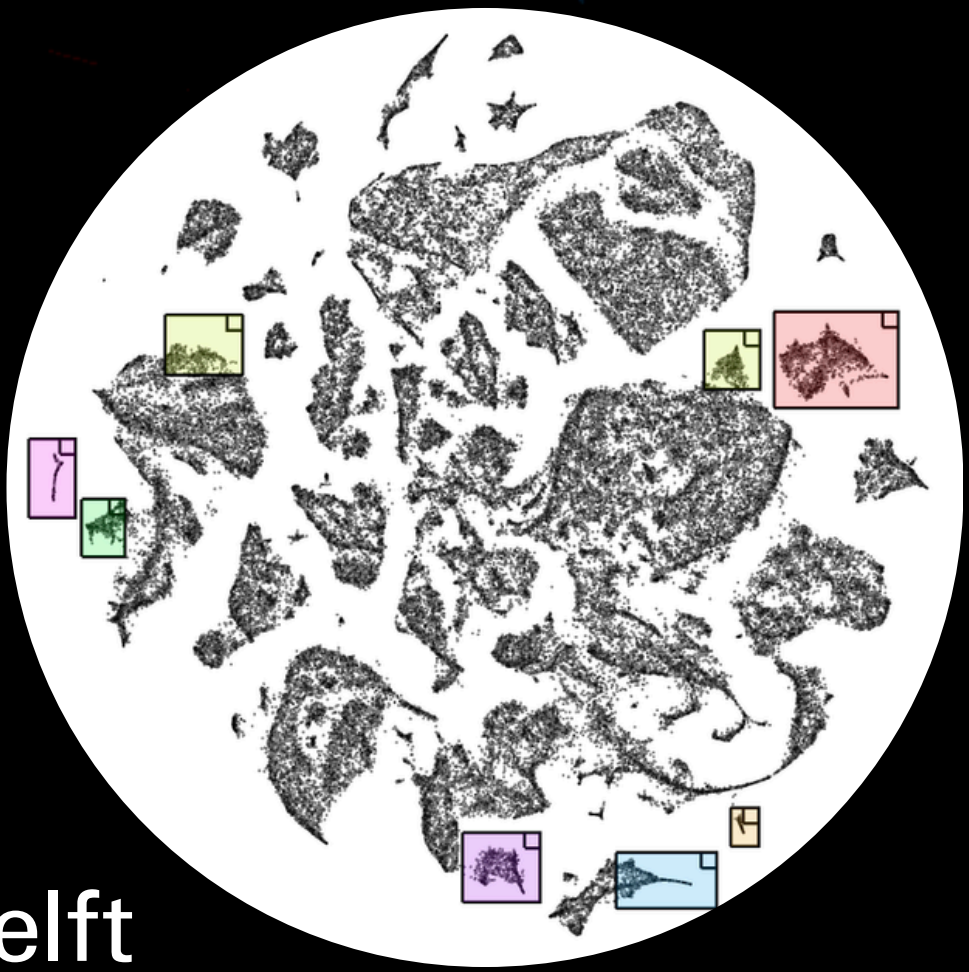
Hierarchical Dimensionality Reduction for Scalable Multivariate Volume Rendering

Olaf Heijl



Delft University of Technology

TU Delft

Hierarchical Dimensionality Reduction for Scalable Multivariate Volume Rendering

by

Olaf Heijl

Student number: 5342090

Thesis committee: T. Höllt — TU Delft, thesis advisor, daily supervisor
P. Kellnhofer — TU Delft, chairman
C. Raman — TU Delft

TU Delft

Preface

This thesis marks the completion of my master's degree in Computer Science at Delft University of Technology. It has been written as the final result of a research project on hierarchical dimensionality reduction for scalable multivariate volume rendering.

During this project, I learned a great deal about scientific visualization, transfer function design, dimensionality reduction, and the challenges involved in turning research ideas into a working implementation. At the beginning of the project, the direction was not yet fully clear to me. Over time, through experimentation, reading, implementation, and discussion, the focus became more concrete. This process was sometimes challenging, but it also made the project valuable and rewarding.

I would like to thank my thesis advisor and daily supervisor, T. Höllt, for the guidance, feedback, and support throughout the project. I also want to thank P. Kellnhofer and C. Raman for being part of my thesis committee and for taking the time to evaluate this work.

Finally, I would like to thank my family, friends, and fellow students for their support during my studies and during the writing of this thesis. Their encouragement helped me throughout the process and made the completion of this work possible.

Olaf Heijl
Delft, July 2026

Declaration of Generative AI Use

During this thesis, I used ChatGPT by OpenAI as an AI-assisted support tool. The tool was used to help brainstorm search terms for finding relevant literature, explore possible perspectives on the topic, support the structuring of text outlines, generate or refine illustrative material, and improve grammar, style, and clarity.

All scientific content, methodological choices, experiments, results, interpretations, and conclusions presented in this thesis are my own. The AI tool was not used to independently generate research results or replace my own analysis. All AI-assisted output was critically reviewed, edited, and verified before inclusion in the thesis. Sources and factual claims were checked independently, and the final responsibility for the content of this thesis lies entirely with the author.

Abstract

Multivariate volumetric datasets are becoming increasingly large and complex, making transfer function design for direct volume rendering more difficult. Recent work has shown that flat dimensionality reduction techniques, such as t-SNE, can support transfer function design by projecting high-dimensional voxel attributes into a two-dimensional embedding space. However, flat dimensionality reduction methods become difficult to scale to datasets containing millions of voxels. They produce visually cluttered transfer function domains and require large nearest-neighbor structures for mapping rendering samples to the embedding. In this work, we use Hierarchical Stochastic Neighbor Embedding (HSNE) as a scalable alternative for dimensionality reduction-based transfer function design in multivariate volume rendering. Instead of defining the transfer function over all voxels, we select a level of the HSNE hierarchy and use its landmarks as a reduced domain, and integrate this representation into the rendering pipeline. Our method is implemented and evaluated in the ManiVault framework using large multivariate tissue datasets. The results show that the HSNE-based approach significantly reduces preprocessing and rendering times compared to a t-SNE-based baseline, while also reducing visual clutter in the transfer function space. Higher hierarchy levels further improve runtime performance and simplify interaction, although they may lose fine detail. These results demonstrate that hierarchical dimensionality reduction can improve the scalability and usability of dimensionality reduction-based transfer function design for large multivariate volumetric datasets.

Contents

1 Introduction

The increasing availability of large, complex volumetric datasets through advances in technology and instrumentation [1, 2, 3] poses significant challenges for effective analysis and interpretation in scientific visualization. In fields such as medical imaging, fluid dynamics, and materials research, each voxel in a volumetric dataset can contain multiple interrelated variables. Direct volume rendering (DVR) enables visualization of such data by mapping the values to visual properties such as color and opacity through transfer functions (TFs). However, defining meaningful TFs becomes increasingly difficult as data dimensionality grows, since users must reason about relationships that are not directly visible in the original data space.

Recent work has shown that neighborhood-preserving dimensionality reduction (DR) can support TF design by projecting high-dimensional voxel attributes into a two-dimensional embedding. In this embedding, similar voxels are positioned close to each other, allowing users to select and color regions that correspond to meaningful data patterns. Snellenberg [4] demonstrated that embeddings created with methods such as t-SNE [5] and UMAP [6] can be used as an intuitive domain for multivariate TF design. Since these nonlinear DR methods do not provide an explicit mapping for newly interpolated samples during rendering, this approach evaluates the TF by using an approximate nearest neighbor (ANN) graph in the original high-dimensional attribute space.

Although this makes DR-based TF design possible for multivariate volume rendering, it also introduces scalability limitations. In a flat embedding, every voxel is represented as a point in the TF domain. For datasets containing millions of voxels, this creates two problems. First, the embedding becomes visually cluttered, making it difficult to identify clusters and define precise TF selections. Second, the ANN graph used during rendering must be constructed over a very large set of points, increasing preprocessing time and query times needed to find the closest neighbors of the samples. Since there are many samples along each viewing ray, this lookup step can become a major bottleneck for interactive rendering.

Our work addresses these limitations by using Hierarchical Stochastic Neighbor Embedding (HSNE) [7] as a hierarchical alternative to flat DR-based TF design. HSNE represents the data at multiple abstraction levels using landmark points, where higher levels summarize larger groups of voxels. Instead of defining the TF over all voxels, our work uses a selected HSNE hierarchy level as the TF domain. This reduces the number of points shown to the user and the number of points used in the ANN structure during rendering. As a result, the method aims to improve scalability and interaction usability while preserving the main structures needed for exploratory visualization.

The main contribution of this thesis is the design and evaluation of a scalable DR-based TF approach for multivariate volume rendering using HSNE landmarks. The contributions of this research are:

- Adaptation of a full data rendering pipeline in which interpolated samples are mapped to HSNE landmarks using nearest neighbor queries in the high-dimensional attribute space.
- Adaptation of a 2D position rendering pipeline that precomputes voxel-to-embedding mappings to reduce rendering costs during interaction.

- Quantitative and qualitative comparison with a flat t-SNE-based baseline, focusing on ANN construction time, rendering performance, TF interaction, visual clutter, and visual quality.

The remainder of this thesis is structured as follows. Chapter 2 provides the necessary background on DVR, TF, and DR techniques that are essential for understanding our proposed method. Chapter 3 reviews related work in DVR, TF design, and hierarchical visualization methods. Chapter 4 describes our proposed HSNE-based TF method and the two rendering pipelines. Chapter 5 presents the runtime, interaction, and visual quality evaluation. Finally, Chapter 6 concludes the thesis and discusses limitations and directions for future work.

2 Background

This chapter introduces the background concepts needed to understand the proposed method. First, the representation of volumetric and multivariate data is discussed in Section 2.1. Then, Section 2.2 explains direct volume rendering (DVR), including ray casting. Transfer functions (TFs) are explained in Section 2.3, along with the difference between pre-classification and post-classification. Section 2.4 introduces dimensionality reduction (DR) techniques, focusing on how they can be used to support TF design. Lastly, hierarchical approaches such as Hierarchical Stochastic Neighbor Embedding (HSNE) are explained in Section 2.5, with their relevance for large volumetric datasets.

2.1. Data Representation

A volumetric dataset can be described as a set V of samples (x, y, z, v), commonly referred to as voxels. Each voxel corresponds to a spatial position (x, y, z) in three-dimensional space and stores an associated data value v. This value represents a property of the underlying physical or simulated phenomenon at that location.

The stored value v can take different forms depending on the application. In the simplest case, it may be binary; for example, a 0 indicates the background, or a label i from a predefined set I indicates the presence of a specific object. However, in many scientific applications, the value represents a single measurable scalar quantity such as density, temperature, pressure, or intensity.

In more complex datasets, each voxel contains multiple attributes. In such cases, v can be represented as a vector, storing several variables at the same spatial location. Examples include velocity fields in fluid simulations, color values such as RGB triples, or measurements from multiple imaging modalities, such as anatomical and functional scans acquired from CT, MRI, PET, or fMRI. These multivariate volumetric datasets are common in modern medical and scientific visualization.

2.2. Direct Volume Rendering

A widely used technique for visualizing volumetric data without first extracting geometric representations such as iso-surfaces or polygonal meshes is DVR. Instead of transforming the data into explicit surfaces, DVR directly maps voxel values to optical properties. This approach enables the visualization of internal structures that would otherwise remain hidden in surface-based techniques.

In DVR, the volume is interpreted as a semi-transparent medium in which each spatial location contributes to the final image according to its optical characteristics. These characteristics are typically defined in terms of color and opacity and are assigned through a TF, which is explained in Section 2.3. The rendered image is generated by integrating these contributions along viewing rays.

2.2.1. Ray Casting and Optical Model

Ray casting is one of the most widely used techniques for implementing DVR. For each pixel in the 2D image plane, a viewing ray is generated from the camera position and traced through the volume domain. As illustrated in Figure 2.1, each ray traverses the volume and is evaluated at discrete sampling positions along its path.

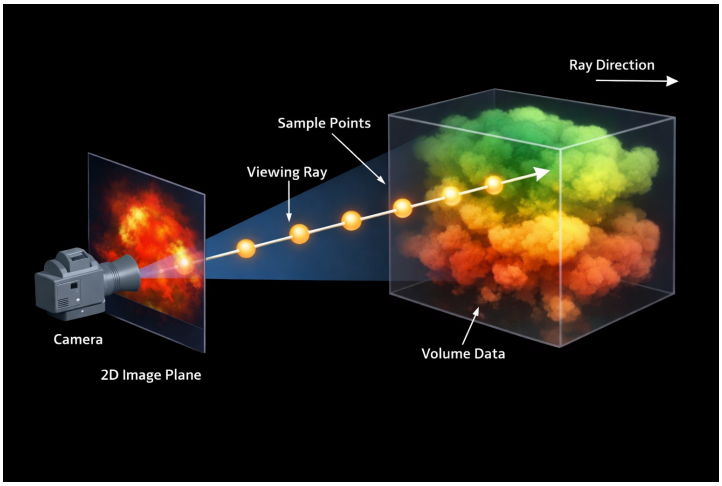


Figure 2.1: A camera casting a viewing ray through a 2D image plane into a semi-transparent volume cube, with discrete sample points marked along the ray inside the volumetric data.

At each sampling position s_i, the data value is usually not directly available since the sample position does not necessarily coincide with a voxel center. Therefore, the value at s_i is reconstructed from the surrounding voxels in the discrete grid using sampling strategies. These sampling strategies are discussed in more detail in Section 2.2.2. This reconstructed value is then mapped to optical properties, typically color C_i and opacity α_i, using a TF. The TF, therefore, determines how the data values contribute to the final image.

The contributions of all samples along a ray are accumulated using an optical compositing model. The emission-absorption model is adopted for our method. In this model, each sample emits light while also attenuating the light accumulated from previous samples along the ray. As the ray progresses through the volume, color and opacity are combined in order of increasing depth.

The accumulated color along a ray is computed incrementally using front-to-back compositing, as shown in Equation (2.1). At each sampling position i, the newly accumulated color C'_i is computed from the previously accumulated color C'_{i-1} and the premultiplied sample color C_i, weighted by the remaining transparency $1-\alpha'_{i-1}$. Here, C_i denotes the sample color after its RGB values have already been multiplied by the sample opacity α_i. The newly accumulated opacity α'_i is computed in Equation (2.2) from the previous accumulated opacity α'_{i-1} and the sample opacity α_i, again weighted by the remaining transparency $1-\alpha'_{i-1}$.

$$C'_i = C'_{i-1} + (1-\alpha'_{i-1})\, C_i \tag{2.1}$$

$$\alpha'_i = \alpha'_{i-1} + (1-\alpha'_{i-1})\, \alpha_i \tag{2.2}$$

The compositing process continues until the ray exits the volume or becomes fully opaque, thus $\alpha'_i \geq 1$. The final accumulated color determines the pixel value in the rendered image. This optical model enables semi-transparent visualization of internal structures, making DVR particularly suitable for medical and scientific volumetric data.

2.2.2. Sampling and Reconstruction Strategies

Volumetric data is stored as discrete samples on a regular 3D grid. However, during ray casting, sampling positions along a ray generally do not coincide with voxel centers. Therefore, continuous data values must be reconstructed from neighboring voxel samples.

The simplest reconstruction strategy is nearest neighbor interpolation, where the value of the closest voxel is used. Although computationally efficient, this method can introduce block-like artifacts and abrupt intensity changes.

More commonly, trilinear interpolation is applied. In this strategy, the value at a sampling position is computed as a weighted combination of the eight surrounding voxels. This produces smoother transitions and improves visual quality while maintaining a reasonable computational cost.

In addition to interpolation, the choice of sampling step size along the ray is important. A smaller step size increases visual accuracy and reduces aliasing artifacts but requires more sampling evaluations and higher computational cost. A larger step size improves performance but may miss fine structures or introduce visual inaccuracies.

Together, the interpolation method and the sampling density directly influence rendering quality and runtime speed. Since the TF is evaluated at every sampling position, the number of samples per ray also determines the computational cost of TF evaluation during rendering.

2.3. Transfer Functions

In DVR, the TF defines how data values are mapped to visual properties such as color and opacity. While ray casting determines how samples are accumulated along a viewing ray, the TF determines what each sample contributes to the final image. As such, the TF plays a central role in revealing structures of interest within volumetric data.

For each reconstructed sample value v, the TF assigns a color $C(v)$ and an opacity $\alpha(v)$. These optical properties directly influence the compositing model, as seen in Equation (2.1) and Equation (2.2). Regions assigned high opacity become visually prominent, whereas regions with low opacity become transparent. Similarly, color mappings allow for differentiation between materials, tissues, or features within the data.

2.3.1. One- and Multi-Dimensional TFs

The most basic form of a TF is one-dimensional (1D), where optical properties depend on a single scalar data value, typically intensity. In this case, the TF can be represented as a curve or a set of control points defined over the data value range.

1D TFs are intuitive and computationally efficient. They are commonly used in applications such as CT or MRI visualization, where intensity values often correspond to specific materials or tissue types. However, in many datasets, different structures may share overlapping intensity ranges. In such cases, a purely intensity-based mapping is insufficient for reliable separation.

To better distinguish structures, multi-dimensional TFs incorporate additional voxel attributes beyond raw intensity. These attributes may include the magnitude of the gradient, curvature, texture measures, or other derived features. By mapping combinations of attributes to optical properties, multi-dimensional TFs enable more selective and expressive visualizations.

For example, the magnitude of the gradient is often used together with the intensity to emphasize the boundaries of the material. In such a 2D TF, opacity may be assigned based on both intensity and gradient magnitude, allowing boundary regions to be highlighted while homogeneous regions remain transparent.

More generally, a multi-dimensional TF defines a mapping:

$$(v_1, v_2, ..., v_n) \rightarrow (C, \alpha), \tag{2.3}$$

where each voxel is described by multiple attributes. While this increases expressive power, it also significantly increases complexity.

As the number of voxel attributes increases, the TF design space becomes high-dimensional and difficult to navigate. Instead of adjusting a single intensity curve, users must reason about relationships between multiple variables simultaneously. Because humans cannot directly visualize more than two or three dimensions at once, understanding how different attributes interact becomes challenging. Small adjustments in one dimension may have unexpected effects when combined with others, making the design process less predictable.

