J Vis
https://doi.org/10.1007/s12650-020-00625-2

REGULAR PAPER

Li Liu · **Deborah Silver** · **Karen Bemis**

Visualizing events in time-varying scientific data

Received: 27 March 2019 / Revised: 22 September 2019 / Accepted: 3 January 2020

Abstract In time-varying scientific datasets from simulations or experimental observations, scientists always need to understand when and where interesting events occur. An event is a complex spatial and temporal pattern that happens over a course of timesteps and includes the involved features and interactions. Event detection allows scientists to query a time-varying dataset from a much smaller set of possible choices. However, with many events detected from a dataset, each spanning different time intervals, querying and visualizing these events pose a challenge. In this work, we propose a framework for the visualization of events in time-varying scientific datasets. Our method extracts features from a data, tracks features over time, and saves the evolution process of features in an event database where a set of database operations are provided to model an event by defining the stages or individual steps that make up an event. Using the feature metadata and the event database, three types of event visualizations can be created to give a unique insight into the dynamics of data from temporal, spatial, and physical perspectives and to summarize multiple events or even the whole dataset. Three case studies are used to demonstrate the usability and effectiveness of the proposed approach.

Keywords Scientific visualization · Time-varying data · Feature tracking · Event detection · Illustrative visualization

1 Introduction

Scientific phenomena such as combustion, ocean currents, and hurricanes are inherently time-varying processes. The ability of scientists to visualize time-varying scientific data is essential to ensure correct interpretation and analysis, provoke insights, and communicate those insights to others (Ma 2003). Visualization of time-series data (i.e., time-varying data) has long been of interest to the visualization community, and various methods have been used including direct volume rendering, data compression, and feature-based techniques. Among these methods, automated feature-based techniques that depend on feature extraction and tracking allows scientists to focus on selected regions-of-interest (i.e., features), both in space and time, which can lead to a better understanding of the underlying dynamics. With multiple features presenting in a dataset, scientists are interested in understanding how features interact with each other and

L. Liu (✉)
School of Computer Science and Technology, Soochow University, Suzhou, China
E-mail: lliu6819@suda.edu.cn

D. Silver
Department of Electrical and Computer Engineering, Rutgers, The State University of New Jersey, New Brunswick, USA

K. Bemis
Department of Marine and Coastal Sciences, Rutgers, The State University of New Jersey, New Brunswick, USA

Published online: 16 January 2020

the fundamental questions of where, when, and how "interesting things" occur. Typically, a scientist hypothesizes about what constitutes important events and what defines these events, then illustrates the events in a publication or presentation as a sequence of timesteps extracted from a large number of timesteps. How are these sequences of timesteps chosen? Traditional approaches involve manual inspection of all timesteps to find significant events, pick out key timesteps that can describe the event, and construct a composite image or sequence of images to visualize and illustrate the events-of-interest. While resulting in effective illustrations, these approaches depend critically on an expert's unquantified judgment and can easily miss some instances of events in datasets. Hypothetically, an automated analysis system could search through a dataset to find the sequences of timesteps corresponding to the events-of-interest.

In a time-varying dataset, an event is a complex spatial and temporal pattern that happens over a course of timesteps and includes the involved features and interactions. Automated detection of events from time-series data based on quantitative criteria defining the events allows users to query the data from a much smaller set of possible choices and thus become an active research area in the visualization community. Approaches of event detection include clustering, machine learning, and semantic-based techniques using rules or graphs. Techniques are mostly concerned with the correspondence problem which involves correlating objects from one timestep to the next. An example is shown in Fig. 1, where two "merge-and-split" events are detected from a simulation of turbulent vortex and are visualized as event actors (i.e., features that participate in an event) in the timesteps where the events take place. Although event detection enables scientists to focus on critical timesteps and work with those timesteps that are of interest, the amounts of features and timesteps, the varying locations, and the inconsistent time intervals pose another challenge to visualize the detected events. For example, one event could occur over a hundred timesteps, while another event takes only ten timesteps. It is a difficult task to look at a hundred timesteps to see an event, let alone viewing multiple ongoing events that may overlap with each other. Furthermore, there could be a lot of "identical" events all occurring at different intervals. Scientists want to ask: What does a typical event look like? How do the features and events interact with each other? How to create a single visualization summarizing all of the detected events? Scientists need a tool to first define the stages or individual steps that make up an event based on their research requirements, then search for the instances of the modeled event from a time-varying scientific dataset, and finally visualize these events in a variety of ways. With a collection of events generated from the event detection, it is tempting to query the "event metadata" in multiple ways to produce tailored illustrations, such as the typical development of an event, the average pattern of a stage of the event, or a summary of all events. In particular, this permits a scientist to specify a criterion by which keyframes (or key timesteps) can be chosen to represent the stages of an event or the motion of a feature in space or the different occurrences of events.

