QVis: **Q**uery-based **Vis**ual Analysis of Multiscale Patterns in Spatiotemporal Ensembles

Ruben Bauer, Quynh Quang Ngo, Guido Reina, Steffen Frey, Michael Sedlmair

***Abstract*—Understanding how dynamic patterns vary across large spatiotemporal ensembles is essential in many scientific domains. In fluid dynamics, for instance, researchers analyze how splash patterns in droplet impact experiments change with physical parameters such as fluid type or impact velocity. These experiments produce large volumes of data where patterns differ in size, shape, and duration, making manual analysis tedious and error-prone. Recently, interactive visualization approaches have been developed to assist analysis using learned similarity models for pattern-based querying. However, they assume fixed-size inputs and only support single-pattern queries, thus limiting their effectiveness for multiscale, multi-pattern analysis and exploration of ensembles. In this paper, we present a visual analysis approach for the interactive exploration of spatiotemporal ensembles through multiscale pattern querying. Our approach extends an existing similarity model to support variable-sized patterns, allowing users to define queries by selecting examples directly on visualized data. Coordinated views enable interactive querying, comparison, and analysis of pattern occurrences and relate pattern occurrences to ensemble parameters. A guidance mechanism supports the user in finding underexplored regions. We demonstrate the utility of our approach on synthetic and real-world datasets. Domain expert feedback confirms that the approach is intuitive, easy to use, and effective for revealing parameter-pattern relationships.**

***Index Terms*—Machine learning, Feature extraction or construction, tracking, clustering**

I. Introduction

In droplet impact experiments, researchers analyze how splash patterns vary with parameters such as fluid type or impact velocity [1]. These experiments are typically recorded with high-speed cameras and produce a large spatiotemporal ensemble of videos. In these videos, different patterns may occur that differ in size, shape, and duration. Understanding how impact parameters relate to these multiscale patterns is relevant for applications such as inkjet [2] or 3D printing [3]. Yet manually reviewing large video ensembles to identify such relationships is tedious, time-consuming, and prone to oversight. This challenge reflects a broader need in physics and related domains, where researchers systematically vary input parameters in simulations or experiments to study dynamical phenomena. Visualization tools play a crucial role in helping domain experts explore the data generated by these studies, detect patterns, and build intuition about underlying physical models.

State-of-the-art visual analysis approaches for large spatiotemporal ensembles increasingly rely on machine learning models that learn similarity metrics to support pattern-based comparison [4], [5]. One notable example is S4 [6], which learns to compare spatiotemporal data based on their dynamic behavior. S4 enables query-driven analysis by allowing users to specify example patterns and retrieve similar instances across an ensemble. However, S4 supports only fixed-size inputs, making it difficult to compare patterns that vary in duration or scale, such as small versus large splash formations that occur under different impact velocities. While S4 and related approaches represent significant progress, they lack in interactive, multiscale, and multi-pattern analysis and comparative exploration of large spatiotemporal ensembles.

To address these limitations, we introduce **QVis**, a visual analysis approach for the interactive exploration of large spatiotemporal ensembles through flexible, multiscale pattern querying. At its core, QVis extends the S4 similarity model to support variable-sized inputs, enabling the comparison and retrieval of patterns that differ in size, shape, and duration. The visual interface of QVis consists of coordinated and linked views that support both pattern-based exploration and the analysis of ensemble members and their relationship to the parameter space. Users define multiscale pattern queries by directly selecting example regions in the visualized data of individual ensemble members. For each query, QVis retrieves similar patterns across the ensemble and visualizes when and where they occur. Timeline views enable side-by-side comparison of pattern occurrences across members, while automated grouping highlights members with similar occurring patterns. These groupings are linked to a parameter space view, supporting interactive analysis of parameter-pattern relationships. A guidance mechanism further supports users in discovering underexplored regions of the ensemble and refining/adding queries accordingly.

We evaluate QVis using both synthetic and real-world spatiotemporal ensembles. First, we compare the extended similarity metric against the fixed-size baseline and demonstrate with synthetic data that our extended model significantly improves retrieval of multiscale patterns. Second, we apply QVis to a droplet impact ensemble, demonstrating how users can query for multiple splash patterns, analyze and compare their temporal occurrences across the ensemble, and relate the patterns to the corresponding droplet impact experiment

Corresponding author: Ruben Bauer

Ruben Bauer, Quynh Quang Ngo, Guido Reina, and Michael Sedlmair are affiliated with VISUS at the University of Stuttgart, Germany. (emails: {ruben.bauer|quynh.ngo|guido.reina|michael.sedlmair}@visus.uni-stuttgart.de)

Steffen Frey is with the University of Groningen, The Netherlands. (email: s.d.frey@rug.nl)

parameters. We validate our findings through two expert interviews, which confirm that QVis efficiently reproduces insights previously obtained through hundreds of hours in manual analysis. Finally, we demonstrate the general applicability of our approach using a first-person vision ensemble, where multi-pattern querying reveals movement dynamics across visitors.

In conclusion, our main contributions are:

- QVis, a visual analysis approach for exploring large spatiotemporal ensembles through flexible multiscale pattern querying using a learned multiscale similarity metric.
- A prototype implementation of QVis that is evaluated with expert feedback confirming its usability and effectiveness for non-visualization experts.
- Further evaluation of our metric and visual approach on synthetic and real-world datasets.

II. Related Work

We review prior work on machine learning-assisted visual analytics for spatiotemporal data and learning-based approaches to pattern similarity, with a focus on enabling multiscale querying.

A. ML-Assisted Visual Analytics for Spatiotemporal Data

In recent years, machine learning (ML) models have been increasingly used in visual analytics (VA) systems to assist the analysis of large and complex datasets, such as spatiotemporal ensembles [7]–[9]. Many of these systems interactively incorporate domain knowledge through active learning [10], [11] or other forms of user input [12]. Other systems support example-based querying, allowing users to define patterns to search and analyze from visualized data [6], [13]–[15]. Among these, S4 by Tkachev et al. [6] is most closely related to our work. S4 enables the user to specify example patterns in spatiotemporal data and retrieve similar instances across an ensemble using a learned similarity metric. However, its rudimentary UI supports only single queries, and S4 itself is restricted to fixed-size patterns, which limits users in the analysis and comparison of patterns that vary in size, shape, or duration. Our work builds on the core ideas of S4 by building a novel visual analysis approach for exploring and analyzing patterns in spatiotemporal ensembles using multiple variable-sized queries.

B. Learning-based Spatiotemporal Similarity

A central requirement for querying and comparing patterns in spatiotemporal data is a suitable similarity metric. To address this need, various unsupervised machine learning approaches have been proposed to learn feature representations of spatiotemporal data, since labeled data is rare in scientific datasets [4], [5]. Among this line of work, several utilize Siamese networks [16], [17], while others use Auto-Encoders [18]–[21]. S4, proposed by Tkachev et al. [6], is also relevant in this regard. It learns a similarity metric via a self-supervised Siamese network trained on fixed-size spatiotemporal patches. These patches represent rectangular subsets of the data, like a bounding box drawn onto an image frame of a video, but which may extend over multiple time steps and contain patterns such as splashes in droplet impact experiments. The model assumes that similar patterns occur in close spatial and temporal neighborhoods and uses this assumption to train without labeled data. Once trained, the model encodes patches into latent space embeddings. There, two embeddings of similar patterns should have a small L1-distance, and a large distance otherwise. The resulting similarity metric, that is, the L1 distance applied to the patch embeddings, enables querying an ensemble by comparing user-provided example patches to randomly sampled patches in the ensemble. We adopt S4's querying mechanism and its similarity metric to implement our visual approach. However, the S4 model is constrained to fixed-size inputs and cannot be used on other input sizes without modifying the architecture and retraining the model. To overcome these limitations, we extend the S4 model with a spatiotemporal pooling layer and adjust the training approach for variable-sized input patches.

Our extension of the S4 model builds upon advances in computer vision, where various approaches have been proposed to extract features from inputs of varying sizes [22]–[25]. One notable example is Spatial Pyramid Pooling (SPP) [24], which uses multiple adaptive pooling operations to generate a pyramid of feature maps. While originally developed to overcome the fixed-size constraint of Convolutional Neural Networks for image processing, SPP has also found applications for spatiotemporal data. For instance, Yang and Yuan [26] proposed a multiscale framework for storm nowcasting by leveraging SPP layers in their model. Inspired by these ideas, we implement a spatiotemporal pooling layer using adaptive pooling, allowing the extended model to generate fixed-size latent space embeddings from variable-sized inputs. This adjustment removes the fixed-size constraint of the original S4 model and enables robust multiscale pattern querying in our visual analysis approach.

III. Learning a Self-Supervised Similarity Metric for Multiscale Pattern Querying

Building on the original S4 model, we train a self-supervised network that encodes variable-size spatiotemporal patches into latent space embeddings, where the L1-distance between embeddings reflects pattern similarity. In this section, we describe how we extend and train the S4 model to enable multiscale pattern querying in our visual analysis approach.

A. Spatiotemporal Pooling Layer

To enable multiscale querying, we extend the S4 architecture with a **spatiotemporal pooling layer**. While Convolutional Neural Networks naturally process inputs of different sizes, their resulting output sizes vary depending on the sizes of the input, making it challenging to compute distances between them [27]. To address this issue, we adopt adaptive pooling, which has been widely used in computer vision to generate fixed-size outputs from variable-size inputs [24], [28], [29].