In high-dimensional settings, the parameter space of the TF grows rapidly, which increases complexity and makes systematic exploration difficult. Structures that overlap in some attributes but differ in others

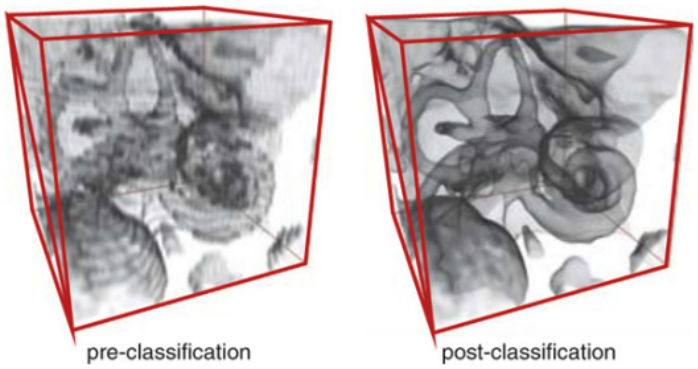


Figure 2.2: Visual comparison between pre-classification and post-classification. Pre-classification can introduce a loss of detail, while post-classification better preserves fine detail and clear boundaries. Image from [8].

may still be hard to separate without careful tuning. As a result, users often rely on trial-and-error, iteratively modifying parameters, and observing the rendered output. This process can be time-consuming and cognitively demanding, especially for large multivariate datasets.

These limitations highlight the need for techniques that simplify the high-dimensional attribute space into more manageable representations. DR methods address this challenge by projecting multivariate voxel attributes into lower-dimensional spaces that are easier to visualize, explore, and use for TF design. These methods are discussed in more detail in Section 2.4.

2.3.2. Pre-Classification and Post-Classification

The evaluation of a TF can be performed either before or after interpolation during ray casting. These two strategies are commonly referred to as pre-classification and post-classification.

In pre-classification, the TF is applied directly to the discrete voxel values stored in the grid. Color and opacity are assigned at the voxel centers, and afterward, interpolation is performed on these optical properties. While computationally efficient, pre-classification may introduce visual artifacts, especially when sharp transitions are present in the TF.

In post-classification, interpolation is first performed on the data values to obtain the reconstructed scalar or multivariate value at each sampling position. The TF is applied to this interpolated value. This approach produces smoother and more accurate visual results, particularly when using non-linear or complex TFs.

Post-classification is generally more expensive than pre-classification, because the TF is evaluated after interpolation at the sampling positions along each ray. In standard scalar volume rendering, however, this additional cost is usually limited, since the TF evaluation is typically a simple lookup. The implications of post-classification become more important when the TF is defined in a DR space, as discussed in Section 2.4.3.

The main advantage of post-classification is the improved visual quality. As illustrated in Figure 2.2, pre-classification can introduce a loss of detail and blur clear boundaries, while post-classification preserves these structures better. For this reason, post-classification is commonly used when visual accuracy is important.

2.4. Dimensionality Reduction for Visualization

Modern scientific and medical datasets often contain multiple variables per data point. Each voxel can therefore be described by attributes such as intensity, gradient magnitude, curvature, and modality-specific measurements. While this increases expressive power, it also makes analysis and interaction more difficult. Humans cannot directly reason about high-dimensional spaces, and direct visualization beyond three dimensions is not possible without transformation.

This challenge is addressed by DR by projecting high-dimensional data into a lower-dimensional space, typically two dimensions. The goal is to make the data easier to visualize and explore while still preserving meaningful structural relationships.

The goal of DR in visualization is, therefore, not merely compression but structure preservation. Points that are similar in the original high-dimensional space should remain close in the low-dimensional embedding, while dissimilar points should remain separated.

In the context of TF design, DR enables users to interact with complex voxel attribute spaces through a 2D representation. Instead of reasoning about multiple attributes separately, users can explore clusters and patterns directly in the embedded space.

2.4.1. Linear Techniques

Linear DR methods assume that the data lies approximately on a linear subspace of the high-dimensional space. This means that the main variation in the data can be captured by a limited number of linear directions. The most widely used linear DR technique is Principal Component Analysis (PCA) [9].

PCA identifies orthogonal directions, called principal components, in which the data experiences maximum variance. By projecting the data onto the first few principal components, a lower-dimensional representation is obtained that captures as much global variance as possible.

Linear methods such as PCA are computationally efficient and mathematically well-defined. They preserve global structure and are easy to interpret. However, they are limited in their ability to represent non-linear relationships. If the data are in a curved manifold embedded in high-dimensional space, a linear projection may distort important neighborhood relationships.

For complex multivariate volumetric datasets, where attributes often exhibit non-linear dependencies, purely linear DR methods may fail to capture intrinsic data structure.

2.4.2. Non-Linear Techniques

To better capture intrinsic structure, non-linear DR techniques have been developed. Unlike linear methods, these approaches assume that the data may lie on a non-linear manifold embedded in high-dimensional space.

Techniques such as t-SNE [5] and UMAP [6] focus on preserving neighborhood relationships. Instead of maximizing global variance, they try to maintain local similarities between data points. Points that are similar in the original space are mapped close together in the embedding, while distinct points are pushed apart.

T-SNE constructs probability distributions over pairwise similarities in both high- and low-dimensional spaces and minimizes their divergence. This often produces visually well-separated clusters, making it particularly suitable for exploratory data analysis. However, t-SNE may distort global relationships and distances between clusters.

UMAP also preserves local structure but introduces a stronger theoretical foundation based on manifold learning and topological structures. It often provides better global organization and improved scalability compared to t-SNE.

2.4.3. Limitations of Flat Embeddings

Although flat non-linear DR methods provide powerful visualizations, they also introduce important limitations when applied to large volumetric datasets and interactive TF design.

The first major limitation is the non-parametric nature of non-linear techniques such as t-SNE and UMAP. Unlike linear methods such as PCA, these approaches do not learn an explicit mapping from the original high-dimensional attribute space to the low-dimensional embedding. Instead, the embedding is constructed only for the set of samples used during optimization.

This limitation becomes important in DVR when using post-classification. During ray casting, sampling positions are reconstructed continuously through interpolation and therefore generally do not correspond exactly to existing voxels in the dataset. In post-classification, the TF must be evaluated for every interpolated sample along every viewing ray. For a standard scalar TF, this is usually a simple

lookup. However, when the TF is defined in a non-linear DR embedding, the sample first needs to be mapped to a position in the embedding space. Because non-linear DR techniques are non-parametric, a new interpolated sample has no directly available position in the precomputed embedding. Adding this sample to the embedding would require recomputing or extending the non-linear embedding, which is not feasible during rendering.

The full data pipeline proposed by Snellenberg [4] addresses this issue by constructing an Approximate Nearest Neighbor (ANN) graph on the voxel attributes. For each interpolated sample, similar voxels are searched in the graph. The corresponding embedding positions of these voxels are then used for TF evaluation. More details on this process are provided in Section 4.3.1.

However, this introduces a second limitation. As the size of the volumetric dataset increases, the ANN graph also becomes significantly larger. Consequently, nearest-neighbor searches become more expensive in both memory usage and runtime. Since these lookup operations must be performed repeatedly for millions of sampling positions during rendering, the ANN search can become a major computational bottleneck and reduce interactive rendering performance.

A third limitation of flat embeddings is visual clutter. In a flat embedding, all voxels are projected into a single global 2D space at the same level of abstraction. For large datasets, this can produce dense embeddings in which clusters overlap and fine structures become difficult to distinguish or select accurately. As a result, users may struggle to navigate the embedding efficiently and isolate meaningful regions for TF design.

These limitations motivate hierarchical DR approaches such as HSNE. By organizing the embedding across multiple abstraction levels, HSNE reduces visual clutter while also reducing the effective search space for neighborhood lookups, thereby improving scalability and rendering performance.

2.5. Hierarchical DR

Traditional DR methods produce a single embedding in which all data points are represented at the same level of detail. While this flat representation may reveal clusters and local relationships, it becomes difficult to interpret when the dataset is large. Millions of points projected into a single 2D space can lead to visual clutter and reduced usability.

Hierarchical DR addresses this limitation by organizing data across multiple levels of abstraction. Instead of constructing one global embedding, the data is represented at several scales. Higher levels provide a coarse overview of the global structure, while lower levels reveal increasingly detailed local neighborhoods. This multi-scale organization supports a structured exploration strategy. Users can first inspect the global distribution of the data at a high abstraction level, identify regions of interest, and then progressively refine their view by navigating to more detailed levels.

The key idea is that abstraction is not achieved through simple subsampling, but through a selection of representative points that preserve neighborhood relationships. A well-designed hierarchical embedding must maintain consistency between levels so that local structure remains meaningful when transitioning between scales.

2.5.1. HSNE

The hierarchical approach HSNE [7] extends t-SNE to support scalable, multi-level exploration of large, high-dimensional datasets. While t-SNE constructs a single embedding that preserves local neighborhood similarities, HSNE builds a hierarchy of embeddings that enables exploration from overview to detail without recomputing the entire embedding. This overview-to-detail idea is illustrated in Figure 2.3, where clusters at the overview level are linked to more detailed embeddings at the data level.

The construction of HSNE begins with a k-nearest neighbor (kNN) graph in the original high-dimensional space. Instead of directly embedding all points, HSNE identifies a subset of representative points, called landmarks, that capture the structure of the dataset at a coarser scale.

The landmarks are selected using a random walk-based importance measure defined on the kNN graph. Points that are frequently visited during random walks are considered structurally important and chosen as representatives of the next abstraction level. This ensures that densely connected regions

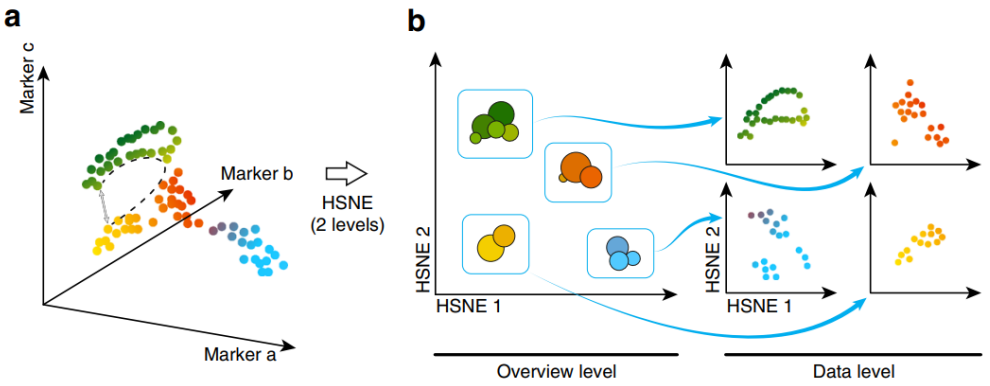


Figure 2.3: Schematic illustration of HSNE overview-to-detail exploration. (a) A small high-dimensional dataset is shown using three marker dimensions, where points form several local clusters. (b) HSNE represents this data using a hierarchy of embeddings. The overview level contains landmark groups that summarize the main cluster structure, while the data level shows more detailed embeddings of the selected regions. The arrows indicate how regions at the overview level are linked to their corresponding finer-scale representations. Image adapted from [10].

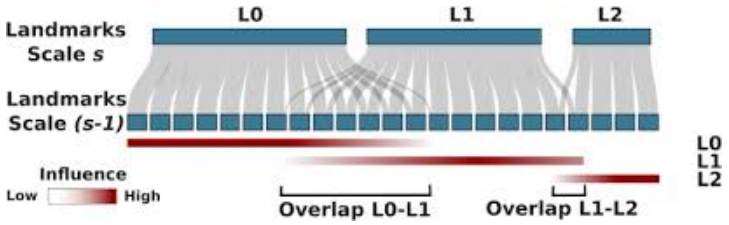


Figure 2.4: The area of influence of landmarks across multiple HSNE levels can be seen. Each landmark at a higher level represents a region of influence over points in the lower level, visualized as converging flows. Overlap between these regions indicates similarity between landmarks and reflects how local structures are preserved across scales. Image from [7].

and meaningful clusters remain well represented at higher levels.

Each landmark represents a group of points on the level below. The influence of lower-level points on higher-level landmarks is captured through area-of-influence relationships, illustrated in Figure 2.4. Importantly, HSNE avoids collapsing unrelated regions together, a common issue in naive subsampling approaches. Instead, neighborhood relationships are preserved across levels.

Once the landmarks are selected, a t-SNE-like embedding is computed only for the landmarks at that level. Because the number of landmarks is significantly smaller than the number of original data points, this reduces computational cost and visual clutter. If a user selects a region of interest in the embedding, HSNE allows exploration of the corresponding subset at the next finer level, where new landmarks are embedded with respect to their local neighborhood structure.

This process can be repeated across multiple levels, forming a hierarchy from a coarse global overview to fine local detail. Crucially, each level preserves meaningful neighborhood relationships without requiring the full dataset to be embedded simultaneously.

3
Related Work

This chapter reviews prior work related to scalable transfer function (TF) design for multivariate volume rendering. Existing approaches for direct volume rendering (DVR) and TF design are discussed in Section 3.1, with a focus on the increasing difficulty of defining meaningful mappings for high-dimensional voxel attributes. Next, dimensionality reduction (DR) methods are reviewed as a way to support TF design through lower-dimensional representations of multivariate data. In Section 3.2, hierarchical and multi-scale approaches are discussed, including hierarchical DR methods such as Hierarchical Stochastic Neighbor Embedding (HSNE). Together, these works provide the context for positioning our proposed method as a hierarchical DR-based approach for improving the scalability of TF design and evaluation.

3.1. TF Design for DVR

Ray casting and DVR have long been used to visualize volumetric data, allowing the exploration of internal structures without the need for explicit surface extraction [11]. Over the years, extensive research has focused on improving rendering quality and efficiency, including the development of advanced optical models and shading methods [12, 13] and GPU-based implementations for better interactive performance [14, 15]. While these advances have significantly improved visual fidelity and interactivity, they assume that meaningful TFs are already available and do not address the challenges of designing TFs for high-dimensional, multivariate volume data.

A central problem in DVR is TF design, since the TF determines how data values are mapped to visual properties such as color and opacity. Previous surveys have shown that TFs can be specified using a wide range of strategies, including data-driven, feature-based, and interaction-driven approaches [16]. As volumetric datasets have become more complex, simple one-dimensional TFs are often not enough to separate structures in multivariate volume data. Early extensions of one-dimensional TFs incorporated additional features such as gradients, curvature, or texture to better highlight structural boundaries and subtle features within the data [17, 18, 19, 20]. Feature-based multi-dimensional TFs allow users to define mappings based on derived voxel attributes rather than raw intensity values, improving the separation of structures that are difficult to distinguish using traditional histogram-based methods [21]. However, as the number of voxel attributes increases, these approaches shift the burden to the user, who must reason about complex relationships in high-dimensional feature spaces that are not directly observable. As a result, TF design remains a largely manual and exploratory process that does not scale well with data dimensionality.

More recent research has focused on reducing manual trial-and-error in TF design by leveraging data-driven and learning-based methods. Optimization-based frameworks guide TF selection using predefined objectives or interactive feedback [22, 23]. Learning-based methods, including differentiable rendering and latent design spaces, model the TF space to enable automatic suggestion or refinement of mappings [22, 23, 24]. These approaches aim to optimize or automate TF design by encoding prior knowledge or learned objectives into the system. While effective for specific tasks, they often reduce direct user control and transparency, which can be limiting in exploratory visualization scenarios where analysts seek to iteratively probe and interpret complex, unfamiliar data. In contrast, this work focuses

on improving the scalability and computational efficiency of TF evaluation for large multivariate volumetric datasets, rather than automating TF decisions.

Other recent work has focused on improving the interaction with multivariate TFs directly. Information-display-inspired TF editors have been proposed for volume datasets with more than three channels, allowing users to inspect and manipulate relationships between multiple variables in a more structured interface [25]. Such approaches improve the usability of TF design for multivariate data, but they do not specifically address the scalability limitations of DR-based TFs, where large flat embeddings and nearest neighbor (NN)-based evaluation can become computational bottlenecks during rendering.

As TFs become more complex for multivariate and high-dimensional volumetric datasets, exploring the full space of voxel attributes becomes increasingly difficult. DR has therefore become a key technique for enabling TF design by projecting high-dimensional voxel attributes into lower-dimensional spaces that can be processed, visualized, and mapped to optical properties. In practice, DR is not only a conceptual aid for understanding multivariate data, but also a practical mechanism for making TF design and evaluation feasible for large volumes. Early work on DR for TF design primarily relied on linear techniques, with Principal Component Analysis (PCA) [26, 27] being the most widely used method. PCA has been applied to guide TF design for multi-dimensional imaging spectroscopy data by capturing directions of maximum variance in the data [28]. While linear DR methods, like PCA, are computationally efficient and easy to interpret, they are limited in their ability to represent non-linear relationships and complex manifolds that frequently occur in multivariate volumetric data.

To better capture intrinsic data structure, non-linear DR methods such as t-SNE [5], UMAP [6], Isomap [29], and Locally Linear Embedding (LLE) [30] have been adopted in the context of TF design. These methods aim to preserve local neighborhood relationships and manifold geometry, resulting in embeddings that more accurately reflect similarities among voxels. Several approaches have used Isomap and LLE for the design of TF in multi-channel datasets [31], while more recent work has demonstrated that t-SNE-based embeddings can effectively support exploratory TF design by clustering voxels with similar multivariate characteristics [4, 32, 33].

Several extensions have been proposed to improve the scalability of t-SNE. Tree-based approximations reduce the computational cost of t-SNE by accelerating the optimization process, making it more practical for larger datasets [34]. Parametric variants learn an explicit mapping from the high-dimensional data space to the low-dimensional embedding space, which can support the projection of previously unseen samples without recomputing the full embedding [35]. However, these methods primarily address the cost of embedding computation or out-of-sample projection. They do not directly reduce visual clutter in the TF domain, nor do they provide a hierarchical landmark structure that reduces the number of elements involved in TF interaction and NN-based evaluation during rendering.

