLI systems highlights the need to resolve query interpretation, data transformation, visual mapping, interaction, and presentation issues to enhance the overall user experience [53].

Researchers have designed various NLI-driven approaches for specific visualization domains to tackle these challenges. For example, FlowSense [73] introduces an NLI designed for dataflow visualization systems, enabling users to construct and modify dataflow diagrams using natural language. FlowNL [26] advances NLI for flow visualization by developing a declarative grammar that translates user queries into visualizations of flow structures. It simplifies complex interactions while supporting domain-specific tasks like flow structure derivation and visualization. Meanwhile, other approaches harness LLMs for automated visualization generation, employing in-context learning and iterative refinement to enhance the accuracy of translating natural language into visual representations [65]. Similarly, VOICE [29] integrates the conversational capabilities of LLMs with visualization, enabling real-time navigation and manipulation of 3D models through natural language commands. Another work, T2TF [27], focuses on volume rendering and employs a vision-language model to generate TFs that align rendered volumes with user-defined textual descriptions.

While these works in SciVis lay the foundation for natural language-driven visualization, they fall short when handling complex semantics in 3D volume data. In contrast, our NLI4VolVis extends the NLI paradigm to VolVis by integrating LLM agents with editable 3DGS. By incorporating open-vocabulary querying through a vision-language model, NLI4VolVis empowers users to interactively manipulate any object within a 3D scene using natural language. This includes adjusting color, opacity, and lighting, selecting the optimal view, and applying style transfer.

Table 1: Comparison of related works on NLI for visualization.

method	NLI for visualization	open-vocabulary querying	real-time scene editing	semantic scene understanding	user-controlled modification
FlowSense [73]	✓	✗	✗	✗	✗
FlowNL [26]	✓	✗	✗	✗	✓
VOICE [29]	✓	✗	✗	✗	✓
T2TF [27]	✗	✗	✗	✓	✓
iVR-GS [58]	✗	✗	✓	✗	✓
LangSplat [49]	✗	✓	✗	✓	✗
AVA [38]	✓	✗	✗	✓	✓
ChatVis [44]	✓	✗	✗	✗	✓
ParaView-MCP [37]	✓	✗	✓	✓	✓
NLI4VolVis (ours)	✓	✓	✓	✓	✓

Novel view synthesis. Recent NVS techniques have significantly enhanced 3D scene reconstruction from multi-view images. NeRF [45] pioneered this direction, providing high-fidelity geometric modeling but facing challenges in training and inference costs. Efficiency-focused methods like Instant-NGP [46] and Plenoxels [11] have since broadened NVS applicability. In DL4SciVis [62], NVS is categorized under visualization generation [2,20,22,24], distinct from data generation [13, 14, 18, 19, 56, 68] and neural compression [12, 40, 55, 67]. In VolVis, methods like StyleRF-VolVis [57] and ViSNeRF [69] have extended NVS to VolVis, with the former focusing on style transfer for volume rendering and the latter leveraging a multidimensional radiance field for dynamic scene exploration.

To overcome NeRF's limitations, 3DGS [30] introduces explicit Gaussian primitives for real-time rendering and efficient scene composition. Its application in cinematic anatomy [47] showcases photorealistic rendering of high-resolution datasets on consumer devices. iVR-GS [58] extends 3DGS to enable real-time editing of scene attributes such as color, opacity, and lighting. It inspires works such as TexGS-VolVis [54] that uses 2DGS for volume stylization and VolSegGS [71] that uses deformable 3DGS for dynamic VolVis scene segmentation and tracking. With its support for real-time rendering and intuitive editing, iVR-GS provides a strong foundation for NLI4VolVis, enabling NLI for interactive and explorable VolVis.

StyleGaussian [36] extends 3DGS to enable instant zero-shot 3D style transfer with strong multi-view consistency. However, as demonstrated in Section 3 of the appendix, applying StyleGaussian to SciVis data introduces noticeable artifacts and requires substantial training time for each scene. Moreover, like most 3D style transfer methods, StyleGaussian alters only the scene's color attributes while preserving the underlying geometry, limiting its expressiveness for more creative or structural editing tasks. In contrast, we adopt a 2D stylization approach using InstructPix2Pix (IP2P) [3], which supports flexible, prompt-driven creative editing of individual scene components. While this approach has known limitations in maintaining multi-view consistency, it offers greater stylistic versatility and immediate control. We plan to incorporate more advanced 3D stylization techniques in future iterations of the system to further enhance its capabilities.

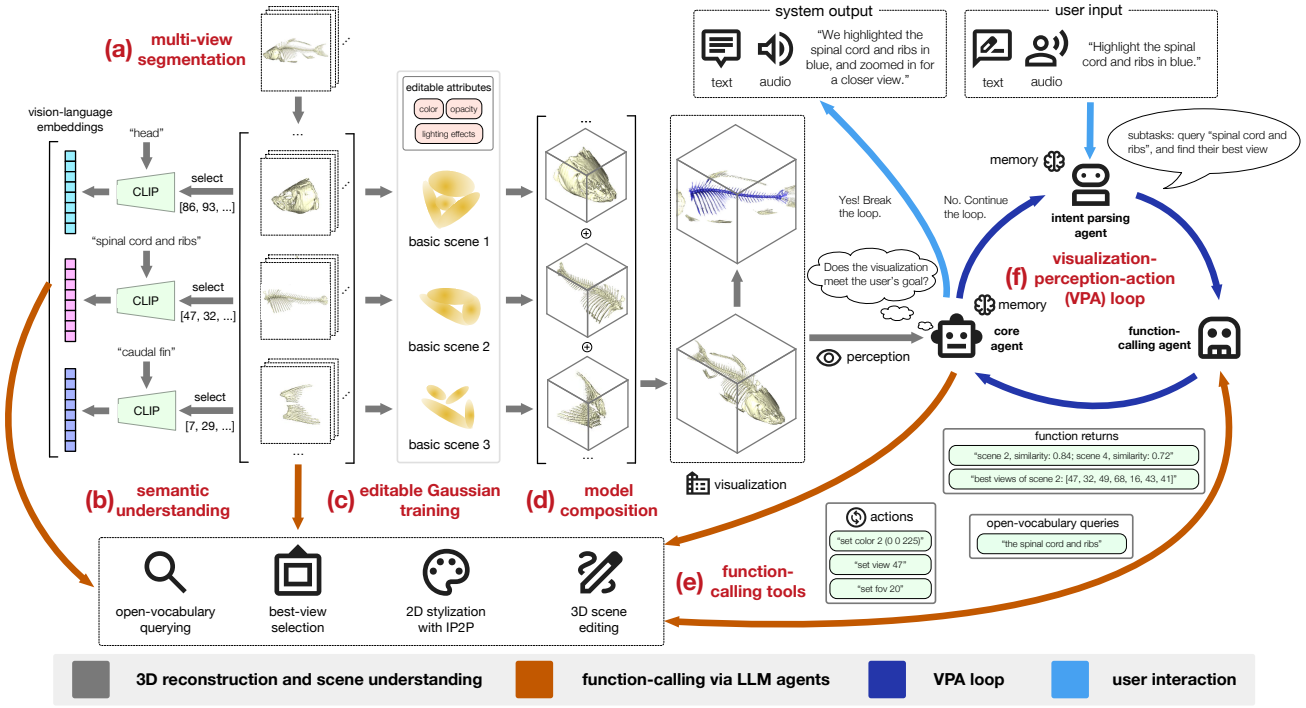

Fig. 2: The NLI4VolVis pipeline. (a) Multi-view segmentation supports cross-frame tracking and 3DGS-based techniques for datasets that are not previously segmented. (b) Each segmented component undergoes entropy-guided view selection, embedding informative frames and textual descriptions with CLIP for semantic understanding. (c) Editable iVR-GS models are trained per component. (d) Individual iVR-GS models are composed into the full volumetric scene. (e) Function-calling tools allow LLM agents to query, analyze, edit, and stylize the scene. (f) LLM agents iteratively refine visualizations via the VPA loop based on user input.

3D open-vocabulary segmentation. 3D open-vocabulary segmentation allows users to interact with any object in a 3D scene using natural language querying, making it a cornerstone of our work. Unlike closed-set methods [8, 75], which are limited by predefined categories, open-vocabulary approaches leverage 2D foundational vision-language models (e.g., CLIP [51] and SAM [33]) to integrate semantic understanding into 3D representations, enabling the recognition and manipulation of arbitrary objects. A common strategy is to map 2D semantic features back to 3D space. Examples like LangSplat [49] and LERF [31] embed language-encoded features from 2D multi-view images into 3D fields, enabling flexible querying. Advanced methods, such as Semantic Gaussians [15] and SAGD [25], improve segmentation accuracy by incorporating radiance and semantic information using 3DGS. OpenGaussian [64] and Gaussian Grouping [72] further this direction by enriching 3D Gaussian primitives with semantic encodings or identity embeddings, enabling point-level and instance-level segmentation and editing. These methods facilitate fine-grained 3D scene understanding and manipulation by extending 2D vision-language representations into 3D space. NLI4VolVis adopts a flexible pipeline that supports a range of open-vocabulary segmentation techniques (e.g., LangSplat [49], SAM 2 [52], and SAGD [25]), as well as pre-segmented datasets, establishing a robust framework for NLI in VolVis.

LLM-based autonomous agents. LLM-based autonomous agents have emerged as a powerful solution for human-like decision-making across diverse domains, driving significant advancements in their development, applications, and evaluation [16, 59, 63]. These agents are designed to dynamically plan, reason, and interact with their environment by leveraging LLMs' vast internal world knowledge. In visualization, such agents have been applied to automate data exploration and visualization workflows. For instance, AVA [38] combines natural language understanding with visual perception to iteratively refine visualizations based on user-defined objectives, such as adjusting TFs or hyperparameters in volume rendering. Similarly, ChatVis [44] automates SciVis processes by generating and iteratively refining Python scripts for tools like ParaView, enabling non-experts to create visualizations using natural language. ParaView-MCP [37] integrates LLM agents with direct API control in ParaView through the *model context protocol* (MCP), enabling seamless information exchange among users, LLM agents, and visualization tools to support human-AI collaborative visualization workflows. NLI4VolVis builds upon these advances by integrating LLM-based agents into the NLI pipeline to interpret user intent and execute VolVis scene manipulations. These agents facilitate interaction with 3D scenes through natural language, supporting open-vocabulary querying, interactive editing, and exploratory tasks. By harnessing LLMs' planning and reasoning capabilities, we make VolVis more accessible across various scenarios.