Our spatiotemporal pooling layer operates separately along the spatial and temporal dimensions of the input. For each

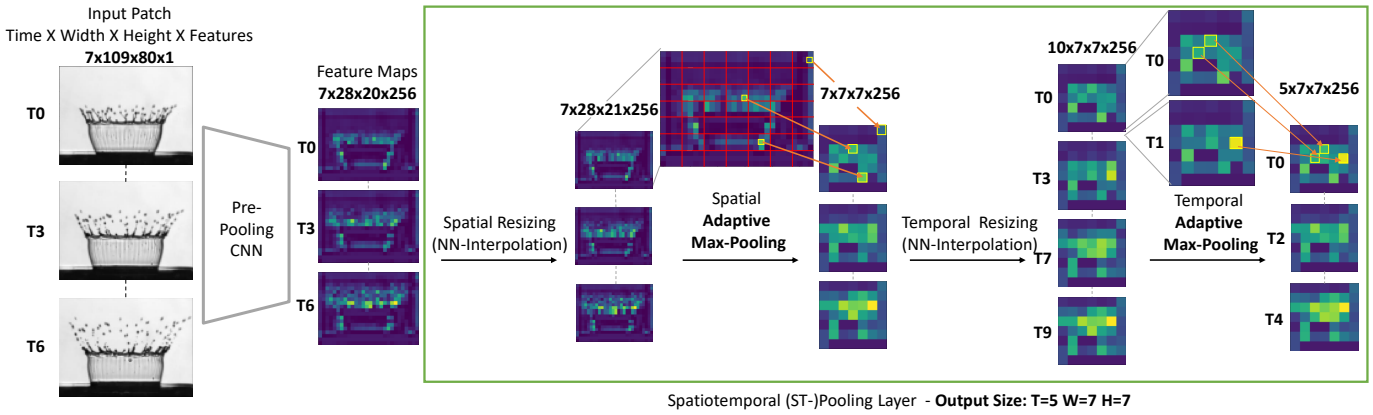

Fig. 1: Illustration of the data flow in our spatiotemporal pooling layer for a patch capturing a crown-splash pattern after a droplet impact. The Pre-Pooling CNN generates feature maps from the input patch, which are then passed to the spatiotemporal pooling layer. The potentially arbitrarily sized input feature maps are then resized to their next multiple of the predefined output size using nearest neighbor (NN) interpolation, and then adaptively pooled in first spatial and then temporal dimensions.

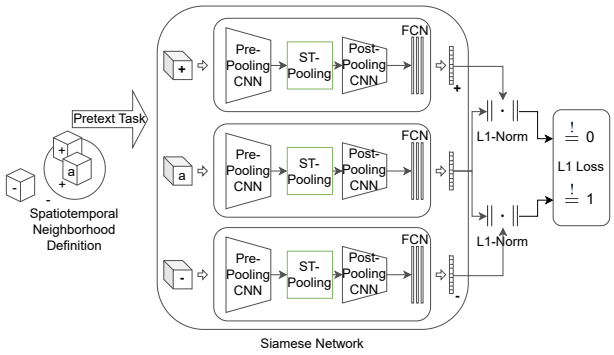

Fig. 2: The architecture of our model. During training, triplets of anchor (a), positive (+) and negative (-) patches are automatically sampled from the ensemble data and passed to the Siamese network. The model is trained such that the L1 distance between the latent space embeddings of (a) and (+) is 0, and (a) and (-) is 1, using the L1 Loss.

dimension, we interpolate the input to a multiple of the predefined output size. This interpolation step ensures that the input is divided into equally sized bins during the adaptive pooling operation, where the number of bins is equal to the predefined output size. We then apply adaptive pooling, where the maximum input value of each bin is taken to form the output, resulting in a fixed-size spatiotemporal output regardless of the input size. We first apply spatial and then temporal interpolation and adaptive pooling.

Figure 1 illustrates the data flow in our spatiotemporal pooling layer applied to the spatiotemporal output (feature maps) of the preceding convolution layers. The resulting fixed-size feature maps can then be passed to downstream layers and embedded into a shared latent space for similarity comparison.

B. Model Architecture and Training

Our complete model architecture is shown in Fig. 2. A single input patch is first fed to the *Pre-Pooling CNN* layer to generate feature maps with reduced spatial extent. We then apply the *Spatiotemporal Pooling Layer* to pool the feature maps to a predefined output size. We then apply additional convolutions in the *Post-Pooling CNN* layer to further reduce the total amount of features. Finally, we flatten the resulting feature maps to a single feature vector and use it as input for the linear layers. The output of the linear layers is the latent space embedding of the input patch. The latent space embeddings of patches serve as input for the L1-distance metric as part of the similarity computation during evaluation and to compute the loss during training.

We train our model based on the same assumption as the S4, but extend it to also include patches of different sizes. Each training sample consists of a triplet of patches: an anchor patch, a positive patch, and a negative patch. The positive patch is from the same spatiotemporal neighborhood as the anchor patch, while the negative patch is from outside this spatiotemporal neighborhood.

We use the L1 loss to train the model such that the L1 distance of the positive pair (anchor patch and positive example) should be zero, and one for the negative pair (anchor patch and negative example). Once trained, our similarity metric for two spatiotemporal patches is equal to the L1-distance between their latent space embeddings generated from the model.

In our implementation, the *Pre-Pooling* and *Post-Pooling* CNNs use 3D convolutions and standard max-pooling operations. The post-pooling layer additionally uses 1D convolutions on the temporal dimension only. We use GELU [30] activations after each convolutional layer throughout the network. We implemented our model using PyTorch [31]. Further details on architecture and training are available in the supplemental material to this paper.

C. Multiscale Querying and Matching

We adopt the same concept as the S4 to enable querying by example using our learned similarity metric. Each query q can consist of multiple positive ($p_{q_{pos}}$) and negative ($p_{q_{neg}}$)

example patches that represent the desired and undesired patterns respectively. Given a query q, the goal is to retrieve patches from the ensemble that are similar to the positive examples and dissimilar to the negative examples, based on our learned similarity metric. In contrast to S4, our model supports variable-size input patches, enabling multiscale pattern querying.

To evaluate how well a test patch p from the ensemble matches a given query q, we use the following scoring function:

$$score(p,q) = \sum_{p_{q_{pos}}} d(p, p_{q_{pos_i}}) - \sum_{p_{q_{neg}}} d(p, p_{q_{neg_j}})$$

Here, $d(p_a, p_b)$ is the L1-distance between the latent space embeddings of patches p_a and p_b, computed using our trained model. Thus, d represents our similarity metric.

During analysis, the score function is applied to each sample in a precomputed search space of patches in the ensemble (see Section IV-A). The user can then specify a threshold θ_q to control which of the sampled patches are considered matches for the query: all samples p_s with a score $score(p_s, q) <= \theta_q$ are considered matches for the query.

This querying and matching mechanism forms the computational foundation of our interactive visual analysis approach, which we describe in the following section.

IV. QVis: Query-based Visual Ensemble Analysis

Our overarching goal is to support real-world analysis workflows for spatiotemporal ensemble data. Initially we derive our core requirements from well-established ensemble analysis tasks [9], including *feature*, *overview*, *parameter*, *compare*, *cluster*, and *trend*. We further refined these requirements through informal but targeted discussions with domain experts over the course of several meetings. These experts have many years of experience in their respective fields, including porous media research, aerospace engineering, and experimental fluid dynamics. In these discussions, we identified a clear need for automated approaches to help accelerate time-costly analysis steps such as identifying complex patterns in large spatiotemporal ensembles. We therefore define the following requirements for our visual analysis approach:

($\mathbf{R_1}$) **Visual Multiscale Querying:** Visual querying has been identified as a fundamental exploratory technique for spatiotemporal data [32]. It provides an intuitive approach for a user to select interesting example patterns and then analyze when and where similar patterns occur. To support a flexible and intuitive pattern selection and to allow the retrieval of similar patterns at different scales, we require that our approach should support a visual querying mechanism for multiscale patterns.

($\mathbf{R_2}$) **Efficient Ensemble Exploration:** Capturing all the essential patterns is imperative to fully reflect the ensemble member's dynamics and draw conclusions about member-pattern relationships. Due to the potentially very large size of spatiotemporal datasets, we require an efficient exploration mechanism that guides the user during the analysis to identify new patterns in the ensemble.

($\mathbf{R_3}$) **Capability in Relating Input Parameters to Pattern Occurrences:** Relating input parameters to occurring patterns in the ensemble helps to understand the underlying models and physics of corresponding simulations and experiments. This has been widely recognized as a key task in ensemble analysis [9], [33]–[35]. To guide the user to new insights and hypotheses about the parameter-pattern relationships during the interactive analysis, we require the capability of relating input parameters to pattern occurrences in the ensemble.

($\mathbf{R_4}$) **High Responsiveness for Interactive Analysis:** To support an interactive analysis, the visual analysis approach must return results quickly despite the large search space.

A. QVis's Core Functionalities

We propose to fulfill the requirements ($\mathbf{R_1}$) - ($\mathbf{R_4}$) by supporting four core functionalities with our approach:

($\mathbf{F_1}$): We fulfill ($\mathbf{R_1}$) by utilizing our proposed similarity metric and query scoring function to perform *multiscale pattern querying* (see Section III-C). It means that users can select positive and negative example patches of patterns in the ensemble, and our approach returns patches of varying sizes that contain similar patterns to the positive examples and dissimilar ones to the negative examples. To support flexible and intuitive selection of example patches, the data is visualized and users are able to select patterns by simply drawing bounding boxes around them in the visualization.

($\mathbf{F_2}$): To address ($\mathbf{R_2}$), we support the use of *multiple simultaneous queries* to capture diverse patterns across the ensemble.