As explained in Section 2.4.3, commonly used flat non-linear embeddings are non-parametric. Therefore, an interpolated sample generated during ray casting cannot be directly mapped to a position in the precomputed embedding space. Snellenberg [4] addressed this limitation by introducing an Approximate Nearest Neighbor (ANN) graph over the voxel attributes. During rendering, each sample is matched to nearby voxels in the original high-dimensional attribute space, after which the corresponding embedding positions can be used for TF evaluation.

Although this approach makes DR-based TF evaluation feasible, it does not fully remove the scalability problem. As volumetric datasets grow to millions of voxels, the ANN graph also grows in size. This increases both memory usage and query time, since NN searches have to be performed repeatedly for a large number of samples during rendering. As a result, the mapping from interpolated samples to the embedding space can still become a major performance bottleneck, limiting immediate visual feedback during interactive TF design.

This motivates the use of a hierarchical representation. Instead of constructing the TF domain and NN structure over all voxels in a flat embedding, this work investigates the use of an HSNE landmark level as a computational abstraction layer. By defining the TF over a selected level of the hierarchy, the number of embedded elements used for interaction and NN queries is reduced, while the relevant neighborhood structure is still preserved. In this way, HSNE is not used primarily as an overview-to-detail exploration technique, but as a mechanism for improving the scalability of DR-based TF design and evaluation in multivariate volume rendering.

3.2. Hierarchical and Multi-Scale Approaches

Previous work has also used rendering-side acceleration and hierarchical representations to improve the interactivity of volume visualization. GPU-based ray-casting techniques have been used to achieve real-time rendering of discrete isosurfaces while supporting advanced shading effects [36]. Other work has used multiresolution texture-based volume visualization, where large volumes are represented at multiple spatial resolutions to support interactive exploration of large datasets [37].

These approaches share the goal of improving scalability and interactivity for large volumetric datasets. However, they operate primarily on the rendering or spatial representation of the volume. In contrast, our work does not introduce a new isosurface renderer, texture-based renderer, spatial octree, or level-of-detail volume representation. Instead, the hierarchy is introduced in the high-dimensional attribute space through HSNE. The aim is therefore not to reduce the spatial resolution of the volume, but to reduce the number of elements involved in DR-based TF design and evaluation. This distinction makes hierarchical DR complementary to traditional rendering-side acceleration techniques.

Hierarchical representations, abstraction techniques, and multi-scale interaction have also been studied more broadly as ways to manage complexity in visualization. Abstraction-based approaches simplify complex data or visual representations while aiming to preserve the most relevant structures for analysis [38]. Focus+context and multi-scale interaction techniques allow users to inspect data at different levels of detail without losing awareness of the surrounding context [39, 40]. These ideas are relevant to scalable visualization because they reduce visual complexity and support structured exploration. However, they do not directly address the specific challenge considered in this work: defining and evaluating TFs over high-dimensional voxel attribute spaces. In contrast, our approach uses hierarchy in the DR domain, where HSNE landmarks reduce the number of elements involved in TF interaction and NN-based evaluation.

In parallel, hierarchical DR methods have been proposed to support scalable visual analysis of high-dimensional data. HSNE constructs a hierarchy of representative landmark points, where landmarks at higher abstraction levels summarize regions of influence over points from lower levels [7]. Instead of embedding all data points simultaneously, this hierarchy allows users to inspect a smaller set of representative points while preserving relevant neighborhood relationships across levels.

This hierarchical structure has been applied successfully in biomedical visualization, where HSNE is used to analyze large mass cytometry datasets and reveal rare cell populations through interactive exploration across multiple scales [10]. In that setting, the hierarchy primarily supports an overview-to-detail workflow: users start from a coarse embedding, select regions of interest, and then navigate to finer levels for more detailed analysis. In contrast, our presented work does not use HSNE mainly as an interactive navigation mechanism. Instead, it uses a selected HSNE landmark level as a computational abstraction layer for DR-based TF design and evaluation. The goal is therefore not only to support multi-scale exploration but also to reduce the number of elements involved in TF interaction and NN-based evaluation during volume rendering.

Hierarchical ideas have also been explored in TF design itself. For example, hierarchical clustering of material boundaries has been used to automate TF generation by organizing volumetric structures in a coarse-to-fine manner [41]. This demonstrates that a hierarchy can help structure the TF design process and reduce manual trial-and-error. However, this type of approach focuses on classification and TF specification rather than on improving the runtime cost of evaluating a DR-based TF during rendering.

4

Method

To improve scalability, reduce visual clutter in dimensionality reduction (DR)-based transfer function (TF) design, and preserve the main structures in the visualization, we propose a method that combines hierarchical DR, TF design, and direct volume rendering (DVR). The method uses a selected level of the Hierarchical Stochastic Neighbor Embedding (HSNE) hierarchy as the domain of the TF. Instead of defining the TF over all voxels in a flat embedding, the user interacts with a reduced set of HSNE landmarks, each representing a subset of voxels in the original volumetric dataset. An overview of the method is shown in Figure 4.1.

The input of the system is a multivariate volumetric dataset in which each voxel is described by a set of attributes. These attributes are used to construct an HSNE hierarchy, from which one two-dimensional landmark embedding is selected for TF design. In this embedding, nearby landmarks represent voxels with similar high-dimensional attribute patterns. The user defines the TF by selecting regions in the embedding and assigning color and opacity values to them. These selections determine which structures become visible in the final rendering and how they are visually represented.

To generate the final visualization, the TF is integrated into the DVR process. For each sample along a viewing ray, the method determines a corresponding position in the selected embedding. The TF is then evaluated at this position to obtain color and opacity values, which are accumulated using front-to-back compositing.

Our system supports two rendering pipelines that differ in how this mapping to the embedding space is performed. The first pipeline, the full data renderer, operates on the original high-dimensional voxel attributes and maps each interpolated sample to the embedding using NN queries. This preserves a closer relation to the original attribute space but is computationally more expensive. The second pipeline, the 2D position renderer, precomputes embedding positions for voxels and interpolates these positions directly during rendering. This enables faster rendering but may introduce visual artifacts when interpolation in the embedding space crosses unrelated TF regions.

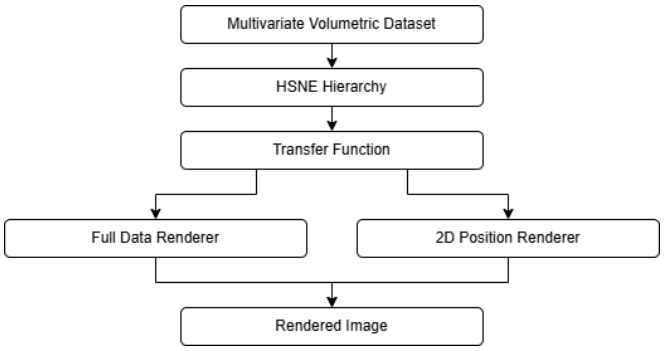


Figure 4.1: Overview of our proposed method. The input multivariate volumetric dataset is processed using HSNE to generate a hierarchical embedding. A single selected layer from this hierarchy is used to construct a TF, which is then integrated into two rendering approaches: the full data renderer or the 2D position renderer, ultimately producing the final visualization.

Our proposed method builds on the DR-based TF approach introduced by Snellenberg [4], which uses a flat t-SNE embedding as the TF domain in ManiVault. In contrast, our method replaces this flat embedding with a selected HSNE landmark level. This changes both the interaction space and the rendering pipeline: the number of points shown in the TF domain is reduced, and the number of elements involved in NN-based TF evaluation can be decreased. As a result, the method aims to improve scalability, interaction usability, and rendering performance for large multivariate volumetric datasets, while preserving the visual quality of the final rendering.

Section 4.1 explains the use of a single HSNE hierarchy level as the TF domain. Section 4.2 discusses how the landmark embedding affects TF design, particularly for large volumetric datasets. Section 4.3 describes the two rendering pipelines and their trade-offs.

4.1. HSNE-based DR

We use HSNE [7] in this method as a scalable alternative to flat DR techniques to construct the TF domain. Rather than embedding all voxels, HSNE obtains a reduced representation consisting of landmarks. This reduction is essential to avoid visual clutter in the TF domain; more in-depth details about this are given in Section 4.2. Another reason why this reduction is essential is to integrate it into the rendering pipeline, as it significantly decreases both the preprocessing time and the computational cost of algorithms such as nearest neighbor (NN) queries. More in-depth details on this are provided in Section 4.3.

Although HSNE provides multiple levels of abstraction, in this work, we select a single level of the hierarchy. This design choice is motivated by several considerations. First, restricting the method to a fixed embedding simplifies the user interaction, as users can define and refine the TF within a stable two-dimensional space without the need to navigate between levels. Second, it avoids the complexity of maintaining consistency between TF definitions across different levels of abstraction. Finally, it reduces computational overhead during rendering, as no additional logic is required to dynamically select or interpolate between hierarchy levels.

However, this simplification introduces a trade-off between abstraction and detail in the TF domain. Coarser levels of the hierarchy provide a more compact representation, improving usability and runtime speed, but they may merge distant structures or fine-grained features. In contrast, finer levels maintain more detail but they increase the number of landmarks, partially reducing the scalability benefits of HSNE and increasing interaction complexity. The user should therefore select an abstraction level that is a compromise between preserving relevant structures in the data and maintaining an efficient and usable representation.

Although HSNE supports exploration across multiple hierarchy levels, our work uses a single selected level as the TF domain. A possible alternative would be to allow users to refine selected regions by transitioning to a finer HSNE level. This could reveal additional landmarks and provide more detailed control over local structures. However, such a multi-level interaction would require TF definitions to remain consistent across different hierarchy levels, which is non-trivial because each level contains a different landmark representation. It would also increase the complexity of both the user interface and the rendering pipeline. For these reasons, dynamic refinement is not included in our proposed method and is instead considered as future work.

The HSNE computation is performed using the existing implementation within the ManiVault framework. The resulting two-dimensional embedding of the selected hierarchy level is used as the domain for TF design, forming the basis for the subsequent stages of the visualization pipeline.

4.2. TF of HSNE Embedding

The TF in our method is defined in the two-dimensional embedding generated from the selected HSNE hierarchy level. This embedding serves as an abstract domain in which high-dimensional voxel attributes are represented as points, enabling users to interact with complex multivariate data through a two-dimensional interface. Each point in this space corresponds to a landmark, which in turn represents a subset of voxels in the original volumetric dataset.

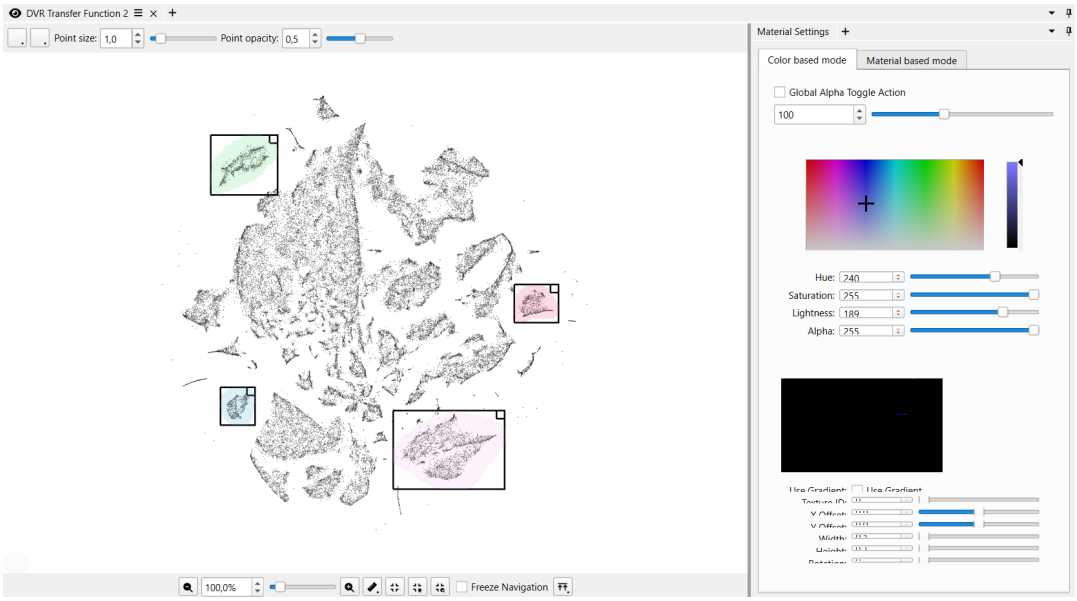


Figure 4.2: Interactive TF design. Landmarks are visualized as points in a two-dimensional layout, where selected regions are assigned color and opacity. These selections define the mapping from embedding space to optical properties used during volume rendering. On the right, color and opacity can be specified for each selection.

The TF is constructed through direct interaction in the embedded space. As shown in Figure 4.2, the user selects regions within the two-dimensional layout to identify groups of landmarks. These selected regions are assigned color and opacity values, defining how the corresponding voxel subsets contribute to the final rendering. In this way, the TF establishes a mapping from regions in the embedding space to optical properties used during DVR.

Compared to embeddings generated using flat DR methods such as t-SNE, the use of HSNE improves the usability of the TF domain, particularly for large volumetric datasets. Since HSNE represents the data using a reduced set of landmarks instead of all voxels, the resulting embedding is significantly less dense. This reduction decreases visual clutter, making it easier to identify and select meaningful regions. An example can be seen in Figure 4.3. Furthermore, because each landmark represents a coherent group of voxels, interactions in the TF naturally operate on aggregated structures rather than individual samples. This potentially leads to more stable and manageable selections compared to densely populated embeddings, where overlapping points can obscure the underlying patterns.

Despite these advantages, defining TFs in nonlinear DR spaces is inherently challenging. In traditional TF design, the axes correspond to physically meaningful quantities, such as intensity or gradient magnitude, allowing users to reason directly about how data values map to visual properties. In contrast, the axes of a nonlinear embedding do not have a direct interpretation. Instead, the structure of the embedding is determined by similarity relationships in the high-dimensional space. As a result, users must rely on spatial patterns, such as clusters and relative proximity, to guide interaction, which can make the TF design less intuitive.

In contrast to traditional TF design, where mappings are often more directly interpretable and reproducible, the TF definition in an embedded space introduces a degree of subjectivity. Different users may interpret the same embedding differently, leading to variations in the resulting visualization. This subjectivity is an inherent aspect of DR-based TF design and should be considered when analyzing and comparing results.

A more fundamental issue is that distances in the embedding space do not necessarily correspond to similarities in the original high-dimensional attribute space. Although HSNE aims to preserve local neighborhood relationships, global distances can be distorted. Hence, points that appear close to each other in the embedding are not guaranteed to represent similar voxel attributes, and similar structures in the data may be separated in the embedding. This introduces ambiguity during interaction: a contiguous region selected in the embedding may group together voxels that are not semantically related,

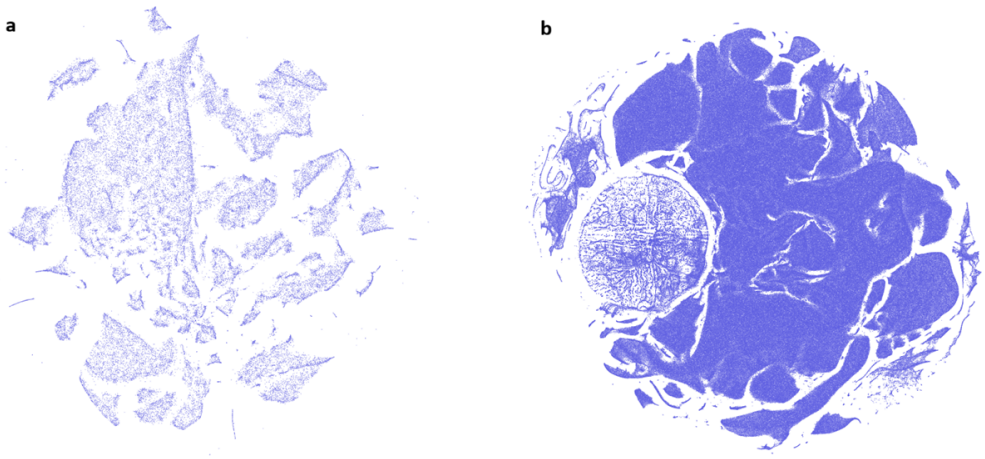


Figure 4.3: Two DR techniques applied to the same tissue dataset containing 5,941,250 data points. (a) HSNE embedding at the first level of a three-level hierarchy, consisting of 53,688 landmarks. (b) t-SNE embedding of the full dataset, showing all data points. The t-SNE embedding shows significant visual clutter due to the high point density, which complicates the selection of meaningful regions for TF design, whereas the HSNE embedding provides a more manageable and interpretable representation.

while meaningful structures may be fragmented across multiple regions. In addition, the cluster boundaries are not explicitly defined, requiring users to interpret visual patterns that depend on the specific embedding configuration.

The effectiveness of our approach therefore depends on the assumption that the HSNE embedding provides a meaningful organization of the data with respect to the features of interest. If relevant structures are not well separated or distorted in the embedding, the resulting TF may not be able to isolate them effectively. As such, the quality of the final visualization is inherently tied to the quality of the DR.

These challenges are further influenced by the use of HSNE landmarks. Since each landmark represents a subset of voxels, assigning color and opacity to a single point effectively assigns these properties to an entire group of voxels. While this aggregation reduces the complexity of the TF domain and mitigates visual clutter, it also introduces a loss of detail, as variations within each group are not explicitly represented. As a result, the expressiveness of the TF depends on how well the selected landmarks capture the relevant structures in the data.

During rendering, the TF is evaluated by mapping samples to the embedding space and querying the corresponding color and opacity values. In our method, the TF is applied after the interpolation of voxel attributes along the viewing rays, following a post-classification approach. The motivation for using post-classification is discussed in Section 2.3.2. By interpolating in the original attribute space, smoother transitions between structures are obtained, resulting in higher visual quality. However, this approach introduces an additional computational cost, as the mapping from high-dimensional attributes to the embedding space must be performed for each sample.

4.3. Rendering Pipelines

After defining the TF in the embedding space, the final step is to integrate this TF into the DVR process. In DVR, samples are taken along viewing rays through the volumetric data, and for each sample, color and opacity are determined and accumulated using front-to-back compositing.

A key challenge in our method is that the TF is defined in the two-dimensional embedding space, while the volumetric data exists in a high-dimensional attribute space. Therefore, for each sample during rendering, a mapping from the high-dimensional voxel attributes to a position in the embedding space is required in order to evaluate the TF. Since the non-linear dimensionality techniques are non-parametric,

as explained in Section 2.4.3, we need to find other ways to project the samples into the embedding space.