3 Our Approach

As illustrated in Figure 2, our NLI4VolVis framework offers a comprehensive pipeline that integrates multi-view semantic segmentation, 3D open-vocabulary scene understanding, editable Gaussian training, and NLI for VolVis. The pipeline begins with multi-view image segmentation of the volumetric scene using flexible semantic segmentation strategies (Section 3.1). Each segmented object is then processed through an entropy-guided view selection mechanism, where the most informative frames are embedded using the CLIP model to achieve robust open-vocabulary scene understanding (Section 3.2). This ensures that user queries are accurately aligned with the corresponding volumetric structures. Next, we train editable NVS models for each segmented component as a basic scene using iVR-GS [58], enabling independent editing and seamless composition of the complete scene. The resulting models support real-time editing of color, opacity, and lighting, as well as composability through the concatenation of Gaussian parameters (Section 3.3). We introduce a hierarchical multi-agent system powered by LLMs to enable natural language-driven interaction. LLM agents utilize external function-calling tools to interpret and modify 3D scenes, enabling open-vocabulary querying, optimal view selection, interactive scene editing, and 2D stylization. These collaborative agents follow the VPA loop [38], allowing them to interpret user intents, generate declarative commands, and iteratively refine visualizations (Section 3.4). Finally, our interactive user interface (Sec-

tion 3.5) integrates all components of the pipeline, enabling NLI with volume datasets through features such as real-time visualization editing, transparent agent action logs, speech input and audio output, and 2D stylization via IP2P [3].

When a new volumetric dataset is introduced, we begin by rendering multi-view images using a preset TF in ParaView. These images are then processed by a selected multi-view semantic segmentation method. Each segmented semantic component is used to train an independent, editable 3DGS model. The final scene is assembled by combining these component models, enabling real-time editing, open-vocabulary querying, and stylization within NLI4VolVis.

3.1 Multi-View Semantic Segmentation

To enable open-vocabulary querying and interactive scene editing in VolVis, our NLI4VolVis pipeline must support efficient semantic segmentation across various VolVis scenarios. Since an NVS model can generate new viewpoints from multi-view images, a natural approach is to train editable 3DGS models using segmented multi-view images. In this setup, each segmented object corresponds to an independent iVR-GS model [58], ensuring that individual objects can be queried and edited separately.

The first step in this pipeline is multi-view semantic segmentation. Given the diversity of VolVis datasets, a one-size-fits-all segmentation method is impractical. Therefore, we adopt a flexible approach, allowing users to select the segmentation method that best suits their data. Below, we summarize commonly used semantic segmentation techniques in VolVis, discussing their strengths and limitations.

Pre-segmented datasets. Some volumetric datasets, such as medical imaging benchmarks (e.g., FLARE human organ [42]), come with ground-truth semantic labels. These datasets are ideal for controlled environments but do not generalize well to arbitrary VolVis scenes.

TF-based segmentation. Traditional TFs in DVR map voxel values to color and opacity. However, they are not well-suited for intuitive, object-level segmentation, particularly for non-expert users. Even multi-dimensional TFs often fail to segment objects based on semantic meaning [39], as they primarily rely on voxel intensity and derived attributes rather than learned object semantics.

Voxel-based segmentation. Voxel-based techniques extract features using voxel coordinates, scalar values, gradients, and neighborhood information [48]. Approaches like voxel2vec [23] apply representation learning to uncover semantic structures within a volumetric dataset. While voxel-based methods effectively cluster local regions, they often lack semantic awareness and struggle with generalization to complex, real-world 3D scenes. Additionally, since NVS models operate on multi-view images rather than raw voxels, directly applying voxel-based segmentation is not well-suited to our pipeline.

Cross-frame object tracking. A straightforward approach to segmenting multi-view images is to apply 2D foundation models, such as SAM [33], to each rendered viewpoint independently. However, this framewise segmentation often results in view inconsistencies, where an object's mask varies between neighboring frames due to occlusions, lighting changes, or segmentation artifacts. To address this, we adopt SAM 2 [52], a video object segmentation and tracking model that improves segmentation consistency across viewpoints. By sampling along a continuous circular camera trajectory, we treat multi-view images as a video sequence, enabling objects to be tracked across frames and reducing inconsistencies caused by independent framewise segmentation. However, SAM 2 struggles with transparent and nested objects, which are common in VolVis.

Direct segmentation in 3DGS. These methods perform semantic segmentation directly on NVS models by leveraging 2D foundation models to integrate semantic features into 3D Gaussian representations. Since segmentation is applied directly to the 3D Gaussians, segmented renderings can be generated from any viewpoint, ensuring view-consistent object querying. However, this approach modifies the underlying Gaussian structure, reducing flexibility and making it less compatible with other segmentation methods. To ensure scalability and interoperability, NLI4VolVis does not directly segment iVR-Gaussians. Instead, it trains iVR-GS models using multi-view renderings of objects segmented by LangSplat [49] and SAGD [25]. While this method maintains a method-agnostic segmentation approach, it may result in some loss of visualization quality.

In summary, each segmentation method has its own strengths and trade-offs. Given the diverse nature of VolVis scenes, no single method is optimal for all use cases. By leveraging segmented multi-view images as input to editable 3DGS models, NLI4VolVis adopts a flexible strategy that allows users to choose the segmentation method best suited to their specific datasets and visualization goals.

3.2 3D Open-Vocabulary Scene Understanding via CLIP

To facilitate natural language-driven VolVis scene editing, NLI4VolVis integrates 3D open-vocabulary scene understanding using the CLIP model [51]. This integration enables the identification of objects within a 3D scene based on user queries expressed in natural language.

Although CLIP is inherently a 2D model, we aim to generate embeddings for 3D objects. Since different viewpoints of the same 3D object reveal varying information, selecting representative views is crucial. We employ an entropy-based method [28] to choose the top-k frames from evenly sampled multi-view images of a segmented object. Each selected frame is processed to obtain its CLIP embedding, and the mean of these embeddings serves as the final representation.

To identify the most informative viewpoints, we calculate the entropy of each image based on its opacity (alpha) channel. Using opacity alone allows us to assess the structural complexity within the volume data, as it is less affected by lighting and shading effects compared to color. Including color in the calculation may introduce variability, leading to a less consistent representation of the underlying structure. Thus, we focus solely on opacity to ensure a more reliable measure. Images with higher entropy contain more information and are, therefore, more informative. The entropy H of an image I is computed as $H(I) = -\sum_{i=1}^{N} p_i \log p_i$, where p_i denotes the probability (normalized opacity value) at pixel i, and N is the total number of pixels. After computing the entropy for all frames, we select the top-k frames with the highest entropy values.

For each of the selected top-k frames, we generate a CLIP embedding. The vision-based embedding E_{vision} for the 3D object is obtained by averaging these individual embeddings $E_{\text{vision}} = (\sum_{j=1}^{k} E_{\text{frame}_j})/k$, where E_{frame_j} denotes the CLIP embedding of the j-th selected frame. For datasets without additional textual information, the object embedding is defined solely by the vision-based embedding, i.e., $E_{\text{object}} = E_{\text{vision}}$. When textual labels or descriptions are available, we also utilize CLIP to embed the corresponding textual description. The final object embedding is then computed as $E_{\text{object}} = (E_{\text{vision}} + E_{\text{text}})/2$,where E_{text} is the text-based embedding. This integration of vision and textual embeddings creates a unified, multimodal representation for each semantic component. The unified embedding enables more accurate and efficient open-vocabulary querying by capturing visual and semantic cues in a single vector space.

Once the CLIP embeddings are established for each segmented object, our system can interpret user queries expressed in natural language. Upon receiving a query, we generate its CLIP embedding and compute $S(E_{\text{query}}, E_{\text{object}_i})$, the cosine similarity between this query embedding E_{query} and the embedding of each object E_{object_i}

$$S(E_{\text{query}}, E_{\text{object}_i}) = \frac{E_{\text{query}} \cdot E_{\text{object}_i}}{\|E_{\text{query}}\| \cdot \|E_{\text{object}_i}\|}. \tag{1}$$

The objects corresponding to the highest similarity scores are identified as the target of the user's query, enabling efficient scene editing based on natural language input.

3.3 Editable Gaussian Training

Existing NVS methods for VolVis [57, 58, 69, 70] cannot semantically locate and edit objects. One promising solution is to apply semantic segmentation techniques (Section 3.1) to partition VolVis scene components, representing each as a set of multi-view images. These segmented multi-view images are then used as input to train an NVS model for each individual component. The *editability* of the NVS model ensures that each reconstructed 3D component remains editable.

Ideally, the NVS method should also support *composability*, meaning several independently trained models, referred to as basic scenes, can be combined by simply concatenating their corresponding parameters. This feature allows users to view individual semantic components separately while still providing a complete reconstruction of the original VolVis scene.

We select iVR-GS [58] as the NVS method for our work. iVR-GS builds upon 3DGS [30], an explicit 3D scene representation technique. Each 3D Gaussian primitive $G(\mathbf{x})$ is characterized by its spatial mean $\mu \in \mathbb{R}^{3\times1}$ and covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$

$$G(\mathbf{x}) = \exp(-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)), \tag{2}$$

where $\mathbf{x} \in \mathbb{R}^{3\times1}$ represents a spatial coordinate. Each original 3D Gaussian also has two additional attributes: an opacity value $o \in [0,1]$ and a view-dependent color $\mathbf{c} \in \mathbb{R}^{3\times1}$. For differentiable optimization, the covariance matrix Σ is further parameterized by a scaling vector $\mathbf{s} \in \mathbb{R}^{3\times1}$ and a rotation vector $\mathbf{q} \in \mathbb{R}^{4\times1}$. 3DGS uses a set of 3D Gaussians to reconstruct a 3D scene, and the i-th Gaussian primitive is parameterized with $\{\mu_i, \mathbf{s}_i, \mathbf{q}_i, o_i, \mathbf{c}_i\}$.

3D Gaussians can be projected onto the image plane through rasterization when rendering from a novel view. With the viewing transformation matrix $\mathbf{W}$ and the Jacobian matrix $\mathbf{J}$ of the projective transformation, the projected 2D covariance matrix Σ' is computed as $\Sigma' = \mathbf{JW}\Sigma\mathbf{W}^\top\mathbf{J}^\top$. For each pixel, its color $\mathbf{C}$ is determined by the sum of N sequentially layered 2D Gaussians alone the ray

$$\mathbf{C} = \sum_{i\in N} \mathbf{c}_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{3}$$

where α is the alpha value computed by multiplying the Gaussian weight with the opacity o learned per-primitive. iVR-GS extends the original 3DGS by augmenting primitives with extra attributes and optimizing them to formulate editable Gaussian primitives.