($\mathbf{F_3}$): Building on ($\mathbf{F_2}$), we introduce a *guided exploration* mechanism. This mechanism is based on existing queries, whose matching patches implicitly label parts of the ensemble data. By providing an overview that automatically reveals the data that is not labeled yet, the user can be guided to potentially new patterns. Iteratively creating new queries for patterns in the unlabeled data will facilitate discovering all the essential patterns. Functionalities ($\mathbf{F_2}$) and ($\mathbf{F_3}$) in combination, therefore, fulfill ($\mathbf{R_2}$).

($\mathbf{F_4}$): To support ($\mathbf{R_3}$), relating input parameter configurations to the member's dynamics, we propose an interactive *pattern map* of the ensemble, which maps the ensemble's dynamics to the respective members and parameters. We characterize the member's dynamics by the temporal order of detected pattern occurrences. This characterization allows us to compactly describe each member's temporal trend as a pattern sequence and group members with similar trends together.

Finally, to fulfill ($\mathbf{R_4}$) and enable a fast and interactive analysis, we propose to sample the ensemble and compute the sample's latent space embeddings in a pre-processing step. For the sake of simplicity, we choose uniform random sampling to sample the search space while limiting the total number of samples. For each sample, we randomly choose a member from the ensemble and then sample a patch from that member. The pre-processing step yields a set of samples, each consisting of a patch description (location and size of the patch) and the patch's latent-space embedding representation. This set of samples is the search space during the analysis.

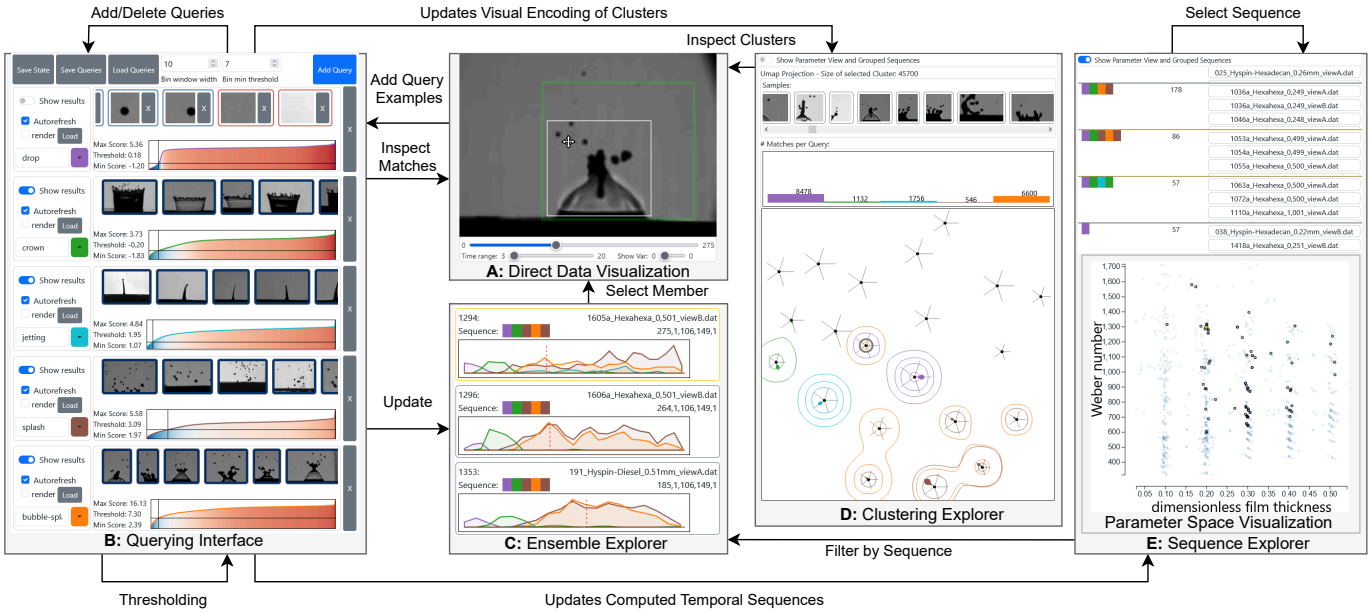

Fig. 3: Our visual analysis approach consists of five linked views: (A) direct data visualization, (B) querying interface, (C) ensemble explorer, (D) clustering explorer, and (E) sequence explorer. Here, we used (A) for *multiscale querying* and selected droplet impact splash patterns from the data to create the *multiple queries* shown in (B): drop, crown, jetting, splash, and bubble-splash. In (B), we specified thresholds to define which samples in a precomputed search space of the ensemble can be considered a match for each query. The temporal distributions of these matches per ensemble member are visualized in the timeline views of (C), allowing to compare the members based on their pattern occurrences over time. Our approach then computes a short pattern sequence for each member, describing the temporal order in which the found patterns occur. The color-coded overview in (D) enables *guided pattern exploration*, by allowing us to search new patterns in the clustered ensemble's search space for which no matching queries exist yet (uncolored glyphs). Ensemble member with similar pattern sequence are grouped together and visualized in (E). The accompanied *pattern map* in (E) then allows to explore the associated parameters for each identified pattern sequence. Here, we see that splash and bubble-splash patterns (dark circles) can occur for mid to high values of Weber number and dimensionless film thickness.

B. Visual Components

To support the core functionalities $(\mathbf{F_1})$ - $(\mathbf{F_4})$, we designed a coordinated set of five visual components (views A-E, see Fig. 3). The views and interaction methods were designed to be intuitive and easy-to-use for non-visualization experts, while considering our identified requirements and visual scalability for large spatiotemporal ensembles. Each view contributes to a specific part of the analysis workflow and aligns with one or more of the core functionalities. The views are tightly linked, allowing users to iteratively select patterns, issue and modify queries, explore results, and gain insight into parameter-pattern relationships across the ensemble.

We will now describe the individual views in detail:

View A: Direct Data Visualization provides a direct visualization of the data in a currently selected ensemble member and time step. A slider allows the user to control the visualized time step and observe how the spatial data changes over time. The user can draw a bounding box on the visualized data to select a patch from the ensemble member (as shown in Fig. 3 (A) with the gray rectangle and arrow cursor to move it around) and add it as either positive or negative example in the currently selected query by pressing the plus or minus keys respectively. This patch contains the visualized data inside the drawn bounding box from the current time step plus up to several subsequent time steps, allowing to capture the pattern's dynamics over time. The user can specify this number of subsequent time steps with a second slider below. With view A, we partially support $(\mathbf{F_1})$ regarding intuitive pattern selection for multiscale querying.

View B: Querying Interface visualizes all current queries. It supports $(\mathbf{F_2})$ by allowing the user to create additional queries using the "Add Query" button, modify a query by adding and removing examples (using the "X" button next to each example patch), and to remove a query (by using the large "X" button on the right side of the query). A switch button allows toggling between showing the query's example patches as shown in the first drop query in Fig. 3 (B), or to show the query's result as shown in the other queries below. The example patches are color-coded to have a blue border for positive examples, and a red border for negative examples. The query results are the sorted samples of the ensemble's search space scored by our scoring function from Section III-C. The borders of the visualized sorted samples are linearly color-coded according to their scores: blue for low scores (similar patches) and red for high scores (dissimilar patches). The user can use scroll bars to navigate query example patches, corresponding results, and other queries in the querying interface. Each query is accompanied by a graph that shows the scores of the sorted

samples using the same color scheme (blue for low, red for high scores). The user can use this graph to interactively specify the threshold θ_q, which determines the samples that are considered matches for the query. Only samples with a score $\leq \theta_q$ are considered matches. A checkbox can then be clicked to enable or disable the rendering of each query's matches in view A. To distinguish multiple queries from each other when linking them with other views, the user can assign name and color to each query q.

View C: Ensemble Explorer allows the user to quickly navigate the ensemble. It visualizes the temporal distribution of each query's matches in one timeline view per member. Each timeline view contains superimposed line-charts that visualize the member's temporal distributions of query matches over their time-axis. The user can analyze the timeline views to identify when which patterns occur in the member, and how many corresponding query matches have been found. Many matches suggest strong, and few matches suggest weak presence of the queried pattern at the corresponding time steps. Time intervals with few or no matches suggest unidentified patterns that the user can explore by navigating to them (supporting functionality ($\mathbf{F_3}$)). This navigation is achieved by clicking on the respective time step in the timeline view, which will select and visualize it in view A. The timeline views include a vertical dashed line that shows the current time step in all members simultaneously, allowing to compare the dynamics at the same time step across members that vary in length.

Inspired by Luboschik et al. [36] towards creating a compact representation relating parameters to dynamics and support functionality ($\mathbf{F_4}$), we propose to condense the information in the timeline views by describing each as a short pattern sequence. The pattern sequence describes the temporal order of the queries with the most matches per ensemble member. To compute this sequence, we divide the timeline into fixed time intervals and count how many matches each query has in each interval. For every interval, we assign the query with the highest number of matches as the dominant one. Intervals with too few matches can be ignored, and consecutive intervals with the same dominant query are merged to keep the sequence short and readable. We visualize the pattern sequence per member as a row of colored rectangles placed above its timeline view, where each color represents a different query. For example, in Fig. 3 (C), all members from the droplet impact experiment share the same pattern sequence. In this case, the sequence shows that the droplet first falls, then creates a crown, followed by a splash, a bubble-splash, and again a splash pattern.

View D: Clustering Explorer allows the user to efficiently explore the different patterns in the ensemble, and therefore, supports functionality ($\mathbf{F_3}$) *guided pattern exploration*. It contains a projection view that shows a clustering of the ensemble's search space, though only visualizes the cluster centroids instead of individual samples to alleviate occlusion. The centroids are positioned corresponding to the 2D UMAP [37] projection of their centroids. We use the K-means clustering algorithm [38], [39] with K-means++ initialization [40] to compute the clustering. Both clustering and projection are based on our learned similarity metric from Section III.