To address this, we adapt two rendering pipelines designed by Snellenberg [4], which differ in how the mapping to the embedding space is performed. The first pipeline, referred to as the full data renderer, finds the landmark that is most similar to each sample in the original attribute space. The low-dimensional embedding position of this landmark is then used to evaluate the TF and obtain the corresponding color and opacity. This approach is computationally more expensive, but generally results in more accurate visualizations.

The second pipeline, referred to as the 2D position renderer, precomputes the mapping. For each voxel, it selects the landmark with the highest influence and stores its 2D embedding position. During rendering, interpolation is then performed directly in the embedding space. This enables significantly faster rendering, but may introduce visual artifacts due to distortions in the embedding. Supporting both pipelines allows users to choose between accuracy and runtime speed depending on the application.

4.3.1. Full Data Renderer

The full data renderer determines the position of each sample in the embedding space during rendering. Since no explicit mapping from the high-dimensional attribute space to the embedding space is available, this mapping is approximated using NN search.

To efficiently perform NN queries, an approximate nearest neighbor (ANN) structure is constructed over the set of landmarks obtained from HSNE. An exact NN search would require comparing each query sample with all landmarks, resulting in a linear complexity of $\mathcal{O}(L)$ per query, where L is the number of landmarks. Since NN queries must be performed for a large number of samples during rendering, this quickly becomes a computational bottleneck. ANN methods reduce this cost by allowing a small approximation error in exchange for significantly faster query times, making them more suitable for interactive rendering scenarios.

In our implementation, we use the Hierarchical Navigable Small World (HNSW) index to build the ANN on the CPU. The HNSW index is chosen because it provides fast query times with high accuracy. HNSW constructs a graph-based structure in which the nodes are connected based on proximity, enabling efficient navigation through the search space. In practice, this results in near-constant time query performance while maintaining high-quality NN approximations. Other ANN structures, such as tree-based methods, often degrade in performance in high-dimensional spaces, making them less suitable for our use case.

The ANN structure is constructed on the CPU rather than the GPU. Although GPU-based ANN methods can offer higher throughput, they typically require transferring data between CPU and GPU memory, which introduces additional overhead. Since our rendering pipeline already involves CPU-side preprocessing and relatively small ANN structures due to the use of landmarks, a CPU-based implementation provides sufficient performance while keeping the system design simpler.

The ANN index is constructed using the high-dimensional attribute vectors of the landmarks and similarity is measured using the Euclidean distance. This choice is motivated by its simplicity and efficiency, as the Euclidean distance can be computed quickly and is widely supported by ANN libraries. However, it is important to note that this distance measure differs from the similarity definition used in HSNE. HSNE constructs neighborhood relationships based on transition probabilities derived from random walks on a similarity graph [7]. In contrast, our ANN structure relies on Euclidean distances in the attribute space. This introduces an approximation, as the notion of similarity used during rendering does not fully match the one used during the construction of the HSNE hierarchy. However, this simplification is necessary to enable efficient NN queries and is consistent with the approach used in the original method.

The use of HSNE significantly reduces the number of points in the ANN structure, since only landmarks are considered instead of all voxels. Let N be the number of voxels and L the number of landmarks. The query time of the ANN structure for a t-SNE approach is approximately $\mathcal{O}(\log N)$, while for HSNE it is approximately $\mathcal{O}(\log L)$, with $L \ll N$. This results in a substantial speed improvement during run-time.

Due to memory limits, the rendering process is divided into batches. Instead of processing all samples at once, groups of samples are processed sequentially. This reduces memory overhead. The batching strategy follows the approach described by Snellenberg [4], and we refer to that work for a more detailed discussion.

Within each batch, the position of each sample in the embedding space is computed using the ANN structure. For each sample with attribute vector x, a k-NN query is performed in the high-dimensional attribute space to identify the k closest landmarks. This is formulated as retrieving the set of landmarks that minimizes the Euclidean distance:

$$\mathcal{N}_k(x) = \underset{\substack{\mathcal{S}\subseteq\{x_1,x_2,\ldots,x_L\}\\|\mathcal{S}|=k}}{\arg\min} \sum_{x_i\in\mathcal{S}} d(x,x_i), \tag{4.1}$$

where $\mathcal{N}_k(x)$ denotes the set of k nearest landmarks, x_i represents the attribute vector of landmark i, S is the candidate subset of landmarks, L is the total number of landmarks, and $d(x,x_i)$ is the Euclidean distance, defined as:

$$d(x,x_i) = \sqrt{\sum_{j=1}^{D}(x_j - x_{i,j})^2}, \tag{4.2}$$

where D denotes the dimensionality of the attribute space, j represents the attribute index, x_j represents the j-th attribute of the sample vector x, and $x_{i,j}$ is the j-th attribute of the landmark vector x_i.

In our implementation, we only use $k = 1$. Snellenberg [4] compared $k = 1$ and $k = 9$ and showed that using $k = 9$ only increases the run-time without improving the quality of the final rendered image. Therefore, we use only the single nearest landmark. This means that the embedding position of the nearest landmark is directly assigned to the sample. This approach is computationally efficient and preserves sharp transitions, but may introduce noise or discontinuities due to the discrete assignment.

Because the mapping is based on only a single nearest landmark, small changes in the interpolated attribute values may cause a sample to switch to a different landmark. Although neighboring samples in the high-dimensional attribute space are generally expected to remain close in the embedding, this is not always guaranteed in non-linear DR methods. As a result, neighboring samples along a viewing ray may occasionally map to noticeably different positions in the embedding space, which can introduce local inconsistencies or visual artifacts in the rendered image.

The resulting embedding positions are then passed to the rendering stage. The TF is evaluated at the resulting embedding position to obtain color and opacity values, which are accumulated using front-to-back compositing to produce the final image.

4.3.2. 2D Position Renderer

The 2D position renderer avoids NN queries during rendering by precomputing a mapping from voxel attributes to positions in the embedding space. Instead of determining the embedding position at run-time, we assign a fixed position in the two-dimensional embedding to each voxel during a preprocessing step.

We assign a landmark from the selected HSNE hierarchy to each voxel. To determine the assignment, we use an existing function provided by the HSNE framework. This function computes, for each voxel, the influence of landmarks across the hierarchy levels. The influence values represent how strongly a landmark contributes to the representation of a voxel, based on the probabilistic relationships defined during the HSNE construction.

For each voxel, we retrieve the influence values at the selected hierarchy level and identify the landmark with the highest influence. The voxel is then assigned to this landmark and the corresponding two-dimensional embedding position of that landmark is stored. This results in a single embedding coordinate (x, y) per voxel.

To improve efficiency, we use a minimum influence threshold of 0.05 when retrieving the influence values. Landmarks that have an influence below this threshold on the voxel are ignored. This reduces the number of landmarks that need to be considered per voxel, as many landmarks have negligible contribution. As a result, both computation time and memory usage are reduced.

In our implementation, each voxel is assigned to a single landmark based on the maximum influence. This results in a discrete mapping from voxels to embedding positions, which is simple to compute and efficient to store.

An alternative approach would be to use a weighted combination of multiple landmarks, where the final embedding position is computed as a weighted average based on the influence values. This could better capture smooth transitions between regions and reduce discontinuities in the mapping. However, averaging multiple landmark positions may also blur the separation between clusters in the embedding space, especially near boundaries between different structures. In addition, storing and processing multiple landmark associations per voxel increases the complexity of the preprocessing pipeline and the memory required for the precomputed mapping. Since the 2D position renderer is primarily intended for simple and interactive visualization, we use only the single most influential landmark for each voxel.

The chosen approach represents a trade-off between simplicity and accuracy. By selecting only the most influential landmark, the preprocessing step remains efficient, and the resulting data representation is compact.

During rendering, interpolation is performed directly on the precomputed embedding positions instead of the original voxel attributes. For each sample along a viewing ray, the embedding positions of neighboring voxels are interpolated, and the resulting position is used to query the TF. This design removes the need for NN queries during rendering, significantly improving performance and enabling near-instant interaction.

A major limitation of the 2D position renderer is that interpolation is performed in the embedding space instead of in the original attribute space. In the full data renderer, each sampling position is first reconstructed in the high-dimensional attribute space. The nearest landmark is then found based on this reconstructed attribute vector. This means that the mapping to the TF is based on the actual interpolated data value.

The 2D position renderer works differently. Here, each voxel is first assigned a fixed 2D embedding position during preprocessing. During rendering, these 2D positions are interpolated directly. However, the embedding is a non-linear projection of the original high-dimensional attribute space. Therefore, a straight line between two points in the embedding does not necessarily represent a meaningful transition between the corresponding voxel attributes. As a result, interpolation in the embedding space may create artificial intermediate positions that do not correspond to valid or meaningful data samples.

This can lead to visual artifacts. For example, two neighboring voxels may have embedding positions on different sides of the TF domain or in different clusters. Interpolating between their 2D positions can pass through regions of the embedding that do not represent the actual transition in the original attribute space. If these intermediate regions have different color or opacity values, the renderer may assign optical properties that would not be selected by the full data renderer. An example of this problem is shown in Figure 4.4.

This limitation is the main trade-off of the 2D position renderer. By precomputing and interpolating 2D positions, the renderer avoids expensive NN queries during rendering and becomes much faster. However, this speedup comes at the cost of a less reliable mapping between interpolated samples and the TF domain. Therefore, the 2D position renderer is mainly useful when interaction speed is more important than preserving the most accurate relation to the original high-dimensional data.

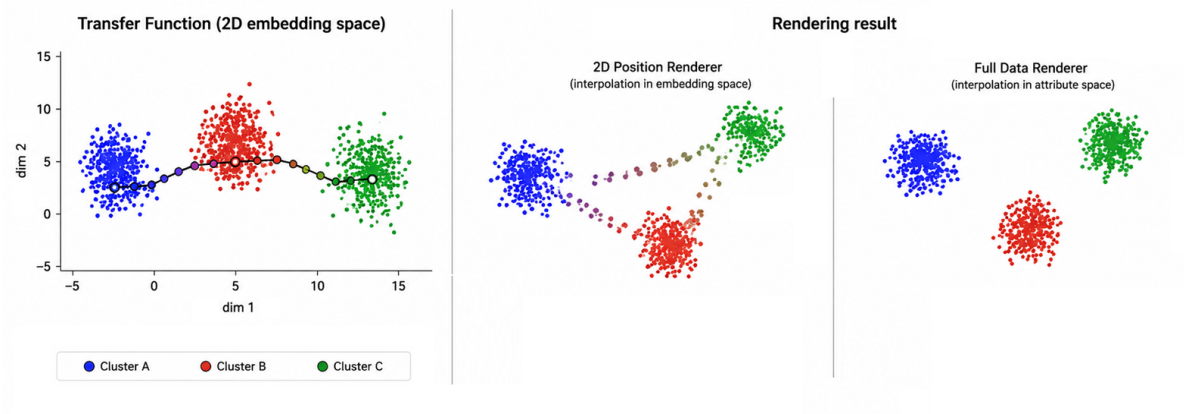


Figure 4.4: Illustration of interpolation artifacts in the 2D position renderer. The left image shows a TF defined in a 2D embedding space with three separated clusters. The middle image shows the result of interpolating directly between embedding positions, as done by the 2D position renderer. This interpolation can create artificial intermediate positions between unrelated clusters, which may lead to incorrect color and opacity values. The right image shows the full data renderer, where interpolation is performed in the original attribute space before mapping the sample to the embedding. This better preserves the intended separation between structures.

5 Results

This chapter presents the evaluation of our proposed hierarchical stochastic neighbor embedding (HSNE)-based transfer function (TF) approach. The goal of the evaluation is to analyze whether the hierarchical landmark representation improves scalability and usability compared to a flat t-SNE embedding, and how this affects the final rendered result. First, Section 5.1 describes the datasets, preprocessing steps, TF setup, rendering configuration, hardware, and evaluation metrics used throughout the experiments. Next, Section 5.2 compares preprocessing and rendering runtime between t-SNE and different HSNE hierarchy levels. Section 5.3 then evaluates how the different embeddings affect TF interaction and visual clutter. Finally, Section 5.4 compares the visual quality of the resulting renderings and analyzes the trade-offs between the full data renderer and the 2D position renderer.

5.1. Experimental Setup

This section describes the experimental setup used to evaluate our proposed method. It outlines the dataset, preprocessing steps, TF configuration, rendering settings, hardware, and evaluation metrics used in the experiments.

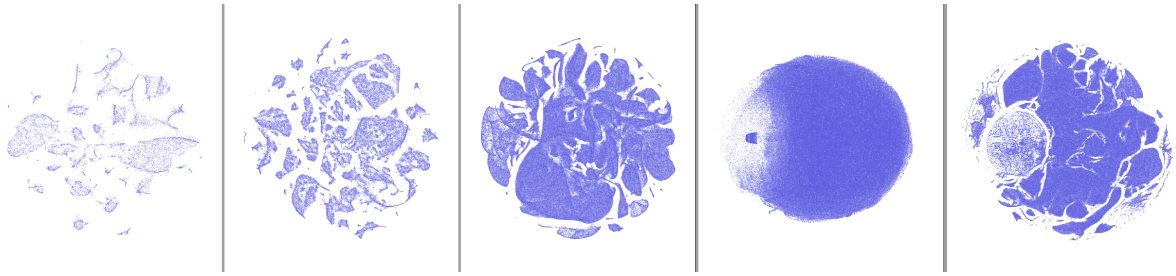
Dataset

We make use of a tissue dataset [42], which was also used and preprocessed by Snellenberg [4]. Using the same dataset allows for a direct and fair comparison with previous work. Two subsets have been derived from this dataset at different resolution levels. The first set, hereafter called Dataset 1, is a high-resolution zoom in and contains $194 \times 175 \times 175$ voxels, resulting in a total of 5,941,250 data points. The second set, hereafter called Dataset 2, has a lower resolution, is more zoomed out, and contains $162 \times 162 \times 194$ voxels, resulting in a total of 5,091,336 data points. Both sets can be seen in Figure 5.1. Each voxel in both sets is described by 26 attributes, representing different measured properties of the tissue.

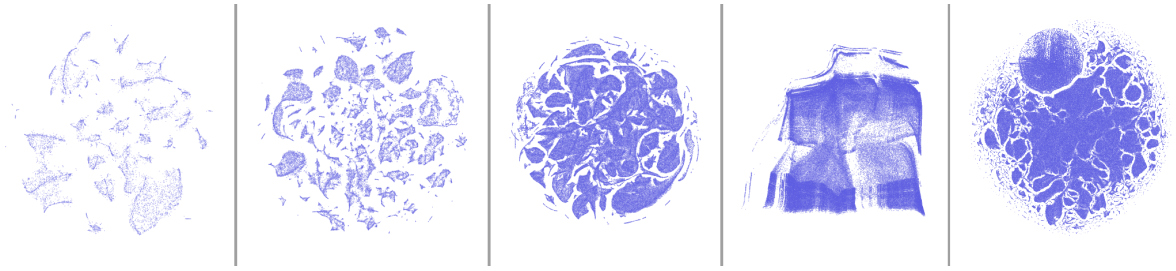
(a) Dataset 1: high-resolution zoom in

(b) Dataset 2: low-resolution zoom out

Figure 5.1: Two subsets derived from the tissue dataset [42].



(a) All DR techniques applied to the zoomed-in dataset.



(b) All DR techniques applied to the zoomed-out dataset.

Figure 5.2: Comparison of the DR embeddings for the zoomed-in (a) and zoomed-out (b) datasets. From left to right for both datasets: the embeddings of the four levels of the HSNE hierarchy, followed by the t-SNE embedding on the far right, illustrating the progressive increase in detail across the hierarchy levels and the visual differences between hierarchical and flat embeddings.

This dataset is well suited for our experiments, as it is both high-dimensional and large-scale, which makes it challenging for traditional dimensionality reduction (DR) and rendering techniques. As a result, it provides a meaningful benchmark for evaluating scalability and visual quality.

Preprocessing

Before rendering, we computed both HSNE and t-SNE embeddings using the same input data to ensure a fair comparison. We performed all computations and renderings within the ManiVault framework, using the existing DR method plugins available in the system. Using the same environment for all methods ensures consistency in the processing pipeline and avoids differences caused by external implementations.

For t-SNE, we use the standard implementation with a perplexity value of 30, which is a commonly used setting that balances local and global structure preservation. We compute the embedding using Euclidean distance, consistent with the preprocessing used for HSNE.

For HSNE, we construct a four-level hierarchy using Euclidean distance in the high-dimensional attribute space. We chose Euclidean distance because it is consistent with both the ANN queries used during rendering and the t-SNE baseline, ensuring that all methods are comparable. The neighborhood graph is constructed using HNSW for the k-NN algorithm, with 90 nearest neighbors (NN) per point. This number of neighbors helps capture local structures in the data and improves the stability of the hierarchy. The choice of four levels provides a range of abstraction from detailed to coarse representations, which is sufficient to study the influence of abstraction level on TF design and rendering results.

We only use three of the four levels of both HSNE hierarchies and exclude the lowest levels. As shown in Figure 5.2, the lowest levels do not show any good clusters and have a lot of overlap and visual clutter. So both levels are not very useful. Also, the lowest levels use all of the voxels, which would remove the benefits of the hierarchical representation, since the number of points becomes all of the data points again.

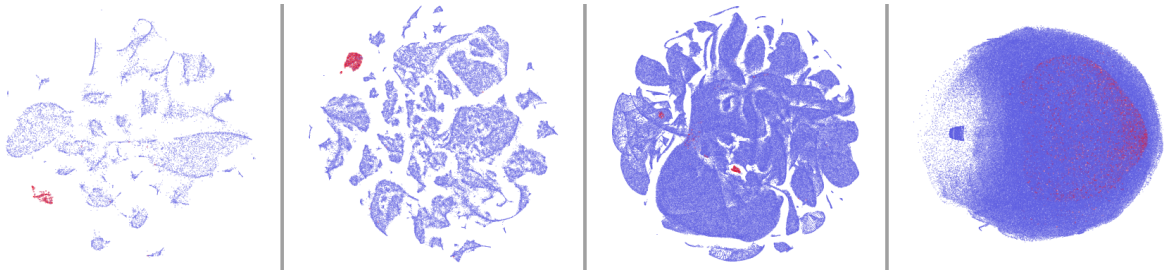


Figure 5.3: Four-level HSNE hierarchy of the tissue dataset. The highest abstraction level is shown on the left, with progressively finer levels towards the right. Some selected data points are highlighted in red at each level. We use this selection mechanism to ensure consistent TF definition across different abstraction levels.