The training of an iVR-GS model starts from optimizing an original 3DGS model with an extra normal attribute $\mathbf{n}$ for each Gaussian. This normal is used to calculate the view-dependent color $\mathbf{c}$ for each Gaussian based on the Blinn-Phong reflection model. To enable color and lighting edits, several extra attributes are assigned to each Gaussian in iVR-GS: an offset color $\Delta\mathbf{c} \in \mathbb{R}^{3\times1}$, ambient, diffuse, and specular coefficients $\{k_a, k_d, k_s\}$, and a shininess term β. Specifically, the voxel color in the Blinn-Phong reflection model is expressed as $\mathbf{c}_p + \Delta\mathbf{c}$, where the palette color $\mathbf{c}_p \in \mathbb{R}^{3\times1}$ is frozen during training as the mean color value of all multi-view training images and is shared by all Gaussians within a basic scene.

In addition to the loss functions used in 3DGS, iVR-GS incorporates two key regularization strategies. First, L1 sparsity regularization is applied to the offset color $\Delta\mathbf{c}$ to prevent excessive color shifts. Second, bilateral smoothness regularization is applied to the shading attributes $\{k_a, k_d, k_s, \beta\}$, ensuring smooth shading transitions in uniform-color regions. After training, the scene is represented by a set of editable Gaussians, where the i-th primitive is defined by shading attributes $\{k_{ai}, k_{di}, k_{si}, \Delta\mathbf{c}_i, \beta_i\}$ and geometry attributes $\{\mu_i, \mathbf{s}_i, \mathbf{q}_i, o_i\}$.

During inference, these editable Gaussians in iVR-GS allow for the editing of color, opacity, light magnitude, and light direction. In particular, we train one basic iVR-GS model for each segmented component in NLI4VolVis. To edit the color of each component, we replace its $\mathbf{c}_p$ with the user-desired color $\mathbf{c}'_p$. The opacity of the component is adjusted by scaling the opacity term o. The coefficients k_a, k_d, k_s and β adjust the lighting magnitude accordingly. Finally, by using the normal $\mathbf{n}$ of each Gaussian primitive, changes to lighting direction are achieved following the Blinn-Phong equations.

In contrast to 3DGS, iVR-GS also supports composability: combining multiple scenes by simply concatenating Gaussian parameters. This composability lets users view each semantic component individually or reconstruct the complete VolVis scene in NLI4VolVis.

3.4 NLI for VolVis

In the NLI4VolVis framework, we combine editable 3DGS for scene representation with various semantic segmentation techniques to extract and manipulate semantic components within VolVis. The NLI component utilizes the extensive world knowledge and visual understanding of multimodal LLMs to interpret user intentions and execute corresponding visualization tasks using a suite of function-calling tools. We also develop a set of declarative command rules that define visualization operations—such as adjusting opacity, color, and lighting—for each semantic component in the scene. To further enhance system capabilities, we adopt a collaborative multi-agent architecture.

Multi-agent collaboration. Recent advances in LLMs have enabled the development of agentic AI, where multiple LLM-based agents can perceive, reason, and collaborate to solve complex tasks in real-world applications. These multi-agent systems consist of intelligent agents that communicate, collaborate, and sometimes compete to address tasks at scale. The relationships among agents can be cooperative, competitive, or a combination of both, and collaboration strategies can be role-based, rule-based, or model-based [16, 59]. In NLI4VolVis, we implement a rule-based, collaborative multi-agent system with a hierarchical communication structure. This system includes agents responsible for tasks such as parsing, open-vocabulary querying, user interaction through dialogue, and declarative command generation. To efficiently manage explicit VolVis tasks, NLI4VolVis employs a hierarchical communication structure where agents are assigned distinct roles across multiple layers. These agents collaborate iteratively, ensuring that the core agent can determine when the user's intent has been fulfilled or when the maximum number of iterations has been reached. This hierarchical structure effectively distributes tasks across different levels, enhancing the robustness and resilience of the reasoning process.

Each agent in the system can observe and perceive its environment, reason based on its observations, and then take action. Current LLM agents typically produce output in the form of *text*, *audio*, or *images*. Their ability to perceive and manipulate the environment can be significantly enhanced using external *function-calling tools* [17, 50]. In NLI4VolVis, each agent has its own memory and a task-driven planning core defined by prompts that describe specific tasks. Agents can also access multiple function-calling tools to achieve visualization goals.

Moreover, agents in NLI4VolVis must comprehend the outcomes of their visualizations and adapt their actions accordingly. To support this, we incorporate the VPA loop introduced by AVA [38]. This loop enables the core agent to manipulate semantic components within VolVis by sending declarative commands to the visualization server. It also leverages the visual perception capability to understand the current visualization, assess whether the user-defined goal has been achieved, and reason about which declarative commands to issue in the next iteration, if necessary. For each user input, the agents will follow this loop iteratively until the core agent determines that the user-defined visualization target has been met.

Function-calling via LLM agents. In NLI4VolVis, LLM agents leverage external function-calling tools to perceive, analyze, and manipulate visualizations in response to user intent. These tools offer structured workflows that extend the capabilities of LLMs, allowing them to execute complex tasks beyond their internal reasoning. The key function-calling capabilities in NLI4VolVis include

- **Open-vocabulary object querying**: Computes CLIP similarity (see Section 3.2) between the user's textual description and precomputed embeddings of segmented components, enabling agents to accurately identify the objects being referred to.
- **Scene editing**: Allows modifications to each segmented component's color, opacity, and lighting parameters, as well as global adjustments to the VolVis scene.
- **Knowledge-based question answering**: Extracts dataset-specific metadata to assist agents in understanding the semantic context of the visualization, enhancing their ability to respond to domain-specific user queries.
- **Best-view selection**: Retrieves the top-k frames with the highest entropy values for each semantic component, allowing agents to adjust the camera view for optimal presentation. Here, entropy is computed over the alpha channel of the rendered frames; higher entropy indicates richer structural complexity, which helps in selecting more informative viewpoints.

- **2D stylization**: Interfaces with IP2P [3], a text-based image diffusion model, enabling agents to apply user-defined styles to specific objects or the entire scene.

By incorporating these function-calling tools, NLI4VolVis ensures that LLM agents can precisely interpret user intentions and execute the corresponding visualization tasks.

Declarative VolVis commands. The iVR-GS engine in NLI4VolVis operates as a server, receiving declarative commands from LLM agents to dynamically modify the visualization. These commands allow precise control over various scene properties and ensure the visualization aligns with the user's intent. Basic visualization settings include adjusting each segmented object's color, opacity, and lighting parameters. Camera controls allow LLM agents to set the viewing direction and field of view, enabling optimal perspectives of specific scene components. Additional rendering operations, such as setting the rendering mode, changing the background color, saving images, and resetting views, provide further flexibility. Moreover, LLM agents utilize declarative commands to apply 2D text-based stylization through IP2P, enabling artistic transformations on selected objects or the entire scene. In essence, these agents control every visualization aspect by executing structured commands, ensuring precise modifications that align with user objectives.

3.5 Interactive Interface

The NLI4VolVis interface is designed to facilitate seamless interaction with volumetric data through natural language inputs and direct visualization manipulation. As shown in Figure 3, the interface consists of four main components, arranged from left to right: the control panel, rendering window, chat widget, and action log.

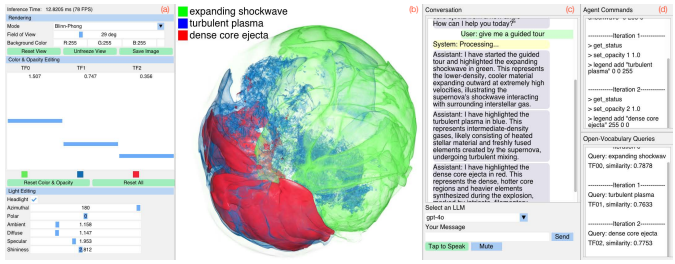

Fig. 3: The NLI4VolVis interface consists of four key components: (a) panel, (b) rendering window, (c) chat widget, and (d) action log.

The control panel allows users to adjust color, opacity, lighting, and viewing angles to highlight specific structures and refine views. The rendering window updates in response to user inputs, reflecting changes made through declarative commands. The chat widget enables NLI with LLM agents. Users can submit queries via text or speech input and receive responses alongside an updated visualization. The widget supports multiple LLMs, including GPT-4o, DeepSeek-V3, and Llama3.2-90B-Vision, allowing users to choose AI models based on their preferences and task requirements. The action log records all executed commands and open-vocabulary queries, offering transparency into the multi-agent decision-making process. This feature helps users understand the reasoning behind visualization changes and refine their queries for more precise results. In the "Open-Vocabulary Query" section of the action log, "TF" refers to the transfer function, representing a semantic component segmented using TFs in the 3D scene. "Similarity" indicates the cosine similarity between the CLIP embedding of the user query and that of each semantic component, enabling open-vocabulary object identification. To further enhance accessibility, NLI4VolVis supports voice-based input and audio responses. Based on user study feedback, we have improved the voice interaction experience and added multilingual support for both input and output. Users can also review past interactions in the action log, allowing for iterative refinement of queries to achieve more accurate visualization outcomes. For a more detailed demonstration, we refer readers to the supplementary video, which showcases real-time interactions with the system.

4 Results and Evaluation

We evaluate NLI4VolVis across eight diverse volumetric datasets to demonstrate its effectiveness and generalizability. As detailed in Table 2, five datasets were obtained through scanning, encompassing everyday objects, animals, and organs, while the remaining three were generated from scientific simulations. For each dataset, we report the optimal multi-view semantic segmentation method in Table 2 and offer clearer guidance on selecting the appropriate method based on volumetric data characteristics in Section 1 of the appendix. Volume sizes range from 50 MB to 4.3 GB, yet 3DGS ensures low-latency inference, consistently achieving over 100 *frames per second* (FPS) on an NVIDIA RTX 4090 GPU, regardless of volume size. In contrast, DVR with ParaView faces a substantial performance bottleneck, achieving only 5.09 FPS when rendering the largest 4.32 GB dataset, chameleon. 3DGS outperforms DVR by up to 20×.