We visualize the individual clusters in the projection as flower glyphs to show the number of the cluster's query matches compared to their total size. The queries represent the dimensions of the glyphs. Flower glyphs have been shown to outperform star glyphs for higher dimensions in outlier and sub-cluster detection [41], improving the visualization's scalability for multiple queries. The user can click on a centroid's flower glyph to view a scrollable list of the corresponding cluster's elements and analyze the number of matches per query in an accompanied bar chart, as shown in Fig. 3 (D).

We draw contours around the flower glyphs to further highlight where certain patterns appear in the projection and indicate to the user where there are still unidentified patterns. By indicating which clusters still contain unidentified patterns, the user can be guided to new patterns and iteratively query these until all essential patterns have been found. Each change to one of the queries also updates the clustering explorer.

View E: Sequence Explorer groups ensemble members by their pattern sequence as previously described in view C, which allows us to support functionality ($\mathbf{F_4}$) in relating the ensemble's dynamics to its members and associated parameters. Each group of members is visualized in a scrollable list sorted by group size. The user can select a group by clicking on the corresponding row in the sequence explorer, which also filters the members in the ensemble explorer, allowing the user to explore these members in detail (view C).

The sequence explorer includes a parameter space visualization, which visualizes the members' associated parameters, allowing the user to interactively explore pattern-member relationships. In the parameter space visualization in Fig. 3 (E), we visualize the 2D parameter space as scatterplot, where each member is represented as a blue and slightly transparent circle such that overlapping circles can still be identified. Selecting a group of members highlights them in the scatterplot by adding a black halo around their circles, improving the visual distinction between currently selected and non-selected members.

We note that ensembles with higher-dimensional parameter spaces would require other approaches to visualize their parameter space, such as a scatterplot matrix. Similar brushing and linking could then be employed to relate the ensemble's dynamics with its parameter space.

C. Analysis Workflow

We designed an analysis workflow to outline how the visual components can be used to make sense of an ensemble. We provide a flow diagram of this workflow in Fig. 4. Our analysis workflow begins by performing an *initial exploration* of the ensemble data. The projection of cluster centroids in the clustering explorer (view D) offers an initial overview of different spatiotemporal patterns. The direct data visualization (view A) and the ensemble explorer (view C) offer means to investigate individual clusters and ensemble members further. During the initial exploration, different spatiotemporal patterns are identified (*identificaiton of spatiotemporal pattern*). A user can then *query the identified patterns* by providing example

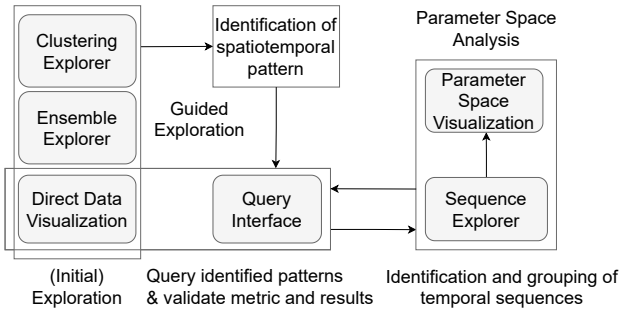

Fig. 4: The analysis workflow for our visual approach. We begin the analysis with an initial exploration to identify different occurring spatiotemporal patterns. Next, we make sense of the ensemble [42] (Fig. 4) by iteratively searching for and querying essential spatiotemporal patterns. The updated clustering and sequence explorer help providing overviews of the ensemble's dynamics and parameter-pattern relationships.

patches found during the exploration step. Based on the query results, a user can then *validate metric and results* and decide on the trustworthiness of the metric or whether the chosen query thresholds are appropriate. Given that the metric returns expected results, which are similar patterns to the provided examples, the user can continue the exploration or fine-tune the query by modifying query thresholds or examples.

Defining queries and corresponding thresholds updates the clustering explorer (view D) and the ensemble explorer (view C) by visualizing where query matches exist. This offers a *guided exploration* by searching for clusters or members that are not well described by the queries yet and contain almost few or no matches for any query. Further exploration and analysis of those clusters and query results then leads to identifying potentially new patterns. This loop of *identification of spatiotemporal pattern*, *querying for patterns & validation and analysis of results*, and *guided exploration* generates a variety of pattern sequences that compactly represent the trends of the ensemble members.

The temporal trends of the ensemble members can then be explored and analyzed with the interactive pattern map in the sequence explorer (view E). *Inspecting the sequences* supports the verification of expectations or the detection of anomalies. *Parameter space segmentations* may become visible and provide further insight for the overall ensemble analysis.

V. Evaluation

In this section, we first evaluate our query-based approach using a case study of a droplet impact experiment ensemble following our proposed workflow. We then conduct two expert interviews to validate our requirements and respective design decisions for our visual approach, and to evaluate its usefulness for the corresponding expert's domains. Finally, we compare our similarity metric to its fixed-size baseline (the S4) using a synthetic dataset, and demonstrate our approach's ability to also generalize to a real-world scenario of point-of-view camera recordings.

A. Case Study: Droplet Impact Experiments

Droplet impact experiments are conducted to investigate the splash dynamics of liquids, which is relevant for many industrial applications such as inkjet or 3D printing [2], [3]. To capture and analyze the dynamics during the many different experiments, high speed cameras are used, which quickly results in large spatiotemporal ensembles.

Droplet Impact Experiment Ensemble: The considered ensemble in this case study contains 1208 different droplet impact experiments, whereas each is captured by two cameras from slightly different perspectives, resulting in a total of 2416 members. Each member is a series of 2D grayscale images recorded with a high-speed camera. All experiments follow the same procedure: a droplet is dropped onto a small fluid film. The goal of the experiments were to investigate the different splash patterns that evolve for different types of droplet and film fluid, and build a characteristic map that puts splash patterns into relation to the experiment's parameters [1]. The two key parameters of interest are the *Weber number* and the *dimensionless film thickness*, both derived from measured properties including fluid type, impact velocity, droplet diameter, and film thickness.

Building a Characteristic Map: In an informal discussion with a domain expert for droplet impacts, we gained insights into their process for constructing a characteristic map:

The starting point for the derivation of a characteristic map for impact outcomes were the individual image sequences of the raw experimental data, each containing between 300 and 3000 images of raw experimental data. For each image sequence, the experimenter must determine the impact outcome (e.g., deposition, transition, or splashing) and annotate the presence, order, and timing of more detailed phenomena such as crown, jet or bubble formation. This information is then manually recorded in a spreadsheet. For an experienced experimenter, it takes between 5 to 10 minutes to analyze one image sequence, depending on the number of images and observed phenomena. Thus, analyzing an ensemble of experimental data with 1200 image sequences takes between 108 and 216 hours.

Based on this discussion, we identified a clear goal for our visual analysis approach to help with such analysis, particularly the time-consuming process of identifying and tracking of multiple impact phenomena (splash patterns).

Expected Analysis Output: Given the ensemble, our task is to identify and label essential multiscale patterns in the data. We then can classify the experiments with respect to the pattern sequences and analyze their relation to the experiment's parameters. The final result should be similar to the characteristic map that domain experts derived manually.

Procedure: We follow the workflow described in Section IV-C applied to the ensemble and provide evidence that our approach supports the requirements ($\mathbf{R_1}$-$\mathbf{R_4}$) from Section IV. For this case study, we first trained our model on the ensemble data, as described in Section III, and performed the sampling and pre-processing steps.

Result: We start with the initial exploration by using the direct data visualization (view A) for the first experiment in the

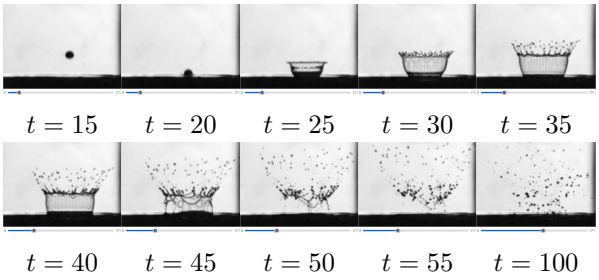

Fig. 5: A droplet impact experiment resulting in a series of different patterns: "drop" $t = 15, 20$, "crown" $t = 25, 30$, "crown-splash" $t = 35, 40$, and "splash" $t >= 45$.

Fig. 6: The patches of the drop query consisting of positive (blue) and negative (red) examples.

Fig. 7: The best retrieved sample patches for the drop query.

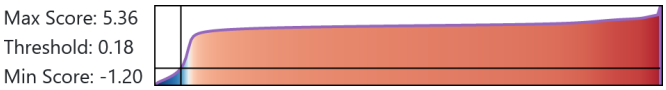

Fig. 8: Score graph over the sorted droplet query results with a set threshold to distinguish matches and non-matches.

ensemble (see Fig. 5), revealing four different patterns that we classify as drop, crown, crown-splash, and splash.

We then start by creating a query for each classified pattern, starting with the drop pattern. To build the drop query, we choose multiple positive example patches that contain droplets under different lighting conditions and without including much background in the selected patches. We also include two negative example patches that contain background to explicitly avoid similarity with the background in our query results. Figure 6 shows the final drop query (evidence for $(\mathbf{R_1})$).

Adjusting a query automatically triggers a re-computation of the scores across all samples regarding that query and provides the results within seconds (evidence for $(\mathbf{R_4})$). Fig. 7 shows the first few results for the drop query with its corresponding score graph below in Fig. 8. The score graph shows that a small percentage of samples have a small score (similar to a droplet), and a large percentage of samples have a high score (not similar). By default, the score threshold for labeling matches is set to one-third between the minimum and maximum scores in the query result. We reduce it slightly to reduce the number of potential false positive matches and to improve the quality of all views.