In this work, we propose a framework for the computation, modeling, and visualization of events in time-varying scientific datasets. Our method extracts features from a data, tracks features over time, and saves the evolution process of features in an event database where a set of database operations are provided to model an event by defining the stages or individual steps that make up an event. Using the feature metadata and the event database, three types of event visualizations are created to give a unique insight into the dynamics of data from different perspectives. Three case studies are used to demonstrate the usability and effectiveness of the proposed approach.

The contributions of this work are trifold. Firstly, the introduction of an event database along with a set of database operations allows scientists to easily define an event according to their research interests and to

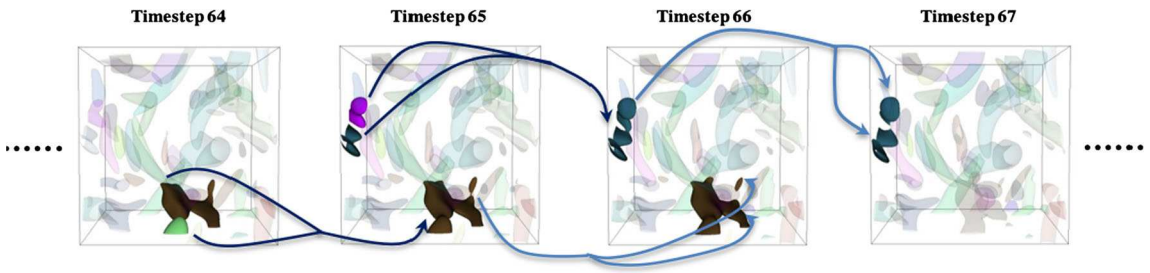

Fig. 1 Two "merge-and-split" events are detected in the timesteps 64–67 from a simulation of turbulent vortex. The features which take part in the events are presented in unique solid colors, while other features turn to transparent

query the instances of the targeted event from a time-varying scientific dataset. Secondly, three types of event visualizations are proposed to provide a unique insight into the dynamics of data from temporal, spatial, and physical perspectives and to summarize multiple events or even the whole dataset into concise visualizations. Last but not least, the proposed visualizations highlight the individuality and the dynamic aspects of events and the connections between events by combining the style of illustrative depiction with feature-based visualizations.

The remainder of this paper is organized as follows: Sect. 2 reviews the related work, and Sect. 3 gives an overview of the proposed framework of visualizing events from time-varying scientific data. The two major parts of the framework: feature-based event computation and event visualization, are introduced in Sects. 4 and 5, respectively. In Sect. 6, the usability and effectiveness of the event visualizations are demonstrated by three case studies. Finally, Sect. 7 summarizes this work and discusses potential future work.

2 Related work

Previous research related to the methodologies adopted in this paper includes visualization of time-varying volume data, event detection, and illustrative visualization.

2.1 Visualization of time-varying volume data

Visualization of time-varying volume data has long been of interest to the visualization community. Methods generally include reducing the storage size of data to make it more manageable, such as compression, temporal sub-sampling, and contouring (Shen et al. 1999; Ma and Shen 2000), preselecting transfer functions for direct volume rendering (Pfister et al. 2001; Jankun-Kelly and Ma 2001), interactive hardware-accelerated volume rendering (Ellsworth et al. 2000; Lum et al. 2002), and computing features and tracking them (Samtaney et al. 1994; Silver and Wang 1997; Reinders et al. 2001; Ji et al. 2003; Ji and Shen 2006; Caban et al. 2007; Muelder and Ma 2009; Clyne et al. 2013). Among these methods, automated feature computation techniques using domain knowledge can significantly reduce the amount of data that needs to be visualized by focusing on just those regions-of-interest. In this paper, we adopt the feature-based techniques that depend on feature extraction and tracking. The process known as feature extraction and tracking consists of identifying interesting regions or coherent structures of a dataset (i.e., features), automatically searching for these features, and tracking these features as they move and interact from one timestep to the next.