Table 2: Datasets and their respective settings.

dataset	volume resolution	volume size	# semantic components	segmentation method
backpack	512×512×373	373 MB	5	TF+SAGD
carp	256×256×512	64 MB	7	SAM 2
chameleon	1024×1024×1080	4.32 GB	4	TF+SAGD
FLARE human organ	512×512×330	330 MB	14	pre-segmentation
kingsnake	1024×1024×795	3.18 GB	2	TF
hurricane	500×500×100	95 MB	3	TF
mantle temperature	360×201×180	49.69 MB	5	TF
supernova	432×432×432	323 MB	3	TF

For Gaussian splatting, we set the image resolution to 800×800 for both training and inference. The training begins with generating 92 multi-view images using icosphere sampling in ParaView with the NVIDIA IndeX plugin [1]. After multi-view semantic segmentation, we train a separate iVR-GS [58] model for each semantic component. The training follows a two-stage process: an initial 30,000 iterations for base 3DGS training, followed by 10,000 iterations for editable Gaussians. All other parameters remain consistent with the original iVR-GS. The average training time for each basic scene is approximately seven minutes. To highlight the capabilities of NLI4VolVis, we present several illustrative case studies and conduct a formal user study to assess interaction performance. Refer to the supplemental video for demonstrations of real-time interactions.

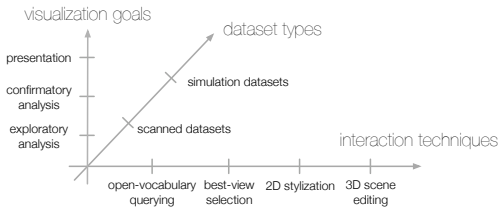

Fig. 4: The three independent axes serve as the foundation for designing our NLI4VolVis case studies.

4.1 Case Studies

Data visualization serves three primary goals: *exploratory analysis*, *confirmatory analysis*, and *presentation* [35]. Exploratory analysis enables open-ended interaction with data, allowing users to uncover patterns and generate hypotheses without predefined assumptions. Confirmatory analysis involves testing specific hypotheses through structured examination, validating or refuting initial insights. Presentation focuses on effectively communicating findings by selecting appropriate visual techniques to convey pre-established facts. NLI4VolVis guides users in achieving these three goals when working with scanned or simulation datasets, leveraging a suite of viewing and editing tools, including open-vocabulary querying, best-view selection, 2D stylization, and 3D scene editing. As illustrated in Figure 4, each case study in NLI4VolVis aligns with a subset of these goals along the three independent axes.

Carp. The carp dataset comprises a CT scan of a carp fish, segmented into seven semantic components using SAM 2. Each component is processed into a basic scene for further analysis.

To facilitate exploratory analysis, we design a guided tour that introduces users to the dataset by highlighting each semantic component in distinct colors while providing natural language and audio explanations. This approach familiarizes users with the dataset, laying the groundwork for deeper analysis. As shown in Figure 5 (a), NLI4VolVis navigates through the segmented components step by step in response to a simple prompt: "*Give me a guided tour.*"

Following the guided tour, users can proceed with confirmatory analysis by testing specific hypotheses. NLI4VolVis supports flexible scene editing, best-view selection, and natural language explanations to help users verify their assumptions. For instance, as illustrated in Figure 5 (b), users can request the system to show only the fins, highlight the pectoral fin in red, adjust the view for better visibility, and increase brightness. NLI4VolVis interprets these requests, identifies the referenced object using open-vocabulary querying, and executes declarative visualization commands while providing explanatory feedback.

For presentation-focused visualization, NLI4VolVis offers extensive 3D scene editing and text-driven 2D stylization to generate artistic representations of the dataset. Since 3D stylization methods [6, 21, 60] require significant computational time for fine-tuning NVS models, we support real-time 2D stylization instead, enabling immediate visual enhancements. Figure 5 (c) shows such a result of the carp dataset generated with the prompt: "*Transform the entire fish into a cyborg.*"

A user study was conducted using this dataset to evaluate the effectiveness of NLI4VolVis in data exploration and interaction efficiency. See Section 4.2 and the supplementary materials for detailed results.

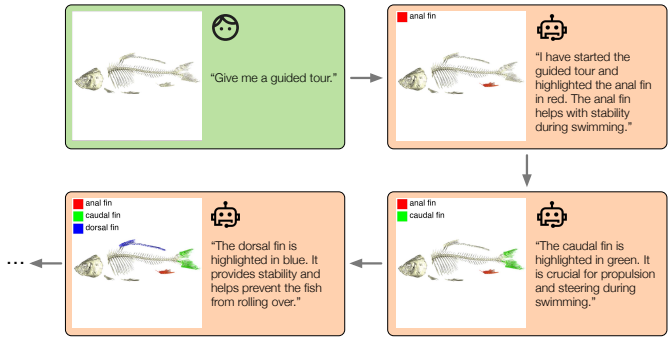

Fig. 5: Case study with the carp dataset demonstrating the three visualization goals.

FLARE human organ. The FLARE human organ [42] dataset is a labeled abdominal organ dataset from the MICCAI 2022 Grand Challenge, designed for medical imaging and educational purposes. Featuring pre-segmented organs, this dataset is well-suited for non-expert users, such as junior medical school students or the general public, to gain a better understanding of human anatomy.

NLI4VolVis enhances data accessibility and comprehension by enabling semantic filtering and interactive visualization. As shown in Figure 6, users can request NLI4VolVis to display only organs related to the digestive system, supporting both exploratory analysis and presentation goals. Using the open-vocabulary query tool, our multi-agent system first identifies relevant anatomical structures—including the liver, pancreas, gallbladder, esophagus, stomach, and duodenum. It then retrieves the corresponding segmented components and applies declarative commands to generate the requested view.

Supernova. The supernova dataset is a scientific simulation that captures the dynamic processes of a stellar explosion. We segment

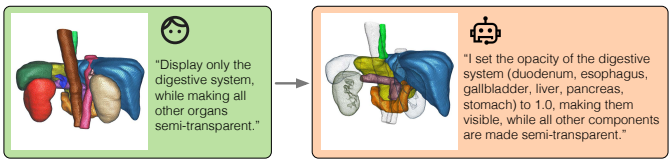

Fig. 6: Case study with the FLARE human organ dataset demonstrating semantic filtering via NLI.

it using a TF into three key components: the expanding shockwave, turbulent plasma, and dense core ejecta.

NLI4VolVis leverages its multi-modal capabilities and the extensive internal knowledge of LLMs to provide users with an interactive and intuitive understanding of complex scientific phenomena. As shown in Figure 7, users request to explain the green region in the visualization. Our system first identifies this region as the expanding shockwave using its perception capabilities. It then adds a legend to the requested region and generates a contextual explanation of the expanding shockwave, enhancing user comprehension of the dataset. Users further request an adjustment to the lighting direction. NLI4VolVis successfully shifts the lighting from the lower right to the upper left by iteratively applying its perception capability through the VPA loop.

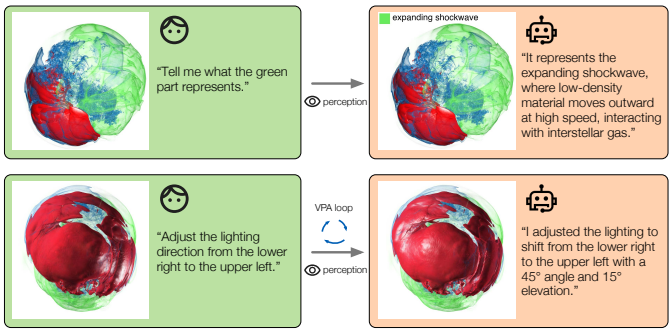

Fig. 7: Case study with the supernova dataset demonstrating the perception capabilities of NLI4VolVis.

Backpack. The backpack dataset is a CT scan of a backpack containing various everyday objects, such as water bottles and ropes. Visualizing such datasets has practical applications, particularly in security screening at airports and train stations.

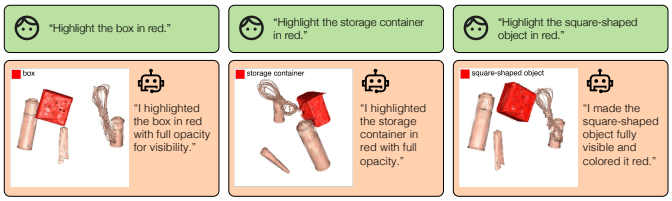

Fig. 8: Case study with the backpack dataset demonstrating how open-vocabulary querying handles vague natural language references.

To process the dataset, we first apply a TF to isolate objects within the backpack before segmenting the entire scene into five semantic components. As shown in Figure 8, users interact with the system by referring to the same object using different descriptions, such as "*the box*," "*the storage container*," or "*the square-shaped object*." Despite these variations, our multi-agent system accurately interprets user intent and employs the open-vocabulary query tool to correctly identify the referenced object, demonstrating its robustness in handling vague or context-dependent descriptions.

Other case studies. We evaluate NLI4VolVis across all eight datasets listed in Table 2, demonstrating its adaptability to diverse data types. Figure 9 presents representative user interactions, highlighting various system functionalities applied to different datasets. These interactions include colormap adjustments for visualizing temperature variations, text-based 2D stylization for adapting to environmental contexts, text-driven TF design, view selection, and result saving. Each

case illustrates how users leverage natural language queries to refine visualizations and extract meaningful insights.

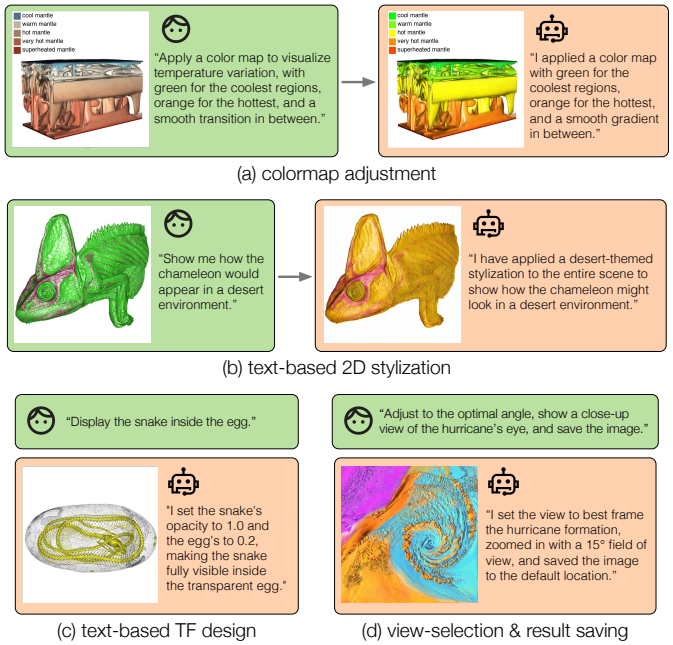

Fig. 9: Representative user interactions across different datasets.

4.2 User Study

Our user study evaluates how effectively NLI4VolVis responds to user queries, the effort required for non-experts to familiarize themselves with the system, its impact on users' understanding of volumetric datasets, and how well users can communicate with the system to achieve their visualization goals.