TF Setup

To ensure a fair comparison between different rendering methods, we use the same TF settings where possible. We define the TF in the embedding space by selecting regions and assigning color and opacity values. These selected regions determine which parts of the data become visible in the final rendering.

For qualitative comparisons that involve manually defining TF widgets in different embeddings, we make use of the interactive selection mechanism available in the system. The embedding highlights selected data points, allowing the user to identify corresponding regions across different hierarchy levels, as shown in Figure 5.3. These highlights can be used to create comparable TF widgets across different embeddings, so similar structures can be visualized consistently across renderings. It should be noted that this process involves user interaction and is not fully automated. However, we took care to select comparable regions across all experiments to maintain consistency.

For the visual quality comparison between t-SNE and HSNE hierarchy levels in Section 5.4.1, we use a more controlled setup. Instead of manually recreating similar TF widgets for each HSNE level, we define the TF once in the t-SNE embedding and reuse this TF for all HSNE renderings. For each HSNE level, we link the landmarks to their corresponding original data points and assign the color and opacity values from the t-SNE-defined TF to these landmarks. During rendering, we map each sample to its nearest HSNE landmark in the original high-dimensional attribute space using $k = 1$, as explained in Section 4.3.1, and then use the propagated color and opacity values. This setup compares the t-SNE and HSNE renderings using the same TF definition, so visual differences can be attributed to the HSNE abstraction level and landmark representation rather than to manually different TF selections.

Rendering Setup

We performed all rendering experiments using the same rendering configuration to ensure comparability. The output resolution is the same, and we used the same sampling step size of 0.5 for all methods. Color and opacity values are accumulated using front-to-back compositing, as described in Section 2.2.1. By keeping these parameters constant, differences in the results can be attributed to the DR and rendering pipeline rather than to variations in rendering settings.

Hardware

We performed all experiments on a system equipped with an AMD Ryzen 7 6800H 8-core CPU running at 3.20 GHz and 24 GB of RAM. The system also includes an NVIDIA GeForce RTX 3070 Laptop GPU with 8 GB of VRAM.

Evaluation Metrics

We evaluated the proposed method based on runtime speed, usability, and visual quality.

The runtime evaluation measures preprocessing time and rendering time. Preprocessing time includes the construction of the ANN tree, but excludes the processing time of the DR technique itself. Rendering time measures the time required to generate an image. Together, these metrics provide insight into the scalability and efficiency of the different approaches.

The usability evaluation consists of a qualitative analysis of the TF design process in the embedding space. Since TF creation in our system relies on direct user interaction with the embedding, the usability of a representation depends on how easily users can select meaningful structures. The analysis focuses on the level of visual clutter in the embedding, the precision required to define selections, the number and size of TF widgets needed to isolate structures, and the ease of reproducing consistent TF definitions across different embeddings.

The visual quality evaluation compares the rendered images qualitatively. It focuses on the clarity of structures, the presence of noise or artifacts, and the ability to distinguish meaningful regions in the data. Although quantitative image quality metrics could be used, qualitative analysis better matches the goal of the system, which is to support user-driven exploration and interpretation of the data.

5.2. Preprocessing and Rendering Runtime Comparison

In this experiment, we evaluated the runtime performance of our proposed HSNE-based method in comparison to a baseline approach using t-SNE on the two different sets. The goal is to determine whether replacing t-SNE with HSNE improves rendering efficiency and what the difference is between the HSNE hierarchy levels.

The comparison focuses on the rendering stage. To ensure a fair evaluation, we evaluated both methods using the full data renderer. This renderer maps each sample to the TF domain during rendering, which means that differences in performance can primarily be attributed to the DR technique and the resulting data representation.

All rendering parameters are kept identical to those described in Section 5.1. In particular, the volume is not repositioned after loading, ensuring that all renderings traverse the same volume extent. Furthermore, we did not create any TF widgets during the runtime measurements. As a result, the opacity lookup for each sample remains below the early ray termination threshold, preventing early ray termination from influencing the measured rendering times. This ensures that the reported runtime differences reflect the cost of the rendering pipeline and TF lookup rather than differences in ray traversal length or opacity accumulation. The final images were thus always black.

To obtain reliable measurements, we performed 20 runs. We measured runtime in milliseconds (ms) and report the average runtime. We also computed the standard deviation to assess the stability of the measurements. The evaluation includes both the preprocessing time required to build the ANN graph and the rendering time required to render the image on the screen.

For t-SNE, the ANN graph construction uses the full embedding, meaning that all voxels are included. This results in 5,941,250 points for Dataset 1 and 5,091,336 points for Dataset 2. For HSNE, the ANN graph construction uses only the landmarks of the selected hierarchy level. This means that the number of points is much smaller. For Dataset 1, HSNE level 1 contains 558,509 landmarks, level 2 contains 86,710 landmarks, and level 3 contains 14,940 landmarks. For Dataset 2, these levels contain 428,169; 65,399; and 11,290 landmarks, respectively.

ANN Construction Time

The results, visible in Figure 5.4, show a clear reduction in ANN construction time when using HSNE compared to t-SNE for both datasets. The t-SNE bars are much higher than the HSNE bars. We expected this behavior because the ANN structure uses only the HSNE landmarks instead of all the voxels in the dataset.

For Dataset 1, the ANN construction time decreases from 349,0 s for t-SNE to 15,0 s for HSNE level 1, 4,4 s for HSNE level 2, and 3,3 s for HSNE level 3. We can see a similar trend for Dataset 2, where t-SNE requires 257,8 s, while HSNE levels 1, 2, and 3 reduce the construction time to 11,2 s, 3,6 s, and 2,8 s, respectively.

The results suggest that ANN construction time mainly follows the size of the point set used to build the graph. We do not claim an exact complexity from these measurements, since only a limited number of dataset sizes and hierarchy levels are evaluated. However, the observed trend is clear: when the number of embedding points decreases, the ANN construction time also decreases strongly.

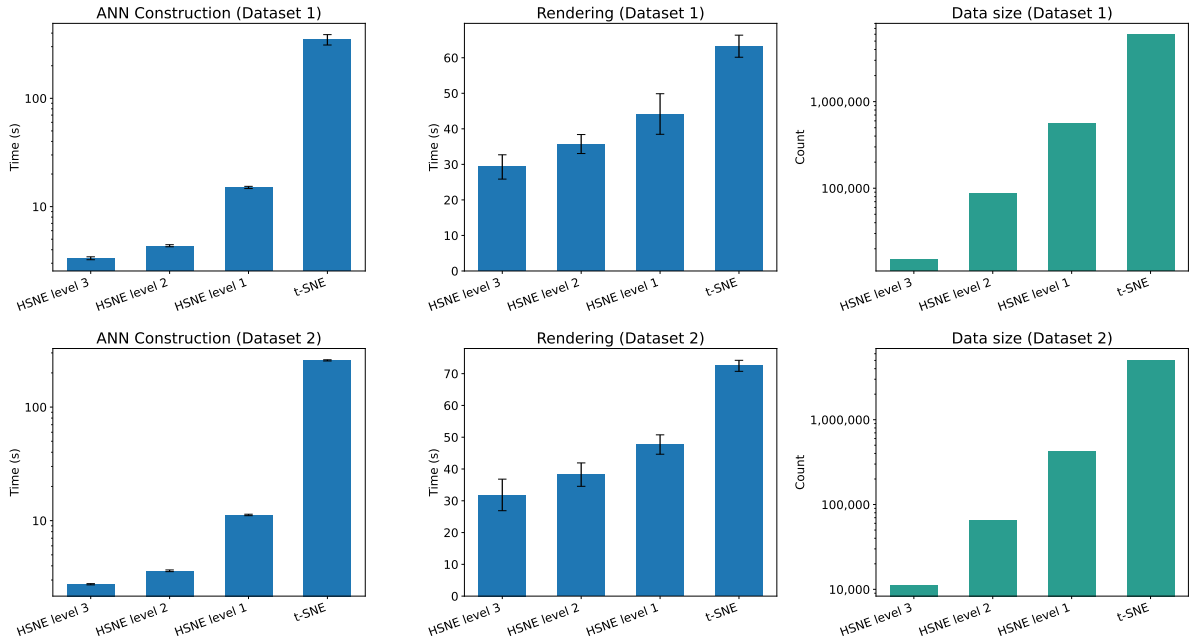


Figure 5.4: Runtime speed comparison between the proposed HSNE-based approach and the baseline t-SNE method for two volumetric datasets. The left column shows the ANN graph construction time, the middle column shows the rendering time, and the right column shows the amount of points in the embedding. Results are reported for three HSNE hierarchy levels and for t-SNE using the full embedding. Each blue bar represents the average runtime over 20 runs, with error bars indicating the standard deviation. The results demonstrate that HSNE significantly reduces both preprocessing and rendering time compared to t-SNE, with higher hierarchy levels providing additional speed improvements due to the reduced number of landmarks.

This explains the large difference between t-SNE and HSNE. For t-SNE, the ANN structure uses all voxels in the dataset, so the index contains more than five million points. For HSNE, the ANN graph uses only the landmarks of the selected hierarchy level. Since higher HSNE levels contain fewer landmarks, the graph becomes smaller and is faster to construct. We can see for both datasets that level 3 is the fastest HSNE level, followed by level 2 and level 1. This confirms that the hierarchical representation improves scalability, especially for large volumetric datasets.

The standard deviation is relatively small for all methods, which means that the ANN construction time is stable across the 20 runs. The t-SNE baseline also has low variance, but its absolute construction time remains much higher. Therefore, the main difference is not caused by unstable measurements, but by the much larger ANN structure required for the full t-SNE embedding.

Rendering Speed

The rendering measurements, visible in Figure 5.4, show that HSNE is consistently faster than t-SNE for both datasets. However, the reduction in rendering time is much smaller than the reduction observed for ANN construction. While the ANN construction time decreases by several orders of magnitude when moving from the full t-SNE embedding to the coarser HSNE levels, the rendering time is reduced by roughly a factor of two.

In Figure 5.4, we see that for Dataset 1, the full t-SNE renderer takes 63.25 s, while HSNE level 3 reduces this to 29.27 s. This means that the coarsest HSNE level needs slightly less than half of the t-SNE rendering time. A similar trend can be seen for Dataset 2, where the rendering time decreases from 72.46 s for t-SNE to 31.84 s for HSNE level 3. The intermediate HSNE levels follow the same pattern, with rendering time decreasing as the hierarchy level becomes coarser.

This pattern shows that the number of points in the ANN graph still influences rendering performance. During rendering, each sample has to be mapped from the high-dimensional attribute space to the embedding space using an ANN query. Since HSNE uses only landmarks instead of all voxels, these queries are performed on a smaller graph, which reduces the lookup cost.

At the same time, the improvement is less extreme than for ANN construction. This is because rendering time is not only determined by the ANN lookup. The renderer still has to cast rays through the same volume, take samples along those rays, interpolate the data, evaluate the TF, and perform front-to-back compositing. These steps are mostly the same for t-SNE and HSNE, because the rendering settings and camera setup are kept fixed. Therefore, reducing the ANN size improves only part of the total rendering pipeline, while the remaining rendering work stays similar.

The hierarchy level again has a clear influence on performance. Higher HSNE levels contain fewer landmarks, which makes the ANN search cheaper and reduces the total rendering time. HSNE level 3 is therefore the fastest configuration for both datasets.

Compared to the ANN construction measurements, the rendering measurements show larger standard deviations. This is expected, because rendering involves more operations than only building the ANN graph, including ray traversal, sampling, interpolation, and repeated ANN queries. Still, the overall trend remains clear for both datasets: HSNE consistently reduces rendering time compared to t-SNE, with the coarsest HSNE level reaching approximately half of the t-SNE runtime.

Comparison Between Datasets

Although both datasets exhibit the same overall behavior, Dataset 2 consistently produces slightly lower ANN construction times but somewhat higher rendering times than Dataset 1. This can be explained by the fact that Dataset 1 contains a few more voxels than Dataset 2; thus, the construction time for Dataset 1 is a bit longer. However, the volume of the space that the rays have to traverse for Dataset 2 is a bit larger than that for Dataset 1, resulting in slightly longer rendering times overall.

Despite these differences, the relative performance improvements achieved by HSNE remain consistent across datasets, indicating that the proposed method generalizes well to different volumetric data characteristics.

Experimental Result

Overall, the experiment shows that replacing t-SNE with HSNE improves both preprocessing and rendering speed, but the size of the improvement differs between the two stages. The largest improvement is found during ANN construction. This is expected, because this stage directly depends on the number of points that are inserted into the ANN graph. For t-SNE, this means that all voxels in the dataset have to be added. For HSNE, only the landmarks of the selected hierarchy level are used. Since this reduces the point set from millions of voxels to a much smaller number of landmarks, the construction time decreases strongly.

The rendering stage also benefits from the smaller ANN structure, but the improvement is less extreme. This is because rendering does not only consist of NN lookup. The renderer still has to traverse the same volume, sample along each ray, interpolate the data, evaluate the TF, and perform front-to-back compositing. These operations remain mostly the same for t-SNE and HSNE, since the same rendering settings are used for all experiments. Therefore, HSNE reduces the cost of the ANN lookup part of the rendering pipeline, but not the full rendering cost. This explains why the rendering time is roughly halved for the coarser HSNE levels, while the ANN construction time decreases much more strongly.

These results confirm that HSNE is more scalable than a flat t-SNE embedding for this rendering setup. The advantage of HSNE is not only that it gives faster preprocessing, but also that it changes the size of the data representation used during rendering. By using landmarks instead of all voxels, HSNE reduces the computational cost of the ANN-based mapping step while still allowing the TF to be defined in a two-dimensional embedding space. This makes the hierarchical representation more suitable for large multivariate volumetric datasets, especially when interactive performance is important.

5.3. TF Interaction Comparison

Besides runtime performance, the usability of the embedding as a TF domain is an important aspect of DR-based volume visualization. A useful embedding should allow users to identify meaningful structures and create selections with limited effort. In this experiment, we qualitatively compare the interaction characteristics of t-SNE and HSNE embeddings and analyze how different HSNE hierarchy levels influence TF design.

Embedding	Visual clutter	Time to reproduce	Ease of isolating structures	Number of widgets	Size of widget	Selection precision
HSNE Level 3	Very Low	Very Low	Easy	Few	Large	Low
HSNE Level 2	Low	Low	Moderate	Moderate	Medium	Medium
HSNE Level 1	Moderate	Moderate	Difficult	Many	Small	High
t-SNE	Very High	High	Difficult	Many	Small	High

Table 5.1: Qualitative comparison of TF interaction characteristics for the different embeddings.

The evaluation focuses on the visual clutter of the embedding, the ease of selecting meaningful regions, and the number and precision of TF selections required to isolate structures. We perform a qualitative evaluation because DR-based TF design involves user interaction. However, a formal user study is not conducted in this work. Consequently, the observations are based on a qualitative analysis of the interaction process and the resulting visualizations. A more extensive user evaluation could be considered in future work to further assess usability and interaction effectiveness.

To perform the comparison, we created TFs on the t-SNE embedding and on the first three levels of the HSNE hierarchy for both datasets. For each dataset, we selected one or more representative structures in a reference embedding, after which we identified comparable regions in the other embeddings to obtain visually similar renderings. This approach allowed us to compare how different embeddings and hierarchy levels influence the complexity of TF design while maintaining comparable visualization goals across the experiments.

During TF creation, we placed selections primarily on clusters corresponding to meaningful structures in the rendered volume rather than on sparse background regions or noise. This ensured that the comparison focused on the ability of the embeddings to represent and isolate relevant structures within the data.

Figure 5.5 and Figure 5.6 show the TF interaction results for the different HSNE hierarchy levels and the t-SNE embedding. The figures include both the embedding used for TF design and the resulting rendering produced using the defined TF.

A summary of the qualitative observations is in Table 5.1. The table compares the different embeddings in terms of visual clutter, interaction complexity, preservation of structural detail, and the number of TF widgets required to isolate meaningful regions.

Visual Clutter

The t-SNE embeddings in both datasets contain all data points and therefore show significant visual clutter, as visible in Figure 5.5 and Figure 5.6. The embeddings become highly dense, with many overlapping and fragmented structures. The high point density also reduces the readability of the embeddings, making it more difficult to visually identify coherent structures.

In contrast, the HSNE hierarchy levels progressively reduce the number of represented points by replacing groups of voxels with landmarks. At HSNE level 1, the embedding already becomes noticeably less dense than the t-SNE embedding while still preserving much of the local structure. Compared to t-SNE, the structures appear more coherent. However, the embedding still contains a relatively high amount of detail, and several structures remain partially overlapping.

At HSNE level 2, the reduction in visual clutter becomes more apparent. The structures in the embedding become more compact and clearly separated, making it easier to identify and isolate meaningful regions. There is little to no overlap between the clusters, which already makes the embedding suitable for distinguishing the different structures.

HSNE level 3 provides the strongest abstraction of the data. The embedding becomes significantly sparser and more visually organized. It is easy to distinguish the different clusters, and therefore the visual clutter is considered very low in Table 5.1.