Creating the drop query also updates the clustering explorer (view D), which now shows the clusters that contain matches for the drop query. The previously empty petals of the flower glyphs in the projection view visualize the number of drop matches compared to the total number of elements in each cluster, see Fig. 9. Isoline-like contours additionally highlight

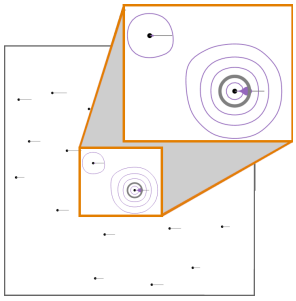

Fig. 9: Highlighted drop cluster. The isoline-like contours encode for density of query matches.

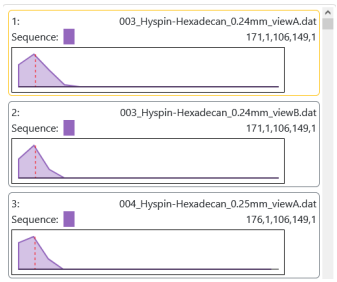

Fig. 10: Ensemble viewer drop timeline and sequence.

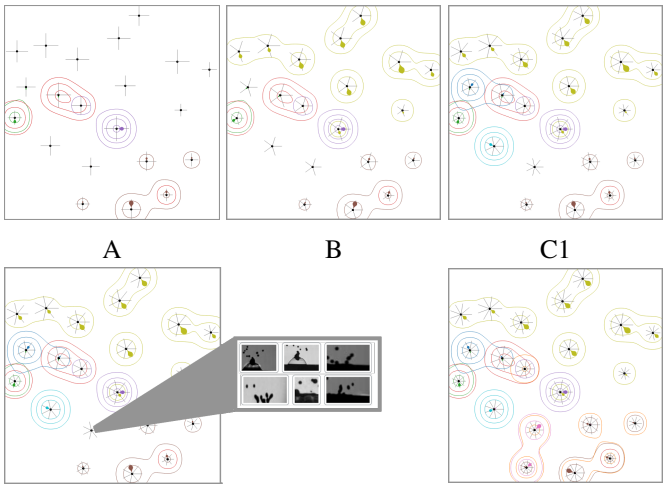

Fig. 11: Figures A, B, C1/C2 and D show how the projection view of the clustering explorer changes with progressing guided exploration (from A to D). As more patterns are captured via new queries, the view becomes more colorful. Uncolored regions indicate not yet captured patterns.

where clusters contain matches for the query.

Similarly, the ensemble explorer (view C) now shows in the individual experiment's timeline views, when drop matches occurs (see Fig. 10). The corresponding pattern sequences consist only of the drop pattern accordingly.

Following the guided exploration, we also create queries for the other so-far identified patterns: crown, crown-splash, and splash. Similar to before, this updates the projection view of the clustering explorer, as shown in Fig. 11(A). A large portion of clusters still has no matches at all. Inspecting the clusters in the upper half reveals that those include only background, for which we create another to improve the match coverage of the search space, resulting in Fig. 11(B). On the left side in the projection view are two more clusters with mostly no matches. In the first cluster, we find many patterns that we identify as bubbles, while in the second cluster, we see mostly patterns which we name jetting. We add a query for both patterns, resulting in Fig. 11(C1). Only

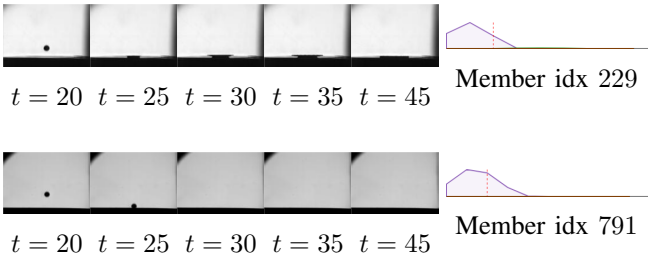

Fig. 12: Example time steps of member 229 (top) and member 791 (bottom) with the corresponding member's timeline for the drop query at the right. Both members have no crown matches detected and fall into the sequence that consists of only drop.

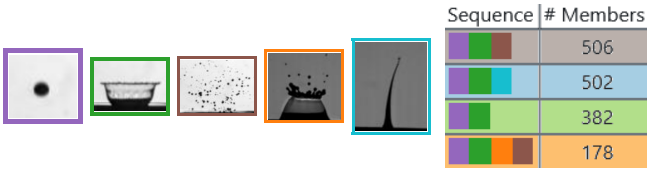

Fig. 13: The four largest groups of members with the same sequence for the queries: drop, crown, splash, bubble-splash, and jetting. The images show example matches for each of those queries, with the border colors representing the colors assigned to those queries. The background colors of the sequences in the right table are used to link with Fig. 14.

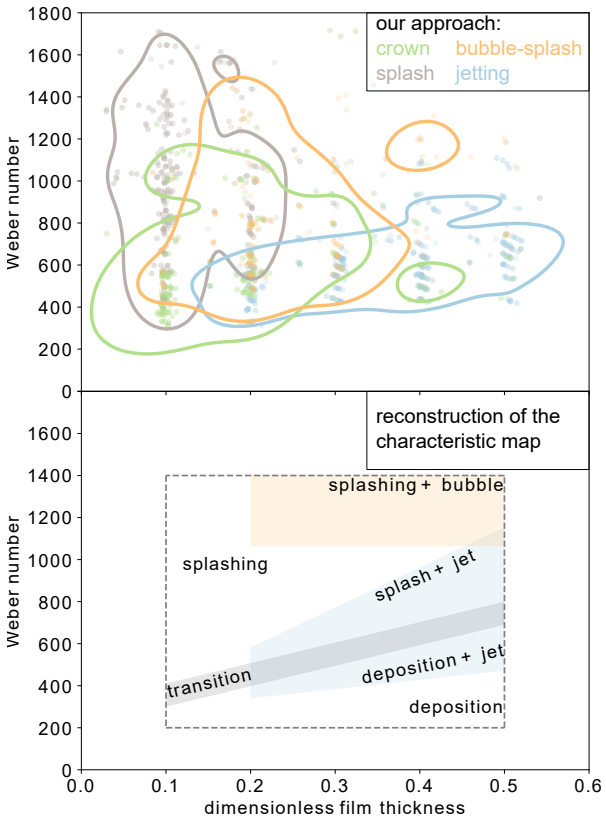

Fig. 14: The above figure shows our resulting characteristic map from the analysis. Individual members are plotted as circles. Isoline-like contours show a density distribution of the members per sequence. The color-coding corresponds to the most distinguishing pattern (top-right legend) for each of the four largest grouped sequences. The figure below shows the reconstructed characteristic map from a tedious manual analysis of domain experts [1](Fig. 1a).

one cluster remains with no isolines, and therefore, more unidentified patterns. Inspecting the cluster reveals bubble-splash patterns and relatively viscous or thick splashes, that we cannot clearly differentiate, see Fig. 11(C2). We still create an additional query for both patterns, which results in the majorly colored projection view in Fig. 11(D) and marks the stop for our guided exploration (evidence for ($\mathbf{R_2}$) for full pattern coverage).

We are now interested in how the different pattern occurrences relate to the parameter space. The sequence explorer (view E) shows the division of the whole ensemble into groups with identical pattern sequences and, thus, similar dynamics. However, the more queries, and thus, potentially false-positive matches exist, the more potentially different sequences are computed, which reduces the effectiveness of grouping individual sequences and yielding an informative pattern map as an overview for the ensemble's parameter-pattern relationships. Furthermore, we are not interested in patterns such as the background, and only keep the relevant and distinguishable pattern queries for this analysis: drop, crown, splash, bubble-splash, and jetting.

Interestingly, while we would expect to see a splash pattern in all of the droplet impact experiments, the sequence explorer now also reveals a small group of 57 members that contain no further patterns besides the drop. Inspecting this group shows that they either develop only a small crown, which is too small to be detected by our model, or none at all, as shown in Fig. 12. However, we determine this group as outlier and focus only at the larger groups in the sequence explorer. This leaves us with four large groups of clearly different pattern sequences and impact dynamics, which we illustrate in Fig. 13, and which we can now relate to the ensemble's parameter space by using the linked parameter space visualization.

The linked two-dimensional parameter space scatterplot of the droplet impact experiments ensemble plots the experiments by their corresponding *dimensionless film-thickness* and *Weber number*, see Fig. 3 (E). By hovering over the individual groups of pattern sequences, we can easily inspect where their member's associated parameters are located in the parameter space, allowing us to build a mental characteristic map. It shows that most experiments with the same pattern sequence also share a close region (range of parameters) in the parameter space, and regions of other pattern sequences do generally not overlap. This interactive visualization provides us with the first insights into the composition and characteristics of the droplet impact parameter space (evidence for ($\mathbf{R_3}$)), which we now use in further analysis.

Finally, we export the final state of our analysis and plot the members of the four largest sequences color-coded by their sequence, to build a static visualization of our interactively built characteristic map. Figure 14 shows our (top) charac-

teristic map side-by-side to a reconstruction of the manually built characteristic map from domain experts (bottom) from [1](Fig. 1a).

Comparing both maps reveals a number of similarities and differences, which can be explained as follows. The observed differences are mainly due to two factors. First, the ensemble used in our case study covers a broader parameter space and includes additional experiments compared to the original characteristic map. Second, our analysis emphasizes a slightly different set of patterns: crown, splash, bubble-splash, and jetting. In contrast, the original map focuses on splash, deposition (i.e., secondary droplets), and transition behaviors, with bubble and jetting highlighted as additional features. Nevertheless, our approach effectively identifies regions associated with jetting and splashing phenomena, and partially captures features related to bubble formation. This outcome was achieved in a case study conducted by non-domain experts in approximately thirty minutes.