Participants and experiment setup. Eight participants were recruited for the study following the University's IRB protocol. The group included three individuals with master's degrees in mathematics or computer engineering and five with bachelor's degrees. Among them, six had experience in computer science-related fields, including machine knitting, human-computer interaction, LLM systems, medical image processing, and graph learning. The remaining two specialized in biomolecular engineering and chemistry. Three participants reported having prior domain knowledge of the datasets due to their academic background; however, none had previously explored these specific datasets using visualization tools before the study. The remaining five participants had no prior knowledge of the datasets. However, only two were highly familiar with natural language-driven visualization systems. Each participant interacted with the system on a workstation equipped with an NVIDIA RTX 4090 GPU and a 32-inch display with a resolution of 2560×1440.

Procedure and tasks. The study followed a four-phase workflow: *introduction*, *exploration*, *task execution*, and *post-questionnaire*. Participants were first introduced to NLI4VolVis through a ten-minute tutorial and a demonstration video using the backpack dataset to illustrate the system's core functionalities. They then explored the interface freely with the backpack and chameleon datasets, with the opportunity to ask questions. Exploration continued until they felt confident in using the system.

In the task execution phase, participants worked with three datasets: carp (biological dataset), mantle temperature (simulation dataset), and FLARE human organ (medical dataset). The study included 23 tasks covering various visualization and interaction challenges: guided dataset exploration (3 tasks), answering domain-specific knowledge questions (9 tasks), modifying visualization properties (e.g., color, opacity, lighting) (4 tasks), optimizing views (1 task), applying 2D stylization (1 task), and semantic filtering (e.g., identifying the largest organ) (5 tasks). Finally, to evaluate the system's natural language processing capabilities, participants who initially completed tasks using only the *graphical user interface* (GUI) were asked to replicate those tasks exclusively through NLI.

The study concluded with a post-questionnaire, where participants rated usability, efficiency, and interaction quality using a five-point Likert scale. Open-ended feedback was collected to identify system strengths, challenges, and areas for improvement. All eight participants completed all four phases, and each was compensated $30. Their interactions were recorded for subsequent analysis, with performance measured based on task completion time, the number of queries required, and response accuracy. Additionally, we recorded the time each participant spent exploring the system and the total time taken to complete all tasks.

Table 3: User task performance across seven metrics.

metric	P1	P2	P3	P4	P5	P6	P7	P8	mean
exploration time (min)	8	24	5	8	4	5	16	9	9.9
task completion time (min)	34	32	42	32	34	32	40	35	35.1
NL interactions	35	26	27	16	20	20	28	23	24.4
effective NL interactions	28	21	21	15	18	19	24	19	20.6
non-NL interactions	8	0	1	9	4	2	2	1	3.4
NL usage rate (%)	81.4	100.0	96.4	64.0	83.3	90.9	93.3	95.8	88.1
knowledge QA accuracy	8/9	9/9	9/9	9/9	9/9	9/9	8/9	9/9	8.75/9

Task performance. Table 3 summarizes user task performance from the study, with P1 to P8 representing the eight participants. The reported metrics include: "exploration time"—the duration each participant spent familiarizing themselves with the system before starting the tasks, "task completion time"—the total time taken to complete all assigned tasks, "NL interactions"—the number of interactions using natural language, "non-NL interactions"—the number of interactions using other methods, such as the GUI, "effective NL interactions"—the number of natural language queries that successfully contributed to task completion, "NL usage rate"—the proportion of interactions conducted via natural language, and "knowledge QA accuracy"—the number of correctly answered domain-specific questions (out of nine).

All participants completed the tasks, although their interaction patterns and completion times varied. Exploration time ranged from 4 to 24 minutes, averaging 9.9 minutes. Some participants, such as P2 and P7, spent more time exploring the system. However, exploration time did not correlate strongly with task completion time, which averaged 35.1 minutes, with individual times ranging from 32 to 42 minutes.

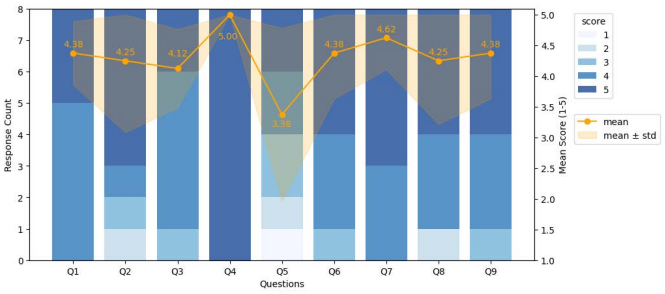

Fig. 10: Survey responses to nine post-questionnaire questions, along with mean scores (on a 1–5 scale) and standard deviations.

Participants demonstrated differing preferences between NLIs and traditional GUI controls. For example, P4, who was highly familiar with interactive visualization systems and had prior experience in computer graphics, relied on NL for only 64% of interactions, which was notably lower than the average NL usage rate of 88.1%. In contrast, two participants with no computer science background, P2 and P8, used NL almost exclusively, with usage rates of 100% and 95.8%, respectively.

All participants answered domain-specific knowledge questions well, achieving an average accuracy of 8.75/9. The number of interactions required to complete the tasks varied across participants. On average, participants performed 24.4 NL interactions, with 20.6 being effective, and made 3.4 non-NL interactions while completing 23 tasks. Since 14 tasks required at least one interaction (excluding knowledge QA tasks), some participants made additional interactions beyond the minimum required. Analyzing the recorded interactions revealed that this behavior was driven by two main factors: some participants lacked patience and

initiated new interactions before the system responded, while others deliberately explored the system's capabilities.

Survey responses. We assessed the accessibility and usability of NLI4VolVis through a post-questionnaire. After completing the tasks, each participant completed a survey that included their background information, ratings of the system on a 1–5 scale across various aspects, a description of one highlight of the system, notes on challenges encountered during the study, and suggestions for improvements. The survey results are presented in Figure 10.

Overall, participants gave positive feedback, with an average score of 4.38/5 for Q1 ("*The system allows me to efficiently interact with and manipulate volumetric datasets*"). Three questions focused on accessibility: Q2 ("*I can easily complete visualization tasks without assistance*," 4.25/5), Q4 ("*The system is easy to learn, even for first-time users*," 5.0/5), and Q8 ("*The system's interface is intuitive and user-friendly*," 4.38/5). Most participants found NLI4VolVis highly accessible, especially compared to traditional VolVis tools like ParaView. Four participants (P1, P3, P4, and P5) mentioned the system's ease of learning as its most valuable feature.

The survey also assessed usability, with participants rating all aspects 4.1 or higher, except for Q5 ("*I find the system's voice input and output features useful for interaction*," 3.38/5). While four participants rated the voice features 4 or 5, two rated them 1 or 2. Observations from the user study revealed that voice input usage varied significantly: P1 and P8 primarily used voice commands, while others preferred keyboard input. Additionally, some participants found the voice output distracting and chose to mute the system. This suggests that while voice interaction is convenient for some users, it may not be universally preferred.

User behaviors. Survey responses indicate that NLI significantly simplifies complex visualization tasks, making them more intuitive. One participant noted, "*Most existing visualization tools are difficult for beginners, but this system transforms complex adjustments into natural language, making it very easy to use.*" This was particularly evident in lighting manipulation, which traditionally requires adjusting Blinn-Phong parameters—selecting between headlight or orbital lighting, modifying azimuthal and polar angles, and fine-tuning ambient, diffuse, specular, and shininess coefficients. These adjustments require technical knowledge, as increasing the first three parameters strengthens lighting while increasing the last one reduces brightness. Before the user study, none of the participants fully understood these settings. However, all were able to achieve their desired lighting effects simply by stating commands such as "*make the scene brighter*" or "*change the lighting direction to the upper left*."

When asked about the most useful aspects of the system, participants frequently highlighted the intuitive natural language interface, as well as features such as color and lighting adjustments, the guided tour, and scene segmentation. Some also appreciated the convenience of voice input, which reduced the need for manual adjustments.

Despite its strengths, latency emerged as the most frequently mentioned challenge, with several participants reporting slow response times and uncertainty about whether the system was still processing their input. Others noted occasional misinterpretation of commands, interface aesthetics, and a need for additional documentation for first-time users. One found that the legend was sometimes inaccurate, while another emphasized the need for clearer explanations of traditional rendering techniques for users unfamiliar with computer graphics.

For improvements, participants suggested refining zoom and rotation functions, adding a "Reset All" button, and enhancing real-time system feedback. Several users requested status indicators to confirm when the system was actively processing their commands. One specific suggestion was to display system messages such as "Processing..." in the terminal window during execution. Participants also suggested adding hover-based tooltips for quick access to information and enhancing visual cues, such as highlighting selected objects and toolbars. These insights provide valuable feedback on the strengths and areas for improvement in NLI4VolVis, guiding future enhancements to improve efficiency, usability, and system responsiveness.

System improvement. In response to participant feedback, we have made several enhancements to NLI4VolVis. To address latency issues, we optimized the system by implementing a deque-based memory management strategy for the LLM agents, enabling customization of memory length to balance performance and responsiveness. Nevertheless, advanced memory pruning techniques, such as summarization-based approaches [43, 66] or retrieval-based hierarchical methods [74], could further improve efficiency and long-term coherence, which we plan to explore in future work. Additionally, we have introduced a "Reset All" button for easier scene reinitialization and added real-time processing indicators to inform users when the system is handling commands.

5 Limitations and Future Work

Extension of multi-agent system. NLI4VolVis is designed specifically for *cooperative* interactions among agents without incorporating *competitive* dynamics. Additionally, our collaboration strategy follows a *rule-based* approach, where predefined rules dictate agent communication. While this ensures coordinated actions, it limits adaptability in uncertain scenarios. We plan to explore *role-based* and *model-based* collaboration strategies to enhance flexibility. Regarding agent communication, we employ a *hierarchical* structure, which is well-suited for question-answering and reasoning tasks. However, this design introduces challenges such as increased latency and potential system failure if a higher-level node becomes unresponsive. To mitigate these issues, future research could explore *centralized* or *decentralized* communication models to improve system robustness and reduce latency.

Lack of validation for function-calling tools. NLI4VolVis exhibits strong capabilities in semantic understanding and manipulation of 3D scenes through external function-calling tools. However, we observe that the system tends to be overconfident in these tools' outputs and lacks mechanisms for validating their correctness. For instance, when a user submits a query unrelated to the current scene, the open-vocabulary querying module still computes the CLIP similarity between the query and all segmented components and returns the top match to the LLM agents. For *visionless* LLMs (e.g., DeepSeek-V3), there is no means to assess whether the returned result is contextually valid, often leading to inappropriate or misleading visualizations. In contrast, *vision-enabled* models (e.g., GPT-4o) can detect such mismatches and issue warnings or clarification prompts. In future work, incorporating automatic validation techniques within the LLM agent framework will further enhance system robustness and reliability.