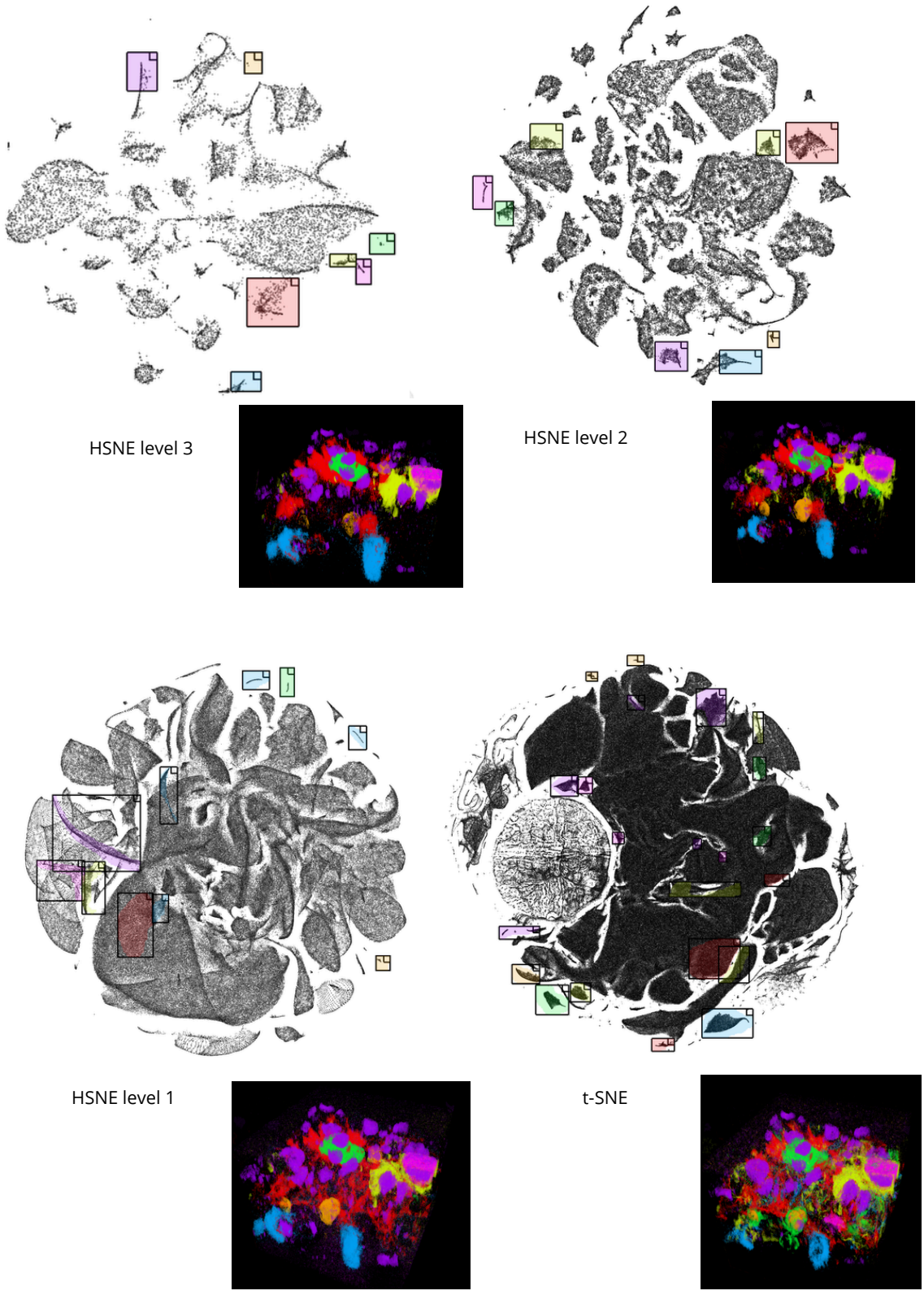


Figure 5.5: Comparison of TF interaction across the three HSNE hierarchy levels and t-SNE for the zoomed-in dataset, together with their corresponding rendered volumes. As the hierarchy level decreases, TF widgets become more numerous, smaller, and require higher precision. The t-SNE embedding exhibits the highest level of visual clutter and the strongest overlap between clusters, making TF design more difficult.

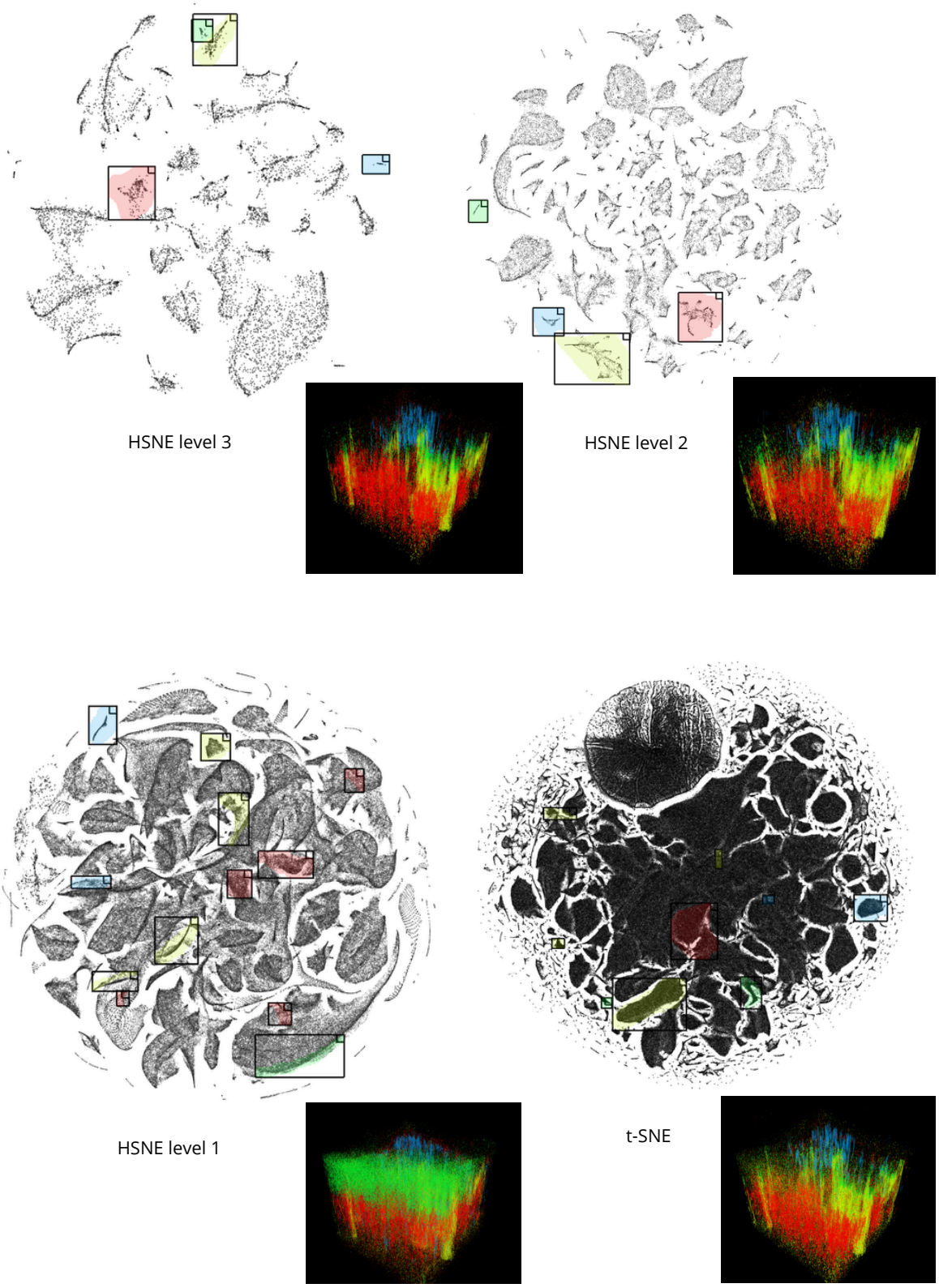


Figure 5.6: Comparison of TF interaction across three HSNE hierarchy levels and t-SNE for the zoomed-out dataset, together with their corresponding rendered volumes. Higher HSNE levels require a similar number of TF widgets, while lower levels introduce increasingly more precise widgets. Visual clutter increases progressively from the highest HSNE level to the t-SNE embedding. HSNE level 1 shows more green regions in the rendered volume compared to the other embeddings, as the green TF widget lies at the overlap of two clusters.

TF Selections

In Figure 5.5, we observe that more detail is rendered as the abstraction level decreases toward the t-SNE embedding. This can be explained by the increased precision required to create TF widgets and the overlap between data points. At HSNE level 1, substantially more red regions become visible than at the higher hierarchy levels. This occurs because the red TF widget corresponds to a specific subset of data points located within a larger cluster. Consequently, when creating the widget, additional data points are automatically selected as well. Another example can be seen in the t-SNE embedding, where even more voxels are rendered. In particular, more green regions become visible because three small clusters required TF definitions within a larger cluster. As a result, it is not possible to isolate only the intended data points using the TF widgets.

Figure 5.6 demonstrates a different behavior. We observe that HSNE levels 2 and 3 require the same number of TF widgets, while HSNE level 1 requires significantly more widgets and the t-SNE embedding requires only a couple more. This occurs because the t-SNE embedding contains many extremely small clusters located within larger clusters. These clusters were so small that it is not possible to create TF widgets that produced meaningful visualizations. Therefore, TF widgets were not created for those regions. Also, in the HSNE level 1 rendering, more green regions are visible. This is because there is a single cluster in the level 1 embedding located within another cluster. Consequently, selecting the desired cluster automatically included surrounding data points as well. In contrast, HSNE levels 2 and 3 contained their own distinct green clusters, while the t-SNE embedding contained one larger selectable cluster, together with several smaller structures that did not receive TF definitions.

In Table 5.1, Figure 5.5, and Figure 5.6, we observe that higher levels of abstraction contain less visual clutter and therefore allow TF definitions to be reproduced more easily, require fewer and larger widgets, and demand less precise selections. However, the increased abstraction also causes smaller structures to merge into larger groups in the embedding, which affects the final visual outcome. More detail on this in Section 5.4.

Another observation is that creating TF widgets is significantly easier in the HSNE level 2 and 3 embeddings than in the HSNE level 1 and t-SNE embeddings. In ManiVault, TF widgets can be created using two interaction modes: rectangle mode, which creates a TF widget over a rectangular region, and lasso mode, which allows users to create highly precise TF selections. In Figure 5.5, all TF widgets in HSNE levels 2 and 3 were created using rectangle mode, whereas the t-SNE and HSNE level 1 embeddings required the use of lasso mode. This demonstrates how visual clutter influences TF design, the precision required for the widgets, and the overall difficulty of isolating structures.

Experimental Result

The experiment demonstrates that increasing the HSNE abstraction level reduces interaction complexity by decreasing visual clutter and simplifying the TF domain, which results in fewer, larger, and less precise TF widgets. Conversely, lower abstraction levels preserve more detail and are better able to separate clusters in high-dimensional space, but they increase interaction complexity due to denser embeddings. The t-SNE embedding represents the extreme case of this trade-off, where all data points are preserved, resulting in the highest level of detail but also the most difficult interaction process.

Overall, the results show that HSNE provides a more manageable and scalable TF domain for large volumetric datasets than t-SNE. The landmark-based representation substantially reduces visual clutter and simplifies interaction while still preserving the dominant structures required for meaningful visualization. Furthermore, the hierarchy levels provide flexibility by allowing users to choose between overview-oriented interaction at coarse levels and more detailed structure exploration at finer levels.

5.4. Visual Quality Comparison

In this section, we evaluate the visual quality of the renderings produced by the proposed HSNE-based TF approach. While the previous sections focused on preprocessing time, rendering performance, and interaction complexity, this section focuses on the final rendered result. The goal is to investigate how the choice of DR representation and rendering pipeline influences the structures that become visible in the volume.

The comparison is qualitative, since there is no direct ground truth TF or visualization for the tissue dataset. Therefore, the renderings are evaluated by visually inspecting the clarity of selected structures, the preservation of fine detail, the amount of noise or clutter, and the ability to distinguish meaningful regions in the volume.

We divided the visual comparison into two parts. First, Section 5.4.1 compares the renderings produced by the different HSNE hierarchy levels and the t-SNE embedding. This experiment evaluates how much visual information is preserved when the t-SNE-defined TF is evaluated through increasingly abstract HSNE landmark representations. Second, Section 5.4.2 compares the full data renderer with the 2D position renderer. This evaluates the visual trade-off between the more accurate full data pipeline and the faster 2D position pipeline.

5.4.1. Comparison of HSNE Hierarchy Levels and t-SNE

In this experiment, we compare the visual results obtained from the different HSNE hierarchy levels with the t-SNE-based rendering. The goal is to evaluate how the abstraction level of the HSNE hierarchy influences the final visualization when the TF itself is kept fixed.

In the interaction comparison shown in Section 5.3, we defined TFs manually in each embedding. This made it possible to evaluate how difficult it is to select meaningful regions in the different TF domains. However, such a setup is less suitable for judging visual quality because differences in the rendered images may be caused either by the DR representation or by small differences in the manually defined TFs. To isolate the effect of the HSNE hierarchy, we therefore used one shared t-SNE TF for all methods in this experiment, as explained in Section 5.1.

This setup allows for a more controlled comparison between t-SNE and HSNE. The t-SNE rendering represents the result obtained when the TF is evaluated using the full flat embedding. The HSNE renderings show how this same TF is approximated when the data is represented by landmarks at different abstraction levels. Therefore, differences between the images indicate the effect of HSNE abstraction and landmark influence rather than differences in TF design.

Figure 5.7 shows the results for Dataset 1. Overall, the main selected structures are preserved across all HSNE levels and in the t-SNE rendering. The large colored structures are visible in all renderings, including the green regions near the top, the cyan structures distributed throughout the lower and central parts of the volume, the red region near the center, and the magenta structure on the right. This shows that the propagated TF remains meaningful across the HSNE hierarchy.

However, several differences between the hierarchy levels are visible. One noticeable difference occurs in HSNE level 2, where part of the structure that appears red in the t-SNE rendering is assigned a cyan color. The landmark-based approximation causes this. During HSNE rendering, we map each sample to the nearest HSNE landmark in the original high-dimensional attribute space. If the nearest landmark corresponds to a point that receives a cyan assignment in the t-SNE TF, the sample will also be rendered cyan, even if nearby samples in the t-SNE rendering are red. This indicates that, at level 2, this part of the data lies close to a landmark with a cyan TF assignment, while the other HSNE levels are closer to a landmark with a red TF assignment. As a result, the boundary between the red and cyan selections differs from the other renderings.

At the same time, the level of abstraction influences the amount of visible detail. In HSNE level 3, the rendering gives the most abstract result. The rendering preserves the main structures, but some of the smaller red, yellow, and green details are less clearly separated. We see this around the central region, where several selected colors are close to each other spatially. At this abstraction level, a single landmark can represent a larger and more mixed group of data points, causing nearby structures with different t-SNE TF assignments to be summarized more strongly.

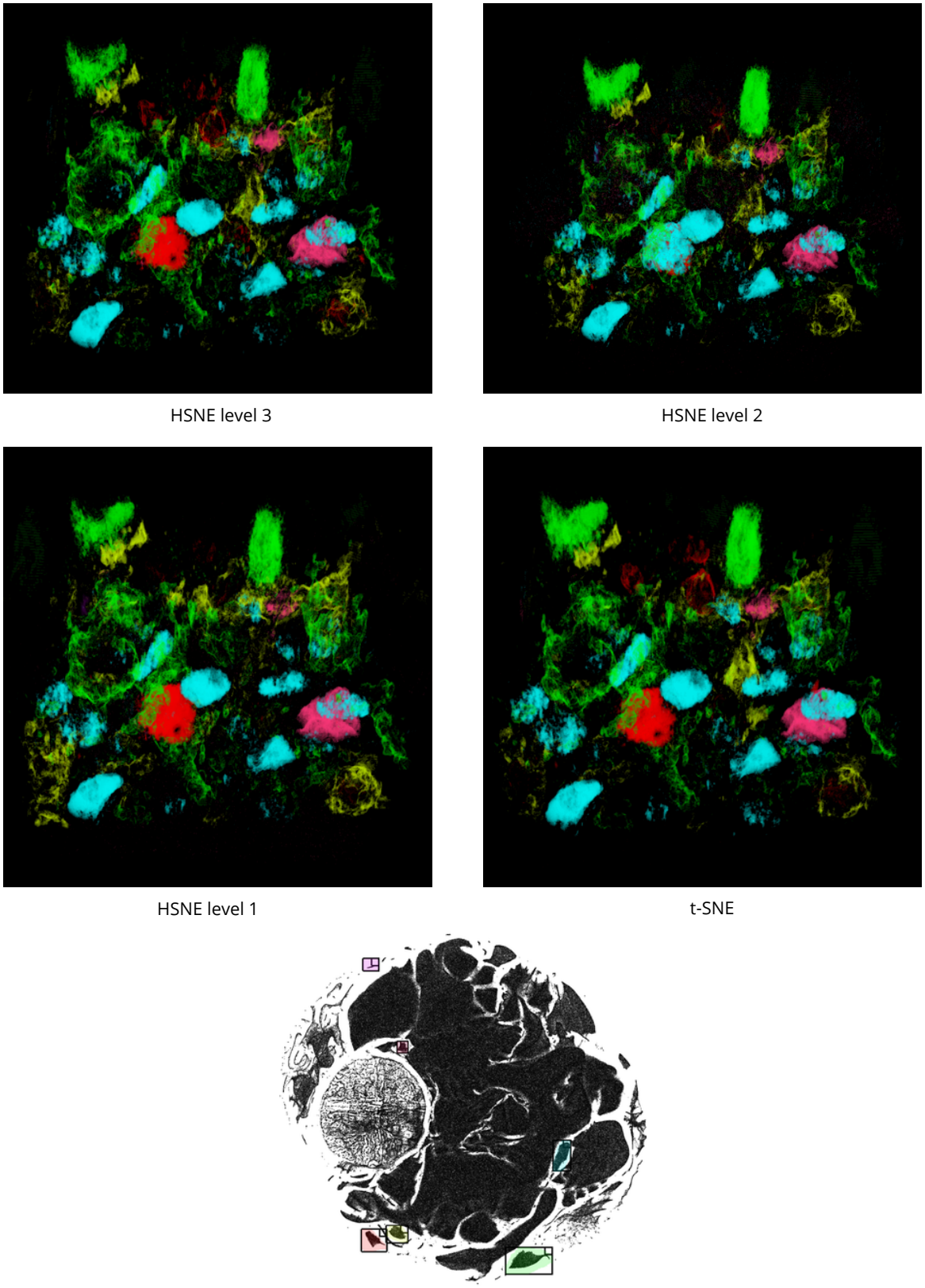
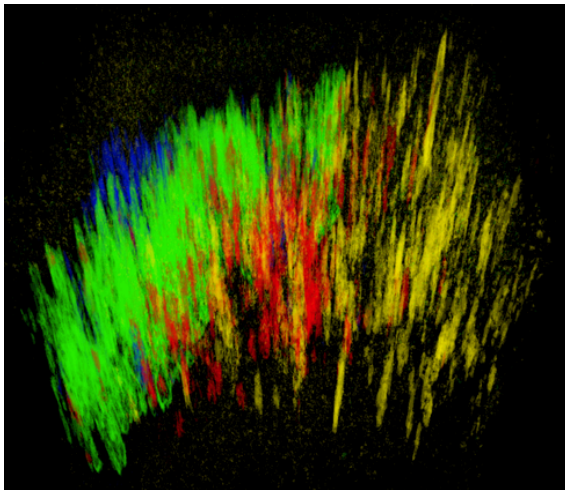
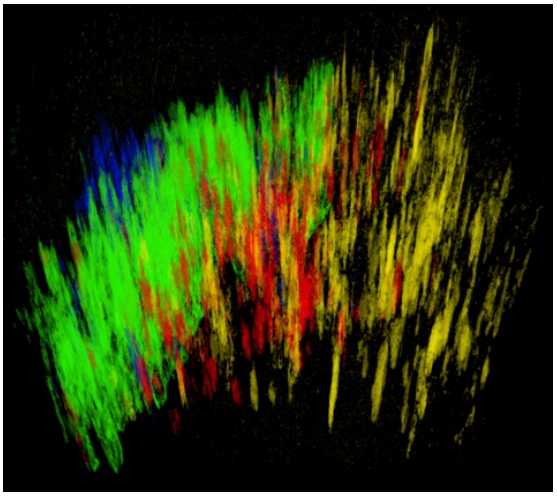