B. Expert Interview: Droplet Impacts

We conducted an expert interview for a qualitative evaluation of our visual approach. The expert is a senior researcher with over ten years of experience working with spatiotemporal ensemble data, including droplet impact experiments. According to the expert, one of the main challenging aspects in the domain when working with such ensembles is the identification and classification of the pattern in the data. This task typically involves a tedious manual analysis with a "*huge time-consuming effort*" of "*several weeks.*"

The interview began with a brief overview of our approach and the dataset. We then asked a few general questions to better understand the expert's background and experience. The expert indicated limited experience with ML techniques for spatiotemporal ensemble analysis, so we provided a concise summary our self-supervised model, training procedure, and pre-processing steps. We then introduced the concept of pattern queries and query matches, followed by a walkthrough of the approach's coordinated views using the droplet impact experiments as example. During this walkthrough, the domain expert raised a few questions regarding dimensionality reduction techniques related to our projection view, which we then elaborated on.

After the initial walkthrough, we discussed the key aspects of our visual approach. The domain expert commended the unbiased analysis approach of the data for classifying patterns. Unlike manual analysis, where expectations about experiment parameters may influence the pattern identification, our model does not know about the parameters, allowing for a more objective pattern identification.

We then conducted a joint analysis similar to the case study in Section V-A, and collected feedback during and after this analysis. The overall expert's response was very positive. The domain expert was impressed by the speed of the implementation, such as querying and visualizing identified patterns in the ensemble during the interactive analysis. Furthermore, the domain expert appreciated the overview of the ensemble explorer, calling it to make it "*much easier*" for conducting analysis. When asked if the workflow and goals aligned with their typical process, the expert confirmed it. The expert also acknowledged the interactively generated "*regime map*" (characteristic map) and stated that this analysis is "*much faster*" compared to the manual analysis of each experiment. However, they noted that the regime map lacked structure and suffered from visual clutter. We note that at the time of the interview, our visual approach lacked interactive filtering of the pattern sequences in the parameter space visualization, leading to the noted visual clutter. This feature which was later implemented in response to the interview feedback. Overall, the expert expressed that they are amazed about "*how easy it is to handle*", that the "*visual encoding is easy to understand*", and that such a visual approach is something they "*would like to work with*" in future analyses.

Finally, we asked the expert for constructive feedback to further improve our visual approach. The expert suggested adding sub-querying capabilities, which means to add secondary queries on only the matches of a previous query. This would allow them to further subdivide the query results for a crown pattern by the crown's shape, such as a pyramidal, rectangular, or v-shape. Next, the expert also suggested improving the separation capabilities of the metric between crowns and bubbles, where a few false-positive bubbles were returned as crown matches during the analysis. As a final suggestion, the expert expressed that filtering the parameter space view such that the regions of individual sequences can be interactively explored, would be helpful to achieve an overview parameter-pattern relations in the whole ensemble. We implemented this final suggestion after the interview, which led to the final version of QVis.

In a brief follow-up interview, the expert confirmed the usefulness of the enhanced parameter space visualization. The expert then also suggested that our final approach could also guide the experiment process by identifying underrepresented regions, areas with high pattern variance or unexpected results in the parameter space, for which more experiments should be conducted.

Overall, we found the expert study evaluation provided valuable feedback for our visual approach's improvement, validated our requirements, and highlighted opportunities for further improvement and future work.

C. Expert Interview: Turbulent Flow

To validate the requirements and applicability of our visual analysis approach, we conducted a second expert interview. The expert has over three years of experience working on fundamental research of flow phenomena in porous media, such as "turbulent pumping" and vortex shedding. The expert works with experimental fluid data captured via particle image velocimetry (PIV).

During the interview, the expert presented us with two key challenges they face in their typical data analysis workflow. First, managing the large data volumes generated by high-speed imaging (200–600 GB per dataset ($\mathbf{R_4}$)). Second, the significant manual effort which is involved in analyzing time-resolved data ($\mathbf{R_2}$). While they mentioned that they have

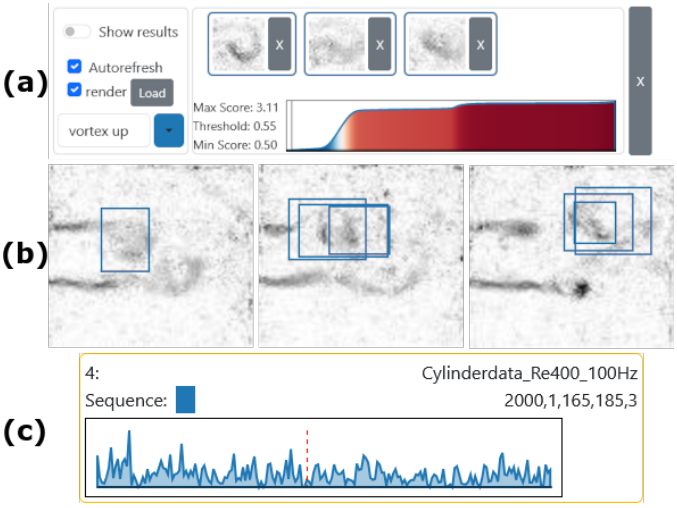

Fig. 15: Query and visualized matches for a vortex shedding dataset. The query contains three positive examples of clockwise rotating vortices (a). Our approach then manages to track and visualize similar vortices in the data (b) and when they occur in the corresponding timeline view (c).

developed a custom algorithm for vortex center detection, which they implemented in their data analysis pipeline, they noted that it cannot be used for other phenomena, requiring new algorithms and development effort for each case.

Their typical analysis begins with automated image processing of the PIV images to enhance the visibility of tracer particles and compute velocity and temperature features of the flow. Subsequently, image sequences are analyzed manually or with their custom algorithm. A typical workflow involves visually inspecting videos to assign physical quantities (e.g., Reynolds number, vertical transport) to time steps. These quantities indicate when flow features like vortices develop. These features are then revisited and manually identified and tracked when and where they appear, allowing subsequent analysis (e.g., computing a vortex shedding frequency) and relating them to experimental parameters such as inflow velocity or temperature.

To the question of what their ideal analysis tool should look like, they answered that they envision a tool where one could load raw video data, provide a phenomenon of interest ($\mathbf{R_2}$), and immediately receive a list of corresponding occurrences in time and space ($\mathbf{R_4}$). These occurrences should preferably be already visualized within the tool.

We then demonstrated our prototype using a vortex shedding dataset familiar to the expert. Our approach managed to identify vortices throughout the entire recording based on a query of just three examples, and visualize the identified vortex occurrences along a timeline (see Fig. 15 (c)). The expert found the interface intuitive and appreciated the arrangement of the views for query creation, result inspection, and timeline visualization. While the clustering and projection views initially caused confusion, they were quickly understood after a brief explanation. However, we noted that these views might offer more value in datasets containing a broader variety of patterns and less pattern periodicity.

Fig. 16: Example images of the synthetic dataset at different time steps.

The expert suggested storing and reusing queries across other datasets, a functionality already supported in our approach, assuming the similarity metric remains valid. However, significantly different datasets require retraining the model first.

Overall, this interview provided essential insights into the practical challenges faced in analyzing spatiotemporal experimental data and validated the core requirements and design of our visual analysis approach. The expert confirmed the relevance and usefulness of query-based pattern search and underlined its potential to significantly accelerate the discovery and understanding of flow phenomena in complex datasets.

D. Generalizability: Synthetic Data & Art Gallery Video

We evaluate our approach's multiscale querying ability with a controlled synthetic dataset and provide an outlook regarding its generalizability using an ensemble of first-person vision (FPV) video recordings containing art exhibits.

Synthetic Dataset: We created a synthetic video dataset of just one ensemble member containing three different patterns at three spatial scales. This dataset provides ground-truth labels for both the occurring pattern type and scale, enabling a controlled comparison of our multiscale approach against the fixed-size alternative. The patterns are the shapes circle, rectangle, and triangle. The scales are small, medium, and large. The dataset has 2000 time steps and a spatial extent of 1000×1000 pixels. To generate the data, we sampled 165 instances of each shape, where each instance was sampled from one size range: $150 - 250$ for small, $350 - 450$ for medium, and $550 - 650$ for large. We placed the instances at random locations in the data, but avoid overlap (see examples in Fig. 16). We then trained two identical model architectures: one fixed-size variant with a fixed input size, and one multiscale variant with variable input size.

To evaluate how well each model retrieves similar patterns at different scales, we then issued queries containing random samples for the different available shapes (circle, rectangle, and triangle), and analyzed which of the available patterns are closest to the respective queries. Thus, using any circle pattern as query example to compare with the other patterns in the search space should result in other circle patterns (at any scale) to have the lower scores than rectangle or triangle patterns. Since the fixed-size variant cannot process different input size, we rescaled the patterns in the search space to the fixed input size of the model (512×512 pixels), while the multiscale variant can process each pattern in the search space by its original size.

To quantify the retrieval accuracy, we calculate the precision@k [43] metric. The precision@k metric describes the

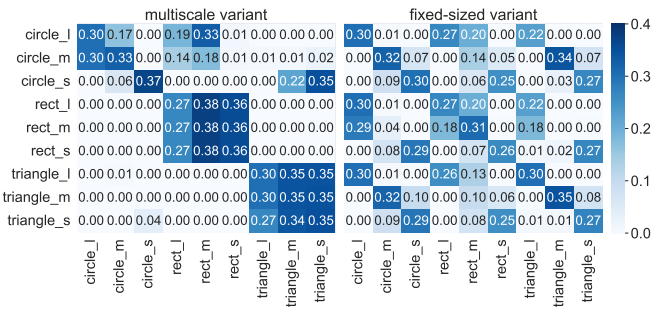

Fig. 17: The precision@k matrix per shape-size combination for the synthetic dataset for the multiscale and fixed-size variant. The row represents the query elements, and the columns represent the percentage of retrieved elements. We queried with $p = 4$ elements of the same label and calculated the percentage of retrieved elements per label out of $k = 165$, which is the total amount of available elements per shape.