Creation of benchmark for VolVis. While our user study incorporated carefully designed VolVis tasks, the field still lacks a standardized, large-scale benchmark for quantitatively evaluating LLM agents in SciVis, particularly in volumetric data analysis. In contrast, information visualization benchmarks [7, 41] have begun integrating natural language queries with ground-truth visualizations and automated evaluations, some even introducing ambiguity and multiple valid outputs to assess LLM reasoning under uncertainty. In future work, we aim to develop a VolVis-specific benchmark featuring diverse datasets, natural language queries, reference visualizations, and automated evaluation of visualization accuracy and interaction quality. Such a benchmark would enhance reproducibility and enable more rigorous assessments of NLI-driven VolVis systems like NLI4VolVis. In addition, extending the application of NLI4VolVis to a wider range of real-world volumetric datasets beyond our current case studies presents an exciting avenue.

6 Conclusions

We have introduced NLI4VolVis, an NLI framework for VolVis. NLI4VolVis integrates multi-view segmentation, vision-language models, and editable 3DGS to extract, interpret, and reconstruct semantic components in volumetric scenes. It enables users to explore, query, and manipulate these scenes through an LLM-based multi-agent system with robust function-calling capabilities. We evaluated NLI4VolVis through case studies across eight datasets, complemented by a formal user study. The results show that participants quickly learned the system, engaged naturally through language, and accurately completed assigned tasks. By making complex VolVis tasks more intuitive and enhancing accessibility to volumetric data exploration, NLI4VolVis has the potential to democratize VolVis, making it more approachable for both experts and non-experts alike.

Acknowledgments

This research was supported in part by the U.S. National Science Foundation through grants IIS-1955395, IIS-2101696, OAC-2104158, and IIS-2401144, and the U.S. Department of Energy through grant DE-SC0023145. The authors would like to thank the anonymous reviewers for their insightful comments.

Appendix

1 Comparison of Segmentation Methods

To evaluate the impact of 3D multi-view semantic segmentation methods, we compared LangSplat [49], SAGD [25], and SAM 2 [52] using the backpack dataset. As illustrated in Figure 1, each method generates semantic components used to train independent editable 3DGS models, which are subsequently composed into a complete volumetric scene. LangSplat preserves lighting and surface details effectively but suffers from lower segmentation accuracy, occasionally merging distinct components. SAGD achieves the most accurate semantic separation, enabling fine-grained, object-level editing; however, it introduces surface blurring due to its dense Gaussian decomposition. SAM 2 provides a middle ground between segmentation accuracy and visual fidelity, offering moderate semantic precision and better temporal consistency than LangSplat, though its reconstructions tend to appear flatter with diminished lighting and texture quality. Given these trade-offs, we selected SAGD for the backpack dataset to take advantage of its precise object segmentation.

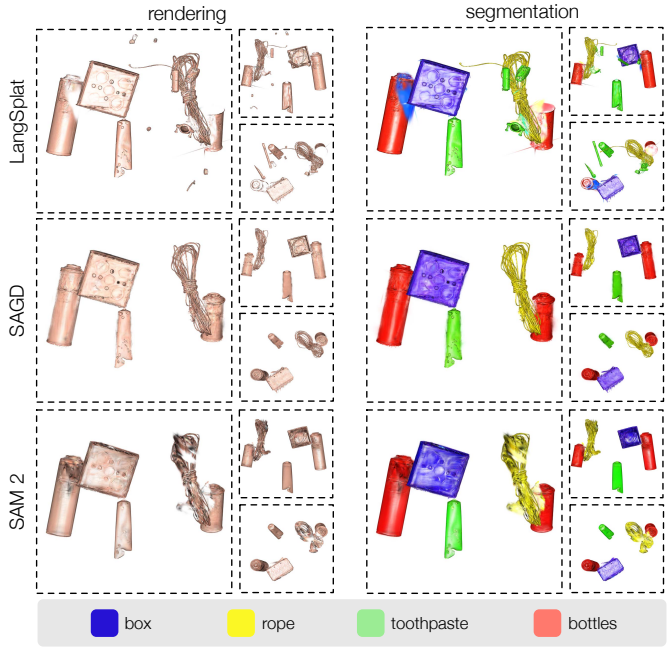

Fig. 1: Comparison of LangSplat, SAGD, and SAM 2 on the backpack dataset. Each row shows the renderings of the reconstructed scene (left) alongside the corresponding semantic segmentation (right).

We offer these guidelines for choosing a segmentation method:

- **LangSplat** is best suited for applications that prioritize high visual fidelity and surface detail, where perfect semantic segmentation is not critical.
- **SAGD** is ideal for scenarios requiring accurate object-level segmentation to enable precise querying and editing, especially in scenes with well-separated components.
- **SAM 2** is a balanced choice when both segmentation accuracy and visual quality are important, particularly for complex scenes where exact surface reconstruction is less essential.

Since no single method is universally optimal, users should select the approach that best aligns with their data characteristics and visualization goals. The modular architecture of NLI4VolVis supports seamless switching between segmentation strategies without requiring changes to the overall pipeline.

2 Comparison of LLMs

To accommodate diverse user preferences and performance requirements, NLI4VolVis supports integration with multiple multimodal LLMs via API. Latency among these models varies significantly depending on the provider. To quantitatively assess system responsiveness, we measured the *time to first token* (TTFT) for four GPT models, varying the number of user messages sent to NLI4VolVis. All tests were conducted using the OpenAI API under identical hardware and network conditions. For each model and message count combination, we repeated the experiment five times to calculate the mean and variance.

As shown in Figure 2 and Table 1, TTFT increases with the number of messages, and the variance in latency also grows, leading to less consistent response times. In our user study, most participants completed their tasks with fewer than ten messages, keeping per-response latency under 7.5 seconds in most cases. Based on these findings, we set the maximum message count in NLI4VolVis to ten, balancing latency and memory efficiency. Future enhancements could incorporate memory optimization techniques such as summarization-based methods [43, 66] or retrieval-based hierarchical strategies [74] to further improve performance.

Among the GPT models, GPT-4o offers the best overall trade-off between latency, output quality, and cost. For other model families (e.g., LLaMA, DeepSeek), we either lacked sufficient local deployment resources or found the available APIs failed to meet low-latency requirements; therefore, TTFT results for these models are not reported.

During the user study, participants freely experimented with GPT-4o, DeepSeek-V3, and LLaMA3.2-90B-Vision. GPT-4o consistently delivered superior performance in instruction following and language grounding. DeepSeek-V3 exhibited high latency—often exceeding 15 seconds for the first token—even with fewer than five messages, largely due to API limitations. LLaMA3.2-90B-Vision struggled with intent interpretation for complex queries, and local deployment was not feasible; its available APIs also suffered from high latency. Nonetheless, NLI4VolVis maintains flexibility: users can easily switch LLMs via API configurations to balance output quality and system responsiveness.

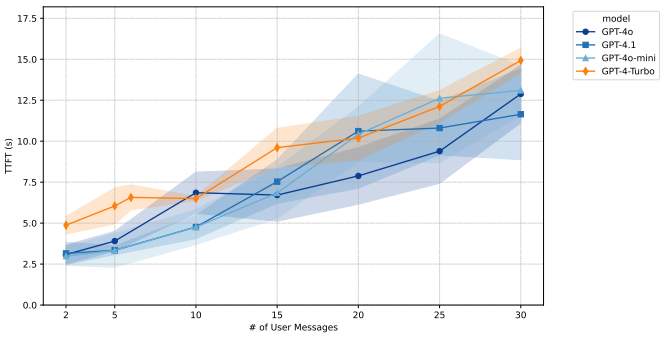

Fig. 2: Average TTFT for NLI4VolVis plotted against the number of user messages across different GPT models. Shaded regions represent standard deviations.

Table 1: Average TTFT (in seconds) across varying numbers of user messages for each GPT model.

model	# user messages						
	2	5	10	15	20	25	30
GPT-4o	3.08 ± 0.58	3.90 ± 0.59	6.86 ± 1.25	6.71 ± 1.60	7.88 ± 1.73	9.39 ± 1.96	12.89 ± 1.76
GPT-4.1	3.15 ± 0.68	3.34 ± 0.27	4.76 ± 0.71	7.53 ± 1.32	10.62 ± 3.49	10.80 ± 1.64	11.64 ± 2.78
GPT-4o-mini	2.98 ± 0.57	3.34 ± 1.03	4.76 ± 1.06	6.84 ± 1.55	10.42 ± 1.66	12.62 ± 3.92	13.10 ± 1.50
GPT-4-Turbo	4.87 ± 0.53	6.05 ± 1.10	6.50 ± 0.12	9.61 ± 1.18	10.19 ± 1.34	12.12 ± 0.97	14.93 ± 0.76

3 Comparison of 2D and 3D Stylization Methods

We compared 2D stylization using IP2P [3] and 3D stylization using StyleGaussian [36] on the mantle and chameleon datasets. Because StyleGaussian does not support text-driven prompts, we first applied IP2P to stylize a single view using textual input, then used the resulting image as a style reference to train StyleGaussian.