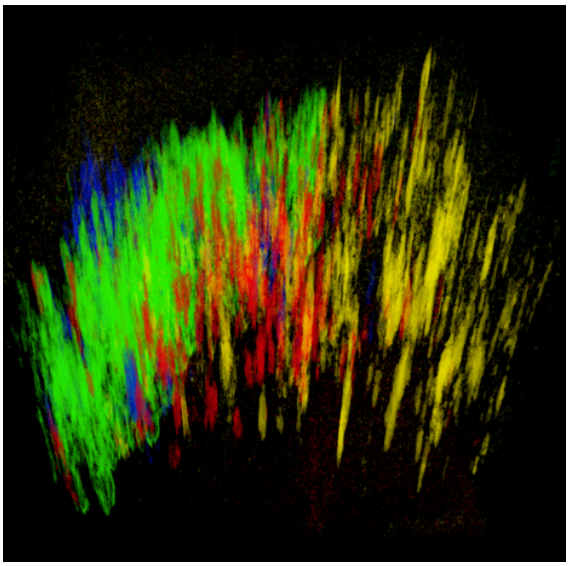
Figure 5.7: Visual comparison of Dataset 1 rendered using the same t-SNE TF for all renderings. The bottom-right image shows the t-SNE rendering, while the other images show the result of propagating this TF to HSNE hierarchy levels 1, 2, and 3. The selected colored structures are largely preserved across the HSNE levels, although local differences in color assignment occur due to nearest-landmark approximation and hierarchy-level abstraction.



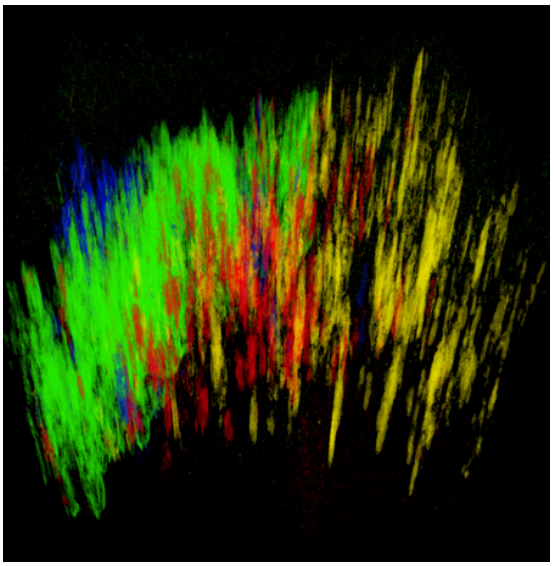
HSNE level 3



HSNE level 2



HSNE level 1



t-SNE

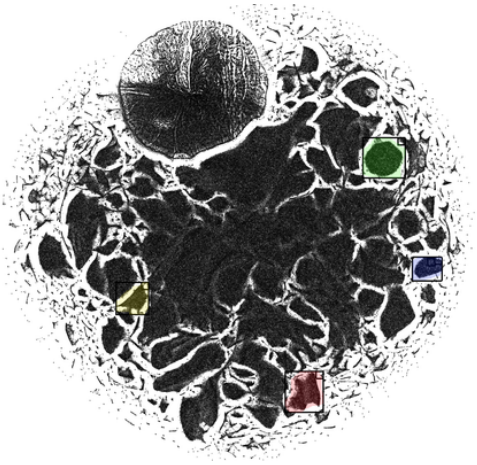


Figure 5.8: Visual comparison of Dataset 2 using a single t-SNE TF. The bottom-right image shows the t-SNE reference rendering, while the other images show the same TF propagated to HSNE hierarchy levels 1, 2, and 3. The main colored structures remain visible across the HSNE levels, showing that the hierarchy preserves the dominant selected regions.

In HSNE level 1, the rendering has more local detail since this level contains more landmarks. Because each landmark represents a smaller subset of the data, the propagated color assignments can follow the local variation in the t-SNE TF more closely. In HSNE level 3, the representation is more abstract, but the dominant regions are still preserved. However, because this level contains fewer landmarks, each landmark represents a larger subset of the original data. As a result, local TF variation can be lost. If a sample is mapped to a landmark whose corresponding t-SNE embedding point does not receive color or opacity, then all samples assigned to that landmark remain transparent, even if some of the underlying data points would have been colored in the full t-SNE rendering. This is more likely to occur at higher abstraction levels, where fewer landmarks are available to represent the variation in the original TF.

Figure 5.8 shows the results for Dataset 2. The same general trend can be observed. The large green region on the left side of the volume, the yellow structures on the right side, and the red structures near the center remain visible in all renderings. This shows that the HSNE landmark representation can preserve the dominant structures selected by the TF, even though the TF is evaluated through a reduced set of landmarks.

The yellow structures on the right side of Dataset 2 in Figure 5.8 also show differences between the levels. In the t-SNE rendering, these structures contain more fine vertical detail. HSNE level 1 preserves more of this fine-scale variation, while levels 2 and 3 show a slightly more compact and less detailed version of the same region. Similarly, the green region on the left is stable across all renderings, but its boundaries become smoother and less locally varied at the higher levels. These observations illustrate the main effect of the HSNE abstraction: the large-scale selected structures are maintained, while smaller color variations and boundary details depend on the hierarchy level.

Experimental Result

The comparison demonstrates that HSNE can reproduce the main visual outcome of a t-SNE-defined TF, even when the TF is evaluated through a reduced set of landmarks. This is important because it shows that the landmark abstraction does not fundamentally change the dominant selected structures in the rendered volume. Instead, it mainly affects the level of detail and the exact placement of color boundaries.

The main trade-off is therefore between visual detail and abstraction. Lower HSNE levels preserve a bit more of the fine structure present in the t-SNE reference, but they also contain more landmarks and introduce more small-scale visual variation. Higher HSNE levels provide a bit cleaner and more compact representation, but they can merge smaller structures and reduce fine detail.

5.4.2. Comparison of Full Data Renderer and 2D Position Renderer

In this experiment, we compare the visual results of the full data renderer with the 2D position renderer. The goal of this comparison is to investigate how the approximation used by the 2D position renderer influences the final rendering. For this comparison, the same HSNE level, TF widgets, colors, opacity values, camera position, and rendering settings are used. The only difference between the two results is the rendering pipeline. This makes it possible to directly compare the visual effect of evaluating the TF in the full data renderer and in the 2D position renderer.

As explained in Section 4.3.1, the full data renderer first interpolates the original high-dimensional voxel attributes at each sampling position. After this interpolation, the nearest landmark is found in the original attribute space using the ANN graph, and the TF is evaluated using the embedding position of this landmark. This means that the TF evaluation is based on the reconstructed data value at the sampling position. This approach is more expensive, because a NN query is required during rendering, but it preserves a stronger relation to the original high-dimensional data.

The 2D position renderer works differently, as explained in Section 4.3.2. Instead of performing NN queries during rendering, each voxel is assigned a precomputed 2D embedding position. During rendering, these 2D positions are interpolated directly. This makes the renderer much faster, but it also introduces problems. Since the embedding is a nonlinear projection of the original attribute space, interpolation in the 2D embedding does not necessarily correspond to a meaningful interpolation in the original data space. As a result, the interpolated position can be within regions of the embedding that

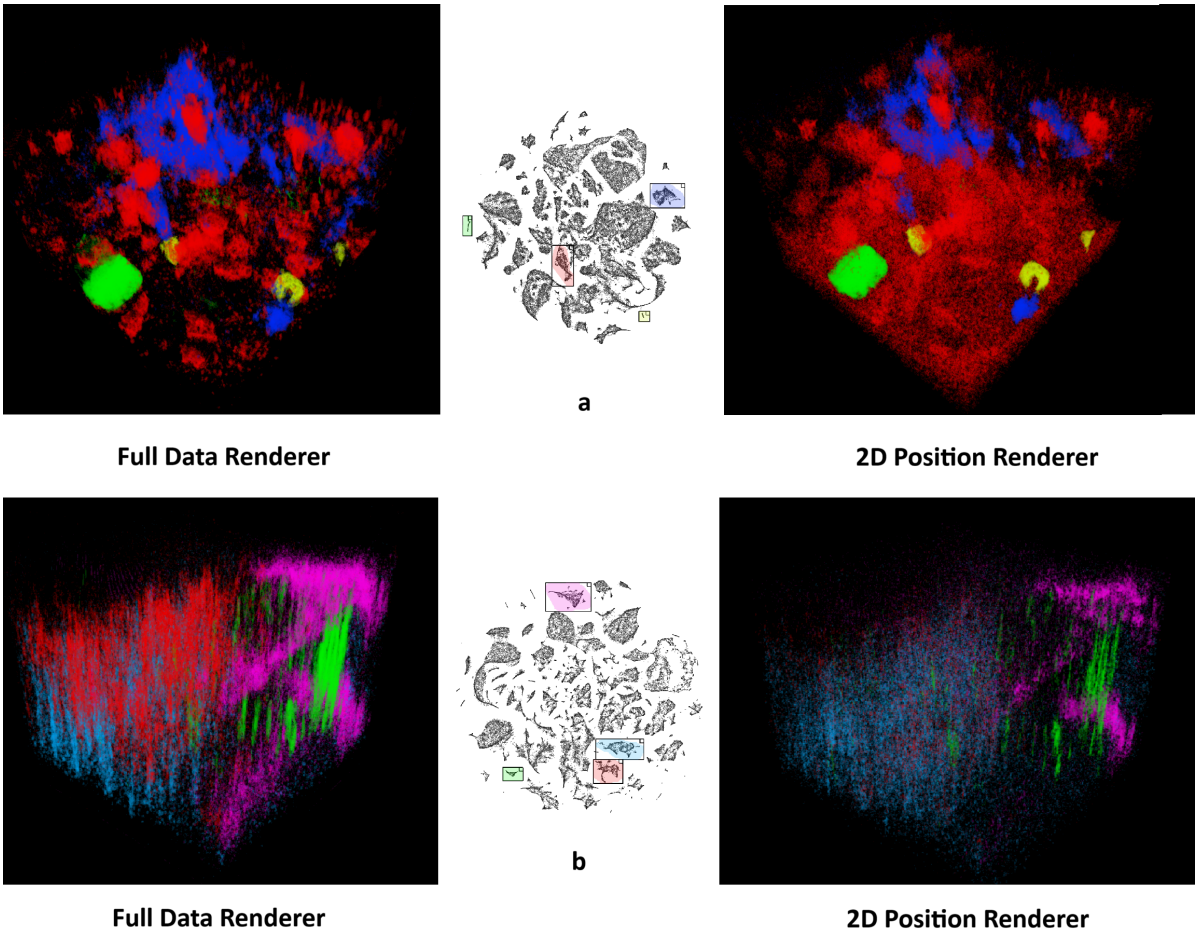


Figure 5.9: Visual comparison between the full data renderer and the 2D position renderer. The same HSNE level, TF widgets, colors, opacity values, camera position, and rendering settings are used for both renderers. (a) shows Dataset 1, where the 2D position renderer introduces clear visual artifacts. In particular, the red selection becomes much more dominant because interpolation in the embedding space can pass through selected TF regions that do not correspond to the actual interpolated high-dimensional data values. (b) shows Dataset 2, where both renderers preserve the same general structures, although the 2D position renderer produces a more compact and less detailed result. This illustrates the trade-off between the more accurate full data renderer and the faster but approximate 2D position renderer.

do not represent the actual data value at that sample position. If a TF widget is located in such a region, the sample can incorrectly receive color and opacity.

This effect can be seen clearly in Figure 5.9a for Dataset 1. The full data renderer shows the selected structures as more separated regions. The red, blue, green, and yellow structures are visible, but the red selection does not dominate the complete volume. In the 2D position renderer, the red color becomes much more widespread and covers a large part of the volume. This indicates that many samples are evaluated inside or near the red TF region after interpolation in the embedding space, even though these samples would not be assigned to the red region by the full data renderer. This is a visual artifact caused by the 2D position approximation.

This problem is more likely to occur when a TF widget is placed in the middle of the embedding, rather than near the edge of the abstraction. When embedding positions are interpolated between neighboring voxels, the interpolated path can pass through the middle of the embedding. If a widget is placed in this central region, it can unintentionally capture interpolated positions that lie between different clusters. These positions do not necessarily correspond to actual landmarks or meaningful data samples. In contrast, widgets near the edge of the embedding are less likely to be crossed by such interpolation paths, because fewer interpolated positions pass through those outer regions. Therefore, central TF widgets are more sensitive to artifacts in the 2D position renderer.

For Dataset 2, shown in Figure 5.9b, the difference between the two renderers is less severe. Both renderers show the same general organization of the volume. The blue region is mainly visible on the left side, the green structures appear more toward the center and right side, and the magenta region remains visible in the upper right part of the volume. However, the 2D position renderer still appears smoother and less detailed in some regions. The full data renderer contains more variation and more mixed structures, while the 2D position renderer gives a more compact result. This suggests that the 2D approximation can work well when the selected regions are more separated or when the interpolation paths do not cross important TF widgets.

Experimental Result

Overall, the comparison shows the main trade-off between the two rendering pipelines. The full data renderer gives a more reliable visual result, because the TF is evaluated after interpolation in the original high-dimensional attribute space. This reduces the chance that samples are assigned to a TF region only because of interpolation artifacts in the embedding. The 2D position renderer is faster and can still preserve the main visual structures, especially for broad and well-separated selections. However, it can introduce incorrect color assignments when interpolation in the embedding space passes through selected regions. This makes the 2D position renderer useful for fast interactive exploration, but less reliable when accurate visual separation of structures is required.

6

Conclusion and Discussion

This thesis presents a hierarchical approach to transfer function (TF) design for multivariate direct volume rendering using Hierarchical Stochastic Neighbor Embedding (HSNE). Our proposed method extends previous dimensionality reduction (DR)-based visualization approaches by replacing flat embeddings such as t-SNE with a hierarchical representation based on landmarks. The goal of this work is to improve the scalability, interaction usability, and rendering performance of DR-based TF design for large multivariate volumetric datasets.

Our proposed system, explained in Chapter 4, integrates HSNE into the visualization pipeline by using a selected hierarchy level as the domain of the TF. Users can interactively define optical properties directly within the low-dimensional embedding, after which the TF is integrated into the rendering process. Two rendering pipelines are adapted: the full data renderer based on nearest neighbor (NN) mapping in the original attribute space and the 2D position renderer, which performs rendering directly in the embedding space using precomputed positions.

The experimental results demonstrate that the use of HSNE substantially improves preprocessing and rendering speed compared to a t-SNE-based baseline, as shown in Section 5.2. Because HSNE represents the data using a reduced set of landmarks rather than all voxels, both ANN construction and rendering times are significantly reduced. Furthermore, higher abstraction levels in the HSNE hierarchy provide additional performance improvements due to the reduced number of landmarks involved in NN queries.

The qualitative interaction experiments further show that HSNE improves the usability of the TF domain for large datasets, as shown in Section 5.3. Compared to t-SNE embeddings, the landmark-based HSNE embeddings contain substantially less visual clutter, making it easier to identify and isolate meaningful regions during TF design. Higher abstraction levels simplify interaction by reducing the number of required TF widgets and lowering the precision needed for selections. Conversely, lower abstraction levels expose more fine-grained structure in the embedding and allow for more detailed TF selections, but they also increase interaction complexity due to denser and more cluttered layouts. The results, therefore, demonstrate a trade-off between interaction simplicity and selection granularity in the TF domain.

The results also highlight that no single hierarchy level is universally optimal. Coarser levels provide cleaner and more manageable embeddings but may group multiple smaller structures into larger landmark regions. Finer levels provide a more detailed TF domain but require more precise and complex interactions. The hierarchy, therefore, provides flexibility, allowing users to choose an abstraction level that best matches their dataset, task, and desired level of detail.

The visual quality evaluation in Section 5.4 shows that the HSNE-based approach can reproduce the dominant visual structures of the t-SNE-based reference, even when the TF is evaluated through a reduced landmark representation. The comparison between hierarchy levels shows that the level of abstraction affects the amount of visual detail in the final rendering. Lower HSNE levels retain more small-scale variation and sharper local differences, while higher abstraction levels provide cleaner and more compact results but may smooth boundaries or merge smaller structures. This visual trade-off is

consistent with the interaction results, where finer levels allow more detailed TF selections but require more complex interaction, whereas coarser levels simplify TF design.

The renderer comparison further shows that the full data renderer produces more reliable visual results, since TF evaluation remains connected to interpolation in the original high-dimensional attribute space. In contrast, the 2D position renderer provides faster rendering and can preserve the main structures for broad and well-separated selections, but they may introduce incorrect color assignments when interpolated embedding positions pass through selected TF regions. Therefore, the visual results show that HSNE improves scalability while maintaining meaningful visual output, but also that the selected hierarchy level and rendering pipeline directly influence the balance between speed, selection precision, visual detail, and rendering reliability.