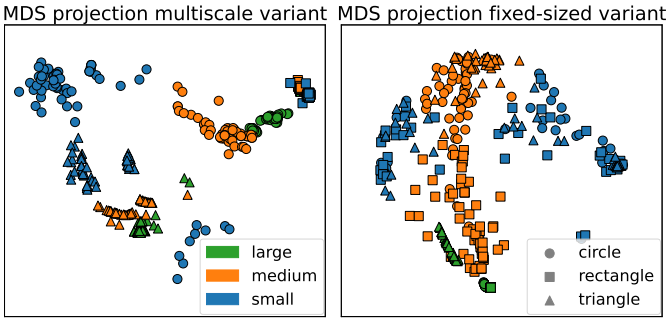

Fig. 18: The MDS projection for multiscale (left) and fixed-size (right) variant of the labeled patches. The marker shape corresponds to the captured pattern shape in the labeled patch and color encodes the size.

number of correctly retrieved patterns out of the top k results of the query.

The results for the precision@k metric for both variants is shown in Fig. 17. The multiscale variant retrieves the same shape across all sizes independent of the input size, except for the small circle. The fixed-size variant retrieves multiple different shapes but those of similar size only. It fails to identify similarities between similar patterns of different sizes.

We visualize the differences between both models by providing the outcome of multidimensional scaling (MDS) [44] of the dataset in Fig. 18, using the respective models as similarity metrics between the pattern's patches. Here, the distance between points encodes for the similarity among them. We can see that the shapes form clusters regardless of size when using the multiscale version, though, which is not the case for the fixed-size variant. This example further provides evidence that our approach fulfills $(\mathbf{R_1})$.

Art Gallery Video Ensemble: We applied our approach to a FPV video ensemble dataset [45] to provide an outlook regarding generalizability of our approach. The dataset contains 27 video recordings in which the visitors move throughout a small art gallery of 5 paintings, while a camera records their point-of-view. In 12 such recordings, the visitor started at the first painting and moved clockwise from painting to painting. In 12 other recordings, they moved in counterclockwise order. The order of the remaining 3 recordings was randomized.

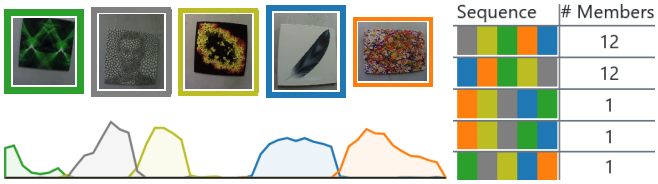

Fig. 19: The images above show the positive example patch of each query, with the image border color representing the color we assigned to the query itself. The grouped sequences reveal the two primary directions of camera movement and three outliers with randomized movements.

Using our approach, we can easily divide the recordings according to the visitor's movements. We first trained the model on the dataset and performed the pre-processing steps similar to the case study in Section V-A. We then created a query for each painting in our visual approach and inspected the results, see Fig. 19. The timeline views allow to easily identify each visitors movement order through the art gallery by highlighting when specific paintings were in their point-of-view. Furthermore, the sequence explorer nicely divides the ensemble into clockwise and counterclockwise moving camera videos, while the three outliers are presented in their separate groups, but also revealing their order of camera movement as compact pattern sequence.

VI. Discussion and Limitations

In this section, we discuss the algorithmic and visual scalability [46] of our approach, as well as limitations regarding the trustworthiness of its results.

Computational Scalability: From a computational perspective, our approach consists of three major parts: *training the model* to yield a similarity metric, *sampling and pre-processing*, and the interactive analysis (primarily *querying*).

Training the model largely depends on the maximum patch size and the overall data complexity of the ensemble, which affects the time until convergence. However, a single encoding step can be considered to be of constant complexity regarding the maximum patch size.

The *sampling and pre-processing* step depends on the number of samples required to sufficiently cover the patterns in the ensemble. Once the average sampling density is determined – likely influenced by factors such as spatial and temporal dimensionality and the expected pattern sizes and occurrence frequency – the total number of samples should scale linearly with the dataset size.

Querying consists of (a) scoring each sample and (b) sorting the samples by score. (a) requires computing the pairwise distances between the number of query examples e and samples n, which follows $\mathcal{O}(ne)$. Sorting n elements (here, the resulting scores) can generally be considered to be of $\mathcal{O}(n \log n)$ complexity and is, therefore, also the complexity of a single query.

Visual Scalability: The views mostly show aggregated information and are not directly related to the dataset size. However, the visual scalability mainly depends on the number of queries and number of ensemble members. Many simultaneous queries can introduce visual clutter in the ensemble and clustering explorer views, and negatively impact the pattern sequence computation. A very high number of ensemble members could introduce visual clutter in the parameter space visualization, requiring adjustments for larger number of members.

Trustworthiness of Results: We have shown that our presented visual approach allows us to explore and analyze large spatiotemporal ensemble datasets effectively. However, the workflow relies on many interacting components and steps where uncertainties can be induced and propagated throughout the analysis.

The most critical component in the analysis is the model and the associated similarity metric. If the model cannot properly distinguish different patterns, it will negatively impact the clustering, querying, and all other analysis steps. We have provided various approaches to help evaluate the model's effectiveness and build trust in its results, such as interactively exploring the clustering or query results via the visual approach. However, even with a robust multiscale similarity metric, the quality of analysis remains sensitive to other factors.

Other sensitive factors in the analysis include the choice of projection algorithm, the number of visualized clusters, query definitions and thresholds, the binning parameters for sequence extraction, and the sampling strategy for patches. Due to this, we keep most of these factors interactive so that they can be adjusted to the current dataset and goals during the analysis. The choices regarding the projection algorithm and sampling approach may have to be considered separately and potentially adjusted before the analysis.

Additionally, numerical problems in the analysis may arise due to variable time-discretization among ensemble members, such as changing frame rates in video recordings, which could lead to over- or under-retrieval of patterns in parts of the ensemble. This problem could be addressed by adapting the sampling algorithm or normalizing the ensemble's spatiotemporal discretization in a pre-processing step.

Finally, while our query-based analysis allows to query similar patterns at different scales, it limits us in cases where this property is not desired and distinctions based on the size of the patterns are intentional.

VII. Conclusion and Future Work

With QVis, we introduce a considerable step forward in the interactive analysis of large spatiotemporal ensembles through flexible, multiscale pattern querying. By extending a learned similarity model to support variable-sized inputs and enabling users to define multiple pattern queries directly within visualized data, our approach allows domain experts to efficiently identify, retrieve, and compare complex dynamic behaviors across ensembles. Our query-based exploration helps reveal missing or underexplored patterns, supports richer multiscale analysis, and enables users to relate patterns to experimental parameters via linked, coordinated views. QVis provides a dynamic, pattern-based signature for each ensemble member and facilitates interactive navigation through the ensemble's parameter space, significantly reducing the manual effort required previously. Expert feedback confirms that our system reproduces complex insights more efficiently and intuitively than existing manual workflows.

In future work, we plan to extend our approach in several directions. First, we aim to support more general input data (e.g., including multiple variables or non-uniform grids), and explore additional—potentially domain-specific—feature extraction approaches. Second, as suggested by the expert during the interview, we intend to support sub-querying, enabling users to explore patterns at varying levels of granularity and construct hierarchical overviews of complex behaviors. This could lead to more expressive abstractions, such as representations of ensemble dynamics akin to scene graphs for better summarizing and comparing complex data. In addition, we plan to conduct a systematic evaluation of different pooling strategies and parameterizations of the underlying model architecture, and investigate their effects on similarity learning performance, especially for ensembles with highly variable spatial or temporal resolutions. Finally, we aim to scale QVis to larger ensembles—such as those in climate and weather simulation—which involve higher dimensional parameter spaces and longer temporal scopes; to support these cases, future work will explore scalable querying strategies and abstraction techniques.

Acknowledgments

The authors thank Gleb Tkachev for fruitful discussions. Large Language Models (LLMs) have been used to improve readability, reduce typos, and correct grammatical mistakes.

Funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2075 - 390740016. We acknowledge the support of the Stuttgart Center for Simulation Science (SimTech). Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - Project-ID 327154368 - SFB 1313, project D01. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 251654672 – TRR 161, project A08.

References

[1] A. Geppert, A. Terzis, G. Lamanna, M. Marengo, and B. Weigand, "A benchmark study for the crown-type splashing dynamics of one- and two-component droplet wall–film interactions," *Experiments in Fluids*, vol. 58, p. 172, 2017. [Online]. Available: https://doi.org/10.1007/s00348-017-2447-2

[2] A. Geppert, D. Chatzianagnostou, C. Meister, H. Gomaa, G. Lamanna, and B. Weigand, "Classification of impact morphology and splashing/deposition limit for n-hexadecane," *Atomization and Sprays*, vol. 26, no. 10, pp. 983–1007, 2016.

[3] M. Hesselmann, R. Fechte-Heinen, L. Mädler, M. Steinbacher, and A. Toenjes, "Smart-alloying – liquid in-situ re-alloying in additive manufacturing," *Additive Manufacturing*, vol. 80, p. 103988, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214860424000344

[4] Q. Wang, Z. Chen, Y. Wang, and H. Qu, "A survey on ml4vis: Applying machine learning advances to data visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 12, pp. 5134–5153, 2022.