Training StyleGaussian on WikiArt took over three hours on an NVIDIA RTX 4090 GPU. Inference performance in our tests fell below

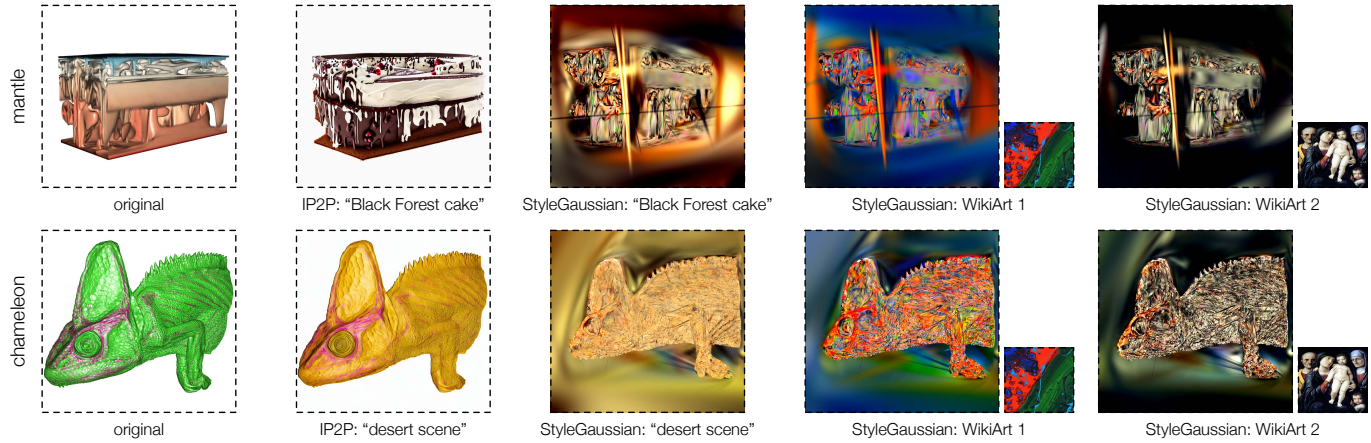

Fig. 3: 2D stylization (IP2P) and 3D stylization (StyleGaussian) results on the mantle and chameleon datasets. The second and third columns show stylizations generated from text prompts, while the last two columns display StyleGaussian's style transfer results using reference images from the WikiArt dataset, shown in the lower-right corners.

the reported 10 FPS; we note that the original paper used an RTX A5000 GPU, which outperforms typical consumer-grade hardware. As shown in Figure 3, StyleGaussian introduces noticeable artifacts when applied to SciVis data, such as the mantle example. Moreover, like most zero-shot radiance field style transfer methods, it only alters color while preserving geometry, limiting its creative flexibility. In contrast, IP2P supports prompt-driven stylization but lacks multi-view consistency. It completes each stylization in about ten seconds.

To mitigate view inconsistencies, our system temporarily freezes the camera view during IP2P stylization and reminds users to unfreeze it afterward if they wish to resume interactive exploration. Considering these tradeoffs—high training cost, limited interactivity, visual artifacts on SciVis data, and the lack of text-driven control—we chose not to incorporate StyleGaussian or similar 3D style transfer methods in NLI4VolVis. We plan to investigate more advanced and consistent stylization solutions in future work.

4 Top-k Frames for View Selection

We conducted an ablation study to assess the impact of the parameter k in our entropy-based view selection strategy. In this approach, the top-k frames with the highest entropy values are selected to generate CLIP embeddings for each segmented semantic component, enabling the 3D open-vocabulary understanding described in Section 3.2 of the paper.

The study was performed on the backpack and carp datasets, which contain five and seven semantic components, respectively. For each component, we calculated the similarity between its aggregated top-k CLIP embedding and the embedding of its ground-truth label. We report the mean similarity and variance across all components for each dataset. As shown in Table 2, the system exhibits low sensitivity to the choice of k when $k \geq 3$. However, using only the top-1 frame (k=1) slightly decreases robustness and semantic accuracy, likely due to limited visual context. Conversely, increasing k leads to longer preprocessing times without noticeable improvements in embedding quality. Based on these findings, we recommend setting k between 5 and 10 to strike a balance between semantic performance and computational efficiency.

Table 2: Similarity between CLIP embedding and ground-truth label across different top-k frames for entropy-based view selection.

	top-k frames				
dataset	k=1	k=3	k=5	k=10	k=15
backpack	0.748 ± 0.00031	0.763 ± 0.00036	0.766 ± 0.00034	0.764 ± 0.00037	0.763 ± 0.00036
carp	0.737 ± 0.00031	0.739 ± 0.00026	0.741 ± 0.00026	0.740 ± 0.00026	0.742 ± 0.00028

References

[1] ParaView NVIDIA IndeX plugin. https://gitlab.kitware.com/paraview/paraview. Accessed: 2025-03-13. 6

[2] M. Berger, J. Li, and J. A. Levine. A generative model for volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 25(4):1636–1650, 2019. doi: 10.1109/TVCG.2018.2816059 2

[3] T. Brooks, A. Holynski, and A. A. Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, pp. 18392–18402, 2023. doi: 10.1109/CVPR52729.2023.01764 2, 4, 6, 10

[4] S. Bruckner and M. E. Gröller. Style transfer functions for illustrative volume rendering. *Computer Graphics Forum*, 26(3):715–724, 2007. doi: 10.1111/J.1467-8659.2007.01095.X 1

[5] J. J. Caban and P. Rheingans. Texture-based transfer functions for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1364–1371, 2008. doi: 10.1109/TVCG.2008.169 1

[6] M. Chen, I. Laina, and A. Vedaldi. DGE: Direct Gaussian 3D editing by consistent multi-view editing. In *Proceedings of European Conference on Computer Vision*, pp. 74–92, 2024. doi: 10.1007/978-3-031-72904-1_5 7

[7] N. Chen, Y. Zhang, J. Xu, K. Ren, and Y. Yang. VisEval: A benchmark for data visualization in the era of large language models. *IEEE Transactions on Visualization and Computer Graphics*, 31(1):1301–1311, 2025. doi: 10.1109/TVCG.2024.3456320 9

[8] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1290–1299, 2022. doi: 10.1109/CVPR52688.2022.00135 3

[9] C. D. Correa and K.-L. Ma. Size-based transfer functions: A new volume exploration technique. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1380–1387, 2008. doi: 10.1109/TVCG.2008.162 1

[10] B. Fei, J. Xu, R. Zhang, Q. Zhou, W. Yang, and Y. He. 3D Gaussian splatting as new era: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 2024. Accepted. doi: 10.1109/TVCG.2024.3397828 1

[11] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5491–5500, 2022. doi: 10.1109/CVPR52688.2022.00542 2

[12] P. Gu, D. Z. Chen, and C. Wang. NeRVI: Compressive neural representation of visualization images for communicating volume visualization results. *Computers & Graphics*, 116:216–227, 2023. doi: 10.1016/J.CAG.2023.08.024 2

[13] P. Gu, J. Han, D. Z. Chen, and C. Wang. Reconstructing unsteady flow data from representative streamlines via diffusion and deep learning based denoising. *IEEE Computer Graphics and Applications*, 41(6):111–121, 2021. doi: 10.1109/MCG.2021.3089627 2

[14] P. Gu, J. Han, D. Z. Chen, and C. Wang. Scalar2Vec: Translating scalar fields to vector fields via deep learning. In *Proceedings of IEEE Pacific Visualization Symposium*, pp. 31–40, 2022. doi: 10.1109/PACIFICVIS53943.2022.00012 2

[15] J. Guo, X. Ma, Y. Fan, H. Liu, and Q. Li. Semantic Gaussians: Open-vocabulary scene understanding with 3D Gaussian splatting. *arXiv preprint arXiv:2403.15624*, 2024. doi: 10.48550/arXiv.2403.15624 3

[16] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, and X. Zhang. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024. doi: 10.

48550/arXiv.2402.01680 3, 5

[17] Z. Guo, S. Cheng, H. Wang, S. Liang, Y. Qin, P. Li, Z. Liu, M. Sun, and Y. Liu. StableToolBench: Towards stable large-scale benchmarking on tool learning of large language models. In *Proceedings of ACL Conference (Findings)*, pp. 11143–11156, 2024. doi: 10.18653/v1/2024.findings-acl.664 5

[18] J. Han and C. Wang. TSR-VFD: Generating temporal super-resolution for unsteady vector field data. *Computers & Graphics*, 103:168–179, 2022. doi: 10.1016/J.CAG.2022.02.001 2

[19] J. Han and C. Wang. VCNet: A generative model for volume completion. *Visual Informatics*, 6(2):62–73, 2022. doi: 10.1016/J.VISINF.2022.04.004 2

[20] J. Han and C. Wang. CoordNet: Data generation and visualization generation for time-varying volumes via a coordinate-based neural network. *IEEE Transactions on Visualization and Computer Graphics*, 29(12):4951–4963, 2023. doi: 10.1109/TVCG.2022.3197203 2

[21] A. Haque, M. Tancik, A. A. Efros, A. Holynski, and A. Kanazawa. Instruct-NeRF2NeRF: Editing 3D scenes with instructions. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, pp. 19683–19693, 2023. doi: 10.1109/ICCV51070.2023.01808 7

[22] W. He, J. Wang, H. Guo, K.-C. Wang, H.-W. Shen, M. Raj, Y. S. G. Nashed, and T. Peterka. InSituNet: Deep image synthesis for parameter space exploration of ensemble simulations. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):23–33, 2020. doi: 10.1109/TVCG.2019.2934312 2

[23] X. He, Y. Tao, S. Yang, H. Dai, and H. Lin. voxel2vec: A natural language processing approach to learning distributed representations for scientific data. *IEEE Transactions on Visualization and Computer Graphics*, 29(10):4296–4311, 2023. doi: 10.1109/TVCG.2022.3189094 4

[24] F. Hong, C. Liu, and X. Yuan. DNN-VolVis: Interactive volume visualization supported by deep neural network. In *Proceedings of IEEE Pacific Visualization Symposium*, pp. 282–291, 2019. doi: 10.1109/PacificVis.2019.00041 2

[25] X. Hu, Y. Wang, L. Fan, J. Fan, J. Peng, Z. Lei, Q. Li, and Z. Zhang. SAGD: Boundary-enhanced segment anything in 3D Gaussian via Gaussian decomposition. *arXiv preprint arXiv:2401.17857*, 2025. doi: 10.48550/arXiv.2401.17857 3, 4, 10

[26] J. Huang, Y. Xi, J. Hu, and J. Tao. FlowNL: Asking the flow data in natural languages. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1200–1210, 2023. doi: 10.1109/TVCG.2022.3209453 2

[27] S. Jeong, J. Li, C. R. Johnson, S. Liu, and M. Berger. Text-based transfer function design for semantic volume rendering. In *Proceedings of IEEE VIS Conference (Short Papers)*, pp. 196–200, 2024. doi: 10.1109/VIS55277.2024.00047 1, 2

[28] G. Ji and H.-W. Shen. Dynamic view selection for time-varying volumes. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1109–1116, 2006. doi: 10.1109/TVCG.2006.137 4

[29] D. Jia, A. Irger, O. Strnad, J. Björklund, A. Ynnerman, and I. Viola. VOICE: Visual oracle for interaction, conversation, and explanation. *arXiv preprint arXiv:2304.04083*, 2023. doi: 10.48550/arXiv.2304.04083 1, 2

[30] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139:1–139:14, 2023. doi: 10.1145/3592433 2, 5

[31] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. LERF: Language embedded radiance fields. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, pp. 19729–19739, 2023. doi: 10.1109/ICCV51070.2023.01807 3

[32] G. L. Kindlmann, R. T. Whitaker, T. Tasdizen, and T. Möller. Curvature-based transfer functions for direct volume rendering: Methods and applications. In *Proceedings of IEEE Visualization Conference*, pp. 513–520, 2003. doi: 10.1109/VISUAL.2003.1250414 1