Overall, the results indicate that HSNE is a suitable alternative to flat DR methods for TF design in large multivariate volumetric datasets. By reducing visual clutter and decreasing computational costs, the hierarchical representation improves both scalability and interaction usability while still preserving the meaningful structures required for exploratory visualization.

6.1. Limitations and Future Work

Several limitations remain in our proposed method. First, the system depends on multiple approximations throughout the pipeline, including the construction of the HSNE hierarchy, the generation of low-dimensional embeddings, and the use of approximate NN queries during rendering. Since the effectiveness of the visualization strongly depends on the quality of the embedding, variations in the HSNE results can affect both the TF design process and the final rendered image. In addition, DR methods such as HSNE are not fully deterministic, meaning that repeated computations may produce slightly different embeddings. This can make interaction less consistent for users and may reduce reproducibility between sessions. The use of approximate NN search further introduces small mapping errors during rendering, which can also influence the visual quality of the final image. Together, these approximations reflect the broader trade-off between rendering accuracy, interaction usability, and computational scalability in large-scale multivariate visualization systems.

Another limitation is the trade-off between scalability and detail preservation introduced by the landmark-based abstraction used in HSNE, as shown in Section 5.3. Higher hierarchy levels reduce visual clutter and improve interaction usability by representing the data with fewer landmarks. However, this abstraction may merge fine-grained structures into larger groups, making it more difficult to isolate smaller features in the data. In contrast, lower hierarchy levels preserve more detail but produce denser embeddings that are more difficult to explore and interact with. As a result, users must choose a hierarchy level that balances detail and usability for their specific dataset and visualization task.

Another important limitation and an important direction for future work is that our current system only uses a single level of the HSNE hierarchy during interaction and rendering. Although HSNE inherently supports multi-scale exploration, dynamic refinement between hierarchy levels is not supported in our proposed method. As a result, users must select a fixed abstraction level beforehand, limiting the ability to seamlessly combine overview and detail during exploration. Prior work, such as CyteGuide [43], shows that users can benefit from interactively navigating between levels to explore regions of interest in more detail. Future work could, therefore, focus on enabling local refinement, where users can select a region in the embedding and transition to a finer hierarchy level to reveal additional structure. The TF definitions should be propagated with this refinement through the hierarchy levels and should be allowed to make local adjustments at the finer scales. This would provide a consistent interaction model while still enabling detailed exploration.

Another limitation and an important direction for future work is improving the mapping between the high-dimensional attribute space and the embedding space during rendering. In the current implementation, the ANN graph described in Section 4.3.1 is constructed using Euclidean distance in the original attribute space. However, the HSNE hierarchy itself is built using probabilistic neighborhood relationships derived from random walks on a similarity graph. As discussed by Pezzotti et al. [7], relying only on Euclidean distance can introduce shortcuts between data points that are close in distance but not truly connected through the underlying data manifold. As a result, samples may be mapped to landmarks that do not accurately reflect the neighborhood structure preserved by HSNE. Future work could

therefore explore ANN methods or similarity measures that better match the probabilistic relationships used during HSNE construction, potentially improving the consistency between the embedding and the rendering process.

Finally, a limitation and a direction for future work is improving the interpretability of embedding-based TF design. Currently, users mainly rely on visible clustering patterns to guide interaction. However, these clusters only indicate that points within the same cluster are similar. The distance between different clusters in the embedding space does not necessarily reflect how similar or different they are in the original high-dimensional space. As a result, it can be difficult for users to identify groups that are distinct but still closely related in the original data. In practice, this often requires significant trial and error during TF creation. Future work could, therefore, explore embedding methods that better preserve relationships between groups in high-dimensional space, such as scvis [44], or incorporate additional information, such as attribute statistics, into the visualization. These additions could help users better understand the meaning of clusters and reduce ambiguity during TF design.

6.2. Reflection on our Work

This work sits in the broader context of scientific visualization and high-dimensional data analysis, where a clear trend is that datasets are getting larger and more complex every day, while at the same time, users still want interactive ways to explore them. Instead of looking at the values, it is better to visualize the data. In that sense, methods like t-SNE and UMAP have been widely used, but they also show their limits when scaling to very large volumetric datasets. HSNE fits into a newer direction where, instead of trying to embed everything at once, the data is organized into multiple levels of abstraction. This idea of looking at the data at different scales is something that also appears in other areas of computer science, especially in large-scale data systems and machine learning, where full global computation is often too expensive.

This also introduces some challenges. Even though HSNE improves structure and reduces clutter, the embedding still does not correspond directly to meaningful physical quantities. Users still have to rely on patterns like clusters and distances, which can feel intuitive visually but are still the result of a complex transformation. In practice, this makes the interpretation of results somewhat subjective. Two users might not necessarily define TFs in the same way, even if they start from the same data, simply because the embedding does not provide a single correct way of interpreting it. When two different users interpret the embedding and visualize the dataset differently, it is possible for them to end up with two different conclusions. In the medical domain, this can lead to problems where users can mistake cancer cells for healthy cells.

Also an important consideration is that many users may not be experienced with either the visualization system or DR techniques. As a result, users may incorrectly assume that points that are relatively close to each other in a DR-based embedding always represent similar structures, or that points that are far apart are necessarily unrelated. However, as shown in Section 5.2, structures that are similar in the high-dimensional space can still become separated into different clusters in the embedding. This introduces a potential risk in real-world applications, where incorrect interpretation of the embedding may lead to misleading conclusions. Therefore, users should be properly trained to understand the limitations and ambiguities of DR-based visualizations.

Furthermore, by using HSNE instead of t-SNE, our visualization technique can be applied with lower computational and memory requirements. This makes the approach more accessible on a wider range of devices and enables the visualization of larger datasets without requiring high-end hardware. As a result, the method can be more easily used by people working with large volumetric datasets, such as medical students, geoscientists, materials scientists, or researchers working with scientific simulations and medical imaging data.

Another thing that became clear during this work is that a lot of the system depends on approximations, especially in the rendering pipeline. Using methods like HNSW and landmark-based mappings makes everything interactive and feasible at this scale, but it also means that the visualization is never an exact representation of the underlying high-dimensional space. This is not necessarily a problem, since approximation is a standard part of modern computer science, especially in large-scale systems, but it does mean that results should be interpreted carefully. In particular, when these techniques are used

in real-world analysis tasks, it is important to be aware that performance optimizations can influence what patterns are visible.

Overall, this project shows both the strengths and limitations of embedding-based visualization. On one hand, hierarchical DR like HSNE makes it possible to work with datasets that would otherwise be completely unmanageable in an interactive setting. On the other hand, it also makes the visualization more abstract, which shifts the burden of interpretation to the user. In that sense, this work reflects a larger trend in computer science: we are increasingly able to compute and visualize very complex data, but understanding that data still depends heavily on how these computational choices shape what we see.

References

[1] Dina Mohammed Sherif El-Torky et al. “3D Visualization of Brain Tumors Using MR Images: A Survey”. In: *Current medical imaging reviews* 15.4 (2019), pp. 353–361. DOI: 10.2174/1573405614666180111142055.

[2] Nawsher Khan et al. “The 51 V’s Of Big Data: Survey, Technologies, Characteristics, Opportunities, Issues and Challenges”. In: *Proceedings of the International Conference on Omni-Layer Intelligent Systems*. COINS ’19. Crete, Greece: Association for Computing Machinery, 2019, pp. 19–24. ISBN: 9781450366403. DOI: 10.1145/3312614.3312623.

[3] Iqbal H. Sarker. “Data Science and Analytics: An Overview from Data-Driven Smart Computing, Decision-Making and Applications Perspective”. In: *SN computer science* 2.5 (2021), p. 377. DOI: 10.1007/s42979-021-00765-8.

[4] Ravi Snellenberg. “Neighborhood-Preserving Dimensionality Reduction for Multivariate Volume Rendering”. MA thesis. Delft University of Technology, 2025.

[5] Laurens van der Maaten and Geoffrey Hinton. “Visualizing Data using t-SNE”. In: *Journal of Machine Learning Research* 9.86 (2008), pp. 2579–2605. URL: http://jmlr.org/papers/v9/vandermaaten08a.html.

[6] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: 1802.03426 [stat.ML]. URL: https://arxiv.org/abs/1802.03426.

[7] Nicola Pezzotti et al. “Hierarchical Stochastic Neighbor Embedding”. In: *Computer Graphics Forum* (2016). DOI: 10.1111/cgf.12878.

[8] Christof Rezk Salama. “Visual Parameters and Transfer Functions”. In: *Trends in Interactive Visualization: State-of-the-Art Survey*. Ed. by Robert Liere, Tony Adriaansen, and Elena Zudilova-Seinstra. London: Springer London, 2009, pp. 99–116. ISBN: 978-1-84800-269-2. DOI: 10.1007/978-1-84800-269-2_5.

[9] Andrzej Maćkiewicz and Waldemar Ratajczak. “Principal components analysis (PCA)”. In: *Computers Geosciences* 19.3 (1993), pp. 303–342. ISSN: 0098-3004. DOI: https://doi.org/10.1016/0098-3004(93)90090-R.

[10] Vincent Van Unen et al. “Visual analysis of mass cytometry data by hierarchical stochastic neighbour embedding reveals rare cell types”. In: *Nature communications* 8.1 (2017), p. 1740. DOI: 10.1038/s41467-017-01689-9.

[11] Marc Levoy. “Display of surfaces from volume data”. In: *IEEE Computer Graphics and Applications* 8.3 (1988), pp. 29–37. DOI: 10.1109/38.511.

[12] Nelson Max. “Optical models for direct volume rendering”. In: *IEEE Transactions on Visualization and Computer Graphics* 1.2 (1995), pp. 99–108. DOI: 10.1109/2945.468400.

[13] Mathias Schott et al. “A Directional Occlusion Shading Model for Interactive Direct Volume Rendering”. In: *Computer Graphics Forum* 28.3 (2009), pp. 855–862. DOI: 10.1111/j.1467-8659.2009.01464.x.

[14] Klaus Engel et al. *Real-Time Volume Graphics (1st ed.)* A K Peters/CRC Press, 2006. DOI: 10.1201/b10629.

[15] Markus Hadwiger et al. “Advanced illumination techniques for GPU volume raycasting”. In: *ACM SIGGRAPH ASIA 2008 Courses*. SIGGRAPH Asia ’08. Singapore: Association for Computing Machinery, 2008. ISBN: 9781450379243. DOI: 10.1145/1508044.1508045.

[16] Patric Ljung et al. “State of the Art in Transfer Functions for Direct Volume Rendering”. In: *Comput. Graph. Forum* 35.3 (June 2016), pp. 669–691. ISSN: 0167-7055. URL: https://doi.org/10.1111/cgf.12934.

[17] Xujia Qin et al. “Research and Implement of Multi-Dimensional Transfer Function Based on Boundaries”. In: *2009 2nd International Conference on Biomedical Engineering and Informatics*. 2009, pp. 1–4. DOI: 10.1109/BMEI.2009.5305709.

[18] Gordon Kindlmann et al. “Curvature-based transfer functions for direct volume rendering: methods and applications”. In: *IEEE Visualization, 2003. VIS 2003.* 2003, pp. 513–520. DOI: 10.1109/VISUAL.2003.1250414.

[19] Joe Kniss et al. “Interactive texture-based volume rendering for large data sets”. In: *IEEE Computer Graphics and Applications* 21.4 (2001), pp. 52–61. DOI: 10.1109/38.933524.

[20] Jesus J. Caban and Penny Rheingans. “Texture-based transfer functions for direct volume rendering”. In: *IEEE transactions on visualization and computer graphics* 14.6 (2008), pp. 1364–1371. DOI: 10.1109/TVCG.2008.169.

[21] Sangmin Park and Chandrajit Bajaj. “Feature Selection of 3D Volume Data through Multi-Dimensional Transfer Functions”. In: *Pattern recognition letters* 28.3 (2007), pp. 367–374. DOI: 10.1016/j.patrec.2006.04.008.

[22] Bo Pan et al. “Differentiable Design Galleries: A Differentiable Approach to Explore the Design Space of Transfer Functions”. In: *IEEE Transactions on Visualization and Computer Graphics* 30.1 (2024), pp. 1369–1379. DOI: 10.1109/TVCG.2023.3327371.

[23] Yiyao Wang et al. *IntuiTF: MLLM-Guided Transfer Function Optimization for Direct Volume Rendering*. 2025. arXiv: 2506.18407 [cs.GR]. URL: https://arxiv.org/abs/2506.18407.

[24] Chunxiao Xu et al. “Efficient Multi-Material Volume Rendering for Realistic Visualization with Complex Transfer Functions”. In: *Journal of Imaging* 11.6 (2025). ISSN: 2313-433X. DOI: 10.3390/jimaging11060193.

[25] Ayush Kumar et al. “RadVolViz: An Information Display-Inspired Transfer Function Editor for Multivariate Volume Visualization”. In: *IEEE Transactions on Visualization and Computer Graphics* 30.8 (2024), pp. 4464–4479. DOI: 10.1109/TVCG.2023.3263856.

[26] Jonathon Shlens. *A Tutorial on Principal Component Analysis*. 2014. arXiv: 1404.1100 [cs.LG]. URL: https://arxiv.org/abs/1404.1100.

[27] Yuan-chin Ivan Chang. *A Survey: Potential Dimensionality Reduction Methods*. 2025. arXiv: 2502.11036 [stat.OT]. URL: https://arxiv.org/abs/2502.11036.

[28] Alexander Broersen and Robert van Liere. “Transfer Functions for Imaging Spectroscopy Data using Principal Component Analysis.” In: Jan. 2005, pp. 117–123. DOI: 10.2312/VisSym/EuroVis05/117-123.

[29] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. “A global geometric framework for nonlinear dimensionality reduction”. In: *Science (New York, N.Y.)* 290.5500 (2000), pp. 2319–2323. DOI: 10.1126/science.290.5500.2319.

[30] Sam T. Roweis and Lawrence K. Saul. “Nonlinear Dimensionality Reduction by Locally Linear Embedding”. In: *Science* 290.5500 (2000), pp. 2323–2326. DOI: 10.1126/science.290.5500.2323.

[31] Han Suk Kim et al. “Dimensionality Reduction on Multi-Dimensional Transfer Functions for Multi-Channel Volume Data Sets”. In: *Information visualization* 9.3 (2010), pp. 167–180. DOI: 10.1057/ivs.2010.6.

[32] Sangbong Yoo, Seokyeon Kim, and Yun Jang. “Two-Level Transfer Functions Using t-SNE for Data Segmentation in Direct Volume Rendering”. In: *IEEE Transactions on Visualization and Computer Graphics* 31.9 (2025), pp. 5948–5961. DOI: 10.1109/TVCG.2024.3484471.

[33] Walter Serna-Serna, Andrés Marino Álvarez-Meza, and Álvaro Orozco-Gutiérrez. “Fast Semi-Supervised t-SNE for Transfer Function Enhancement in Direct Volume Rendering-Based Medical Image Visualization”. In: *Mathematics* 12.12 (2024). ISSN: 2227-7390. DOI: 10.3390/math12121885.

[34] Laurens van der Maaten. “Accelerating t-SNE using Tree-Based Algorithms”. In: *Journal of Machine Learning Research* 15.93 (2014), pp. 3221–3245. URL: http://jmlr.org/papers/v15/vandermaaten14a.html.

[35] Laurens van der Maaten. “Learning a Parametric Embedding by Preserving Local Structure”. In: *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*. Ed. by David van Dyk and Max Welling. Vol. 5. Proceedings of Machine Learning Research. Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA: PMLR, 16–18 Apr 2009, pp. 384–391. URL: https://proceedings.mlr.press/v5/maaten09a.html.

[36] Markus Hadwiger et al. “Real-Time Ray-Casting and Advanced Shading of Discrete Isosurfaces”. In: *Computer Graphics Forum* 24.3 (2005), pp. 303–312. DOI: https://doi.org/10.1111/j.1467-8659.2005.00855.x.

[37] Eric LaMar, Bernd Hamann, and Kenneth I. Joy. “Multiresolution techniques for interactive texture-based volume visualization”. In: *Proceedings Visualization ’99 (Cat. No.99CB37067)*. 1999, pp. 355–543. DOI: 10.1109/VISUAL.1999.809908.

[38] Alexandru C. Telea. *Data visualization: principles and practice*. cRC Press, 2014. URL: https://webspace.science.uu.nl/~telea001/uploads/PAPERS/VISBOOK14/visbook.pdf.

[39] George W. Furnas. “Generalized fisheye views”. In: *Acm Sigchi Bulletin* 17.4 (1986), pp. 16–23. DOI: 10.1145/22339.22342.

[40] Niklas Elmqvist et al. “Fluid interaction for information visualization”. In: *Information Visualization* 10.4 (2011), pp. 327–340. DOI: 10.1177/1473871611413180.

[41] Petr Šereda, Anna Vilanova, and Frans A. Gerritsen. “Automating Transfer Function Design for Volume Rendering Using Hierarchical Clustering of Material Boundaries”. In: *EUROVIS - Eurographics /IEEE VGTC Symposium on Visualization*. Ed. by Beatriz Sousa Santos, Thomas Ertl, and Ken Joy. The Eurographics Association, 2006. ISBN: 3-905673-31-2. DOI: 10.2312/VisSym/EuroVis06/243-250.

[42] Clarence Yapp et al. “Highly Multiplexed 3D Profiling of Cell States and Immune Niches in Human Tumours”. In: *bioRxiv* (2025). DOI: 10.1101/2023.11.10.566670.

[43] Thomas Hollt et al. “CyteGuide: Visual Guidance for Hierarchical Single-Cell Analysis”. English. In: *IEEE Transactions on Visualization and Computer Graphics* 24.1 (2018), pp. 739–748. ISSN: 1077-2626. DOI: 10.1109/TVCG.2017.2744318.

[44] Jiarui Ding, Anne Condon, and Sohrab P Shah. “Interpretable dimensionality reduction of single cell transcriptome data with deep generative models”. In: *Nature communications* 9.1 (2018), p. 2002.