[5] C. Wang and J. Han, "Dl4scivis: A state-of-the-art survey on deep learning for scientific visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 8, pp. 3714–3733, 2023.

[6] G. Tkachev, S. Frey, and T. Ertl, "S4: Self-supervised learning of spatiotemporal similarity," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 12, pp. 4713–4727, 2022.

[7] A. Endert, W. Ribarsky, C. Turkay, B. W. Wong, I. Nabney, I. D. Blanco, and F. Rossi, "The state of the art in integrating machine learning into visual analytics," *Computer Graphics Forum*, vol. 36, no. 8, p. 458–486, Mar. 2017. [Online]. Available: http://dx.doi.org/10.1111/cgf.13092

[8] D. S. Ebert, A. Reinert, and B. Fisher, "Visual analytics review: An early and continuing success of convergent research with impact," *Computing in Science & Engineering*, vol. 23, no. 3, pp. 99–108, 2021.

[9] J. Wang, S. Hazarika, C. Li, and H.-W. Shen, "Visualization and visual analysis of ensemble data: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. PP, pp. 1–1, 07 2018.

[10] M. Raji, A. Hota, R. Sisneros, P. Messmer, and J. Huang, "Photo-Guided Exploration of Volume Data Features," in *Eurographics Symposium on Parallel Graphics and Visualization*, A. Telea and J. Bennett, Eds. The Eurographics Association, 2017.

[11] F. Lekschas, B. Peterson, D. Haehn, E. Ma, N. Gehlenborg, and H. Pfister, "Peax: Interactive visual pattern search in sequential data using unsupervised deep representation learning," *Computer Graphics Forum*, vol. 39, no. 3, pp. 167–179, 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13971

[12] F. Zhu, Y. Pan, T. Gao, H. Walia, and H. Yu, "Interactive visualization of hyperspectral images based on neural networks," *IEEE Computer Graphics and Applications*, vol. 41, no. 5, pp. 57–66, 2021.

[13] K. Kurzhals, N. Rodrigues, M. Koch, M. Stoll, A. Bruhn, A. Bulling, and D. Weiskopf, "Visual analytics and annotation of pervasive eye tracking video," in *ACM Symposium on Eye Tracking Research and Applications*, ser. ETRA '20 Full Papers. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3379155.3391326

[14] Z. Wang, H.-P. Seidel, and T. Weinkauf, "Multi-field pattern matching based on sparse feature sampling," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 807–816, 2016.

[15] Y. Ye, R. Huang, and W. Zeng, "Visatlas: An image-based exploration and query system for large visualization collections via neural image embedding," *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–15, 2022.

[16] H. Wang, L. Yang, X. Rong, J. Feng, and Y. Tian, "Self-supervised 4d spatio-temporal feature learning via order prediction of sequential point cloud clips," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2021, pp. 3762–3771.

[17] M. Chu and N. Thuerey, "Data-driven synthesis of smoke flows with cnn-based feature descriptors," *ACM Trans. Graph.*, vol. 36, no. 4, jul 2017. [Online]. Available: https://doi.org/10.1145/3072959.3073643

[18] J. Han, J. Tao, and C. Wang, "Flownet: A deep learning framework for clustering and selection of streamlines and stream surfaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 4, pp. 1732–1744, 2020.

[19] W. P. Porter, Y. Xing, B. R. von Ohlen, J. Han, and C. Wang, "A deep learning approach to selecting representative time steps for time-varying multivariate data," in *2019 IEEE Visualization Conference (VIS)*, 2019, pp. 1–5.

[20] H. Gadirov, G. Tkachev, T. Ertl, and S. Frey, "Evaluation and selection of autoencoders for expressive dimensionality reduction of spatial ensembles," in *Advances in Visual Computing*, ser. Lecture Notes in Computer Science, G. Bebis, V. Athitsos, T. Yan, M. Lau, F. Li, C. Shi, X. Yuan, C. Mousas, and G. Bruder, Eds. Springer International Publishing, 2021, pp. 222–234.

[21] K. Huesmann and L. Linsen, "Similaritynet: A deep neural network for similarity analysis within spatio-temporal ensembles," *Computer Graphics Forum*, vol. 41, no. 3, pp. 379–389, 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14548

[22] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999, pp. 1150–1157 vol.2.

[23] E. Adelson, C. Anderson, J. Bergen, P. Burt, and J. Ogden, "Pyramid methods in image processing," *RCA Eng.*, vol. 29, 11 1983.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[25] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *CoRR*, vol. abs/2010.11929, 2020. [Online]. Available: https://arxiv.org/abs/2010.11929

[26] S. Yang and H. Yuan, "A customized multi-scale deep learning framework for storm nowcasting," *Geophysical Research Letters*, vol. 50, no. 13, p. e2023GL103979, 2023, e2023GL103979 2023GL103979. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2023GL103979

[27] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," 2016.

[28] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf

[29] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, "Voxel r-cnn: Towards high performance voxel-based 3d object detection," *arXiv:2012.15712*, 2020.

[30] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, "Activation functions in deep learning: A comprehensive survey and benchmark," *Neurocomput.*, vol. 503, no. C, p. 92–108, sep 2022. [Online]. Available: https://doi.org/10.1016/j.neucom.2022.06.111

[31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[32] N. Andrienko, G. Andrienko, and P. Gatalsky, "Exploratory spatio-temporal visualization: an analytical review," *Journal of Visual Languages & Computing*, vol. 14, no. 6, pp. 503–541, 2003, visual Data Mining. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1045926X03000466

[33] J. Kehrer and H. Hauser, "Visualization and visual analysis of multi-faceted scientific data: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 3, pp. 495–513, 2013.

[34] M. Sedlmair, C. Heinzl, S. Bruckner, H. Piringer, and T. Möller, "Visual parameter space analysis: A conceptual framework," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2161–2170, 2014.

[35] X. Chen, L. Shen, Z. Sha, R. Liu, S. Chen, G. Ji, and C. Tan, "A survey of multi-space techniques in spatio-temporal simulation data visualization," *Visual Informatics*, vol. 3, no. 3, pp. 129–139, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2468502X19300452

[36] M. Luboschik, M. Röhlig, A. Bittig, N. Andrienko, H. Schumann, and C. Tominski, "Feature-driven visual analytics of chaotic parameter-dependent movement," *Computer Graphics Forum*, vol. 34, no. 3, pp. 421–430, 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12654

[37] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," 2020.

[38] A. Novikov, "PyClustering: Data mining library," *Journal of Open Source Software*, vol. 4, no. 36, p. 1230, apr 2019. [Online]. Available: https://doi.org/10.21105/joss.01230

[39] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[40] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '07. USA: Society for Industrial and Applied Mathematics, 2007, p. 1027–1035.

[41] C. van Onzenoodt, P.-P. Vázquez, and T. Ropinski, "Out of the plane: Flower versus star glyphs to support high-dimensional exploration in two-dimensional embeddings," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 12, pp. 5468–5482, 2023.

[42] Gennady, F. Jean-Daniel, G. Carsten, K. Jörn, M. G. K. Daniel, and Andrienko, *Visual Analytics: Definition, Process, and Challenges*.

Springer Berlin Heidelberg, 2008. [Online]. Available: https://doi.org/10.1007/978-3-540-70956-5_7

[43] K. Järvelin and J. Kekäläinen, "Ir evaluation methods for retrieving highly relevant documents," in *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '00. New York, NY, USA: Association for Computing Machinery, 2000, p. 41–48. [Online]. Available: https://doi.org/10.1145/345508.345545

[44] M. A. A. Cox and T. F. Cox, *Multidimensional Scaling*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 315–347. [Online]. Available: https://doi.org/10.1007/978-3-540-33037-0_14

[45] M. Koch, K. Kurzhals, Q. Q. Ngo, and D. Weiskopf, "Dataset for NMF-based Analysis of Mobile Eye-Tracking Data," 2024. [Online]. Available: https://doi.org/10.18419/darus-4023

[46] G. Richer, A. Pister, M. Abdelaal, J.-D. Fekete, M. Sedlmair, and D. Weiskopf, "Scalability in visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 30, no. 7, pp. 3314–3330, 2024.

Ruben Bauer is a doctoral student at the University of Stuttgart, where he also completed his master's degree in computer science. His current research topics include combining machine learning with interactive visualizations for the analysis of spatiotemporal ensembles and parameter spaces.

Quynh Quang Ngo is a postdoctoral fellow at VISUS, University of Stuttgart, Germany. He received his Dr. rer. nat. Degree from the University of Münster, Germany, in 2020. His work focuses on conducting research in the field of Visualization and Computer Graphics. His recent research interests include Dimensionality Reduction, Data Transformation, Graph Layout, applying Machine Learning to Visualization, and Real Time Rendering.

Guido Reina is a senior lecturer at the Visualization Research Center of the University of Stuttgart. He received his Ph.D. degree in Computer Science in 2009 from the University of Stuttgart, Germany. His research interests include visual analysis of large-scale data, parallel methods, and sustainable research.

Steffen Frey received his PhD degree in computer science from the University of Stuttgart, Germany in 2014. He is an assistant professor at the Bernoulli Institute at the University of Groningen, Netherlands. His research interests are in visualization methods for increasingly large quantities of scientific data.

Michael Sedlmair is a professor at the University of Stuttgart and heads the VisVAR research group there. He received his Ph.D. degree in Computer Science in 2010 from the Ludwig Maximilians University of Munich, Germany. His research interests focus on Visualization/Visual Analytics, Human-Computer Interaction, and Augmented/Virtual Reality.