[33] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023. doi: 10.1109/ICCV51070.2023.00371 3, 4

[34] J. M. Kniss, G. L. Kindlmann, and C. D. Hansen. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Proceedings of IEEE Visualization Conference*, pp. 255–262, 2001. doi: 10.1109/VISUAL.2001.964519 1

[35] H. Lam, M. Tory, and T. Munzner. Bridging from goals to tasks with design study analysis reports. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):435–445, 2018. doi: 10.1109/TVCG.2017.2744319 6

[36] K. Liu, F. Zhan, M. Xu, C. Theobalt, L. Shao, and S. Lu. StyleGaussian: Instant 3D style transfer with Gaussian splatting. *arXiv preprint arXiv:2403.07807*, 2024. doi: 10.48550/arXiv.2403.07807 2, 10

[37] S. Liu, H. Miao, and P.-T. Bremer. ParaView-MCP: An autonomous visualization agent with direct tool use. *arXiv preprint arXiv:2505.07064*, 2025. doi: 10.48550/arXiv.2505.07064 2, 3

[38] S. Liu, H. Miao, Z. Li, M. L. Olson, V. Pascucci, and P.-T. Bremer. AVA: Towards autonomous visualization agents through visual perception-driven decision-making. *Computer Graphics Forum*, 43(3):e15093, 2024. doi: 10.1111/cgf.15093 2, 3, 5

[39] P. Ljung, J. Krüger, E. Groller, M. Hadwiger, C. D. Hansen, and A. Ynnerman. State of the art in transfer functions for direct volume rendering. *Computer Graphics Forum*, 35(3):669–691, 2016. doi: 10.1111/cgf.12934 1, 4

[40] Y. Lu, P. Gu, and C. Wang. FCNR: Fast compressive neural representation of visualization images. In *Proceedings of IEEE VIS Conference (Short Papers)*, pp. 31–35, 2024. doi: 10.1109/VIS55277.2024.00014 2

[41] T. Luo, C. Huang, L. Shen, B. Li, S. Shen, W. Zeng, N. Tang, and Y. Luo. nvBench 2.0: A benchmark for natural language to visualization under ambiguity. *arXiv preprint arXiv:2503.12880*, 2025. doi: 10.48550/arXiv.2503.12880 9

[42] J. Ma, Y. Zhang, S. Gu, C. Ge, S. Ma, A. Young, C. Zhu, X. Yang, K. Meng, Z. Huang, et al. Unleashing the strengths of unlabelled data in deep learning-assisted pan-cancer abdominal organ quantification: The FLARE22 challenge. *The Lancet Digital Health*, 6(11):e815–e826, 2024. doi: 10.1016/S2589-7500(24)00154-7 4, 7

[43] A. Maharana, D.-H. Lee, S. Tulyakov, M. Bansal, F. Barbieri, and Y. Fang. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 13851–13870, 2024. doi: 10.18653/v1/2024.acl-long.747 9, 10

[44] T. Mallick, O. Yildiz, D. Lenz, and T. Peterka. ChatVis: Automating scientific visualization with a large language model. In *Proceedings of ACM/IEEE SC Workshops*, pp. 49–55, 2024. doi: 10.1109/SCW63240.2024.00014 2, 3

[45] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of European Conference on Computer Vision*, pp. 405–421, 2020. doi: 10.1007/978-3-030-58452-8_24 2

[46] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4):102:1–102:15, 2022. doi: 10.1145/3528223.3530127 2

[47] S. Niedermayr, C. Neuhauser, K. Petkov, K. Engel, and R. Westermann. Application of 3D Gaussian splatting for cinematic anatomy on consumer class devices. *arXiv preprint arXiv:2404.11285*, 2024. doi: 10.48550/arXiv.2404.11285 2

[48] F. Poux and R. Billen. Voxel-based 3D point cloud semantic segmentation: Unsupervised geometric and relationship featuring vs deep learning methods. *ISPRS International Journal of Geo-Information*, 8(5):213:1–213:34, 2019. doi: 10.3390/ijgi8050213 4

[49] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister. LangSplat: 3D language Gaussian splatting. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20051–20060, 2024. doi: 10.1109/CVPR52733.2024.01895 2, 3, 4, 10

[50] Y. Qin, S. Hu, Y. Lin, W. Chen, N. Ding, G. Cui, Z. Zeng, X. Zhou, Y. Huang, C. Xiao, et al. Tool learning with foundation models. *ACM Computing Surveys*, 57(4):101:1–101:40, 2024. doi: 10.1145/3704435 5

[51] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of International Conference on Machine Learning*, pp. 8748–8763, 2021. 3, 4

[52] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, et al. SAM 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. doi: 10.48550/arXiv.2408.00714 3, 4, 10

[53] L. Shen, E. Shen, Y. Luo, X. Yang, X. Hu, X. Zhang, Z. Tai, and J. Wang. Towards natural language interfaces for data visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 29(6):3121–3144, 2023. doi: 10.1109/tvcg.2022.3148007 2

[54] K. Tang, K. Ai, J. Han, and C. Wang. TexGS-VolVis: Expressive scene editing for volume visualization via textured Gaussian splatting. *IEEE Transactions on Visualization and Computer Graphics*, 32(1), 2026. Ac-

cepted. 2

[55] K. Tang and C. Wang. ECNR: Efficient compressive neural representation of time-varying volumetric datasets. In *Proceedings of IEEE Pacific Visualization Conference*, pp. 72–81, 2024. doi: 10.1109/PACIFICVIS60374.2024.00017 2

[56] K. Tang and C. Wang. STSR-INR: Spatiotemporal super-resolution for time-varying multivariate volumetric data via implicit neural representation. *Computers & Graphics*, 119:103874, 2024. doi: 10.1016/J.CAG.2024.01.001 2

[57] K. Tang and C. Wang. StyleRF-VolVis: Style transfer of neural radiance fields for expressive volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 31(1):613–623, 2025. doi: 10.1109/TVCG.2024.3456342 2, 4

[58] K. Tang, S. Yao, and C. Wang. iVR-GS: Inverse volume rendering for explorable visualization via editable 3D Gaussian splatting. *IEEE Transactions on Visualization and Computer Graphics*, 31(6):3783–3795, 2025. doi: 10.1109/TVCG.2025.3567121 2, 3, 4, 5, 6

[59] K.-T. Tran, D. Dao, M.-D. Nguyen, Q.-V. Pham, B. O'Sullivan, and H. D. Nguyen. Multi-agent collaboration mechanisms: A survey of LLMs. *arXiv preprint arXiv:2501.06322*, 2025. doi: 10.48550/arXiv.2501.06322 3, 5

[60] C. Vachha and A. Haque. Instruct-GS2GS: Editing 3D Gaussian splats with instructions. https://instruct-gs2gs.github.io/, 2024. 7

[61] H. Voigt, O. Alacam, M. Meuschke, K. Lawonn, and S. Zarrieß. The why and the how: A survey on natural language interaction in visualization. In *Proceedings of NAACL Conference: Human Language Technologies*, pp. 348–374, 2022. doi: 10.18653/v1/2022.naacl-main.27 2

[62] C. Wang and J. Han. DL4SciVis: A state-of-the-art survey on deep learning for scientific visualization. *IEEE Transactions on Visualization and Computer Graphics*, 29(8):3714–3733, 2023. doi: 10.1109/TVCG.2022.3167896 2

[63] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024. doi: 10.1007/s11704-024-40231-1 3

[64] Y. Wu, J. Meng, H. Li, C. Wu, Y. Shi, X. Cheng, C. Zhao, H. Feng, E. Ding, J. Wang, et al. OpenGaussian: Towards point-level 3D Gaussian-based open vocabulary understanding. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 19114–19138, 2024. 3

[65] Y. Wu, Y. Wan, H. Zhang, Y. Sui, W. Wei, W. Zhao, G. Xu, and H. Jin. Automated data visualization from natural language via large language models: An exploratory study. *Proceedings of the ACM on Management of Data*, 2(3):115, 2024. doi: 10.1145/3654992 2

[66] J. Xu, A. Szlam, and J. Weston. Beyond goldfish memory: Long-term open-domain conversation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 5180–5197, 2022. doi: 10.18653/v1/2022.acl-long.356 9, 10

[67] M. Yang, K. Tang, and C. Wang. Meta-INR: Efficient encoding of volumetric data via meta-learning implicit neural representation. In *Proceedings of IEEE Pacific Visualization Conference (Visualization Notes)*, pp. 246–251, 2025. doi: 10.1109/PacificVis64226.2025.00030 2

[68] S. Yao, J. Han, and C. Wang. GMT: A deep learning approach to generalized multivariate translation for scientific data analysis and visualization. *Computers & Graphics*, 112:92–104, 2023. doi: 10.1016/J.CAG.2023.04.002 2

[69] S. Yao, Y. Lu, and C. Wang. ViSNeRF: Efficient multidimensional neural radiance field representation for visualization synthesis of dynamic volumetric scenes. In *Proceedings of IEEE Pacific Visualization Conference*, pp. 235–245, 2025. doi: 10.1109/PacificVis64226.2025.00029 2, 4

[70] S. Yao and C. Wang. ReVolVE: Neural reconstruction of volumes for visualization enhancement of direct volume rendering. *Computers & Graphics*, 2025. Accepted. 4

[71] S. Yao and C. Wang. VolSegGS: Volumetric segmentation of dynamic visualization scenes using deformable Gaussian splatting. *IEEE Transactions on Visualization and Computer Graphics*, 32(1), 2026. Accepted. 2

[72] M. Ye, M. Danelljan, F. Yu, and L. Ke. Gaussian Grouping: Segment and edit anything in 3D scenes. In *Proceedings of European Conference on Computer Vision*, 2024. doi: 10.1007/978-3-031-73397-0_10 3

[73] B. Yu and C. T. Silva. FlowSense: A natural language interface for visual data exploration within a dataflow system. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1–11, 2020. doi: 10.1109/tvcg.2019.2934668 2

[74] T. Zhang, Y. Liu, B. Li, Z. Zeng, P. Wang, Y. You, C. Miao, and L. Cui. History-aware hierarchical transformer for multi-session open-domain dialogue system. In *Proceeding of Findings of the Association for Computational Linguistics: EMNLP*, pp. 3395–3407, 2022. doi: 10.18653/v1/2022.findings-emnlp.247 9, 10

[75] S. Zhi, T. Laidlow, S. Leutenegger, and A. J. Davison. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, pp. 15818–15827, 2021. doi: 10.1109/ICCV48922.2021.01554 3