2.2 Event detection

Detecting events from a time-varying data has been studied in many fields under the names event detection, event recognition, activity detection, or activity recognition and has mostly been applied to 2D vision. The techniques can be broadly classified into either machine learning techniques using Bayesian techniques, hidden Markov models, and conditional random fields as found in the review papers (Albanese et al. 2008; Lavee et al. 2009; Poppe 2010; Aggarwal and Ryoo 2011), and semantic techniques using rules or graphs. Compared with other event detection methods, semantic-based techniques are more interactive and give the control to the scientist to directly specify the events from a data. Reinders et al. (2001) used a rule-based semantic event detection method to first define a set of atomic events (i.e., actions) then search the defined actions by iteratively comparing the current timestep to a reference timestep from the tracking history. In this work, the event detection extends the method to capture the whole evolution process of features. On the other hand, Ozer et al. (2014) proposed a graph-based semantic event detection method under the name "activity detection for scientific visualization." This method used an enhanced Petri Net formalism (Lee and Shen 2009) to capture the modeling and searching of complex events (i.e., activities) in scientific data. Considering the direct specification of rules fits consistently and smoothly with both changing queries interactively and viewing results interactively, we adopt Petri Net as a graphic way of querying the event database and modeling an event.

2.3 Illustrative visualization

Illustrative visualization focuses on generating more abstract or expressive visualizations typically inspired by works from artists and illustrators (Rautek et al. 2008). Hand-drawn images have been used for centuries to explain and depict natural phenomena, and the main goal of illustrative visualization is to mimic these images by computing and abstracting the data automatically. The resulting visualization is a pictorial abstraction of the datasets as opposed to a direct rendering. Illustrative visualization of time-varying data is especially helpful as it is difficult to convey motion without animation, and watching many time-frames is time-consuming. In time-varying volume data, the process of illustrative visualization often relates to using metadata to enhance perception, manage visibility, and provide focus in renderings of the data or its events (e.g., Joshi and Rheingans 2005, 2008; Joshi et al. 2009; Lu and Shen 2008; Hsu et al. 2009; Born et al. 2010). A variety of techniques including visual effectiveness enhancement (primarily focused on non-photorealistic rendering (Rheingans and Ebert 2001), visibility management (or smart visibility) (Viola and Gröller 2005), focus + context (Viola et al. 2005; Hauser 2006), and interactive visual storytelling (Wohlfart and Hauser 2007; Ma et al. 2012) are useful for scientific visualization. Many of these techniques are summarized in Brambilla et al. (2012). The motion of features over time can be depicted using a combination of tracking, variable transfer functions, and artistic renderings. Compositing of multiple timesteps combined with selective colormaps to control visibility has been used effectively to convey the evolution of volume features (Joshi et al. 2009; Hsu et al. 2009). In many cases, compositing in combination with selective visibility management can accurately convey the motion or evolution of flow features. Many of the techniques to produce an illustrative visualization can be used to enhance the event visualizations discussed in this paper.

3 Approach overview

An overview of the proposed framework of visualizing events in time-varying data is illustrated in Fig. 2. The framework is divided into two major parts: feature-based event computation and event visualization. In the feature-based event computation part, the workflow includes four steps: (1) *feature extraction* identifies regions-of-interest or coherent structures of a dataset (i.e., features) and automatically extracts these features from the background at each timestep using a variant of segmentation; (2) *feature tracking* matches the extracted features from a timestep to the next using volume overlap, pattern matching, or feature-vector matching; (3) *event detection* isolates the evolution process of individual features as events from the tracking history and further saves the events in an event database including an event table and a set of event metadata; (4) *event modeling* provides a bunch of database operations which allow scientists to model an event by defining the stages and individual steps that make up an event and to search for the instances of the modeled event from the event database. In the event visualization part, three types of event visualizations are created from the feature metadata and the event database to enable a structural understanding of a particular event or give a concise summary to multiple or all events in the event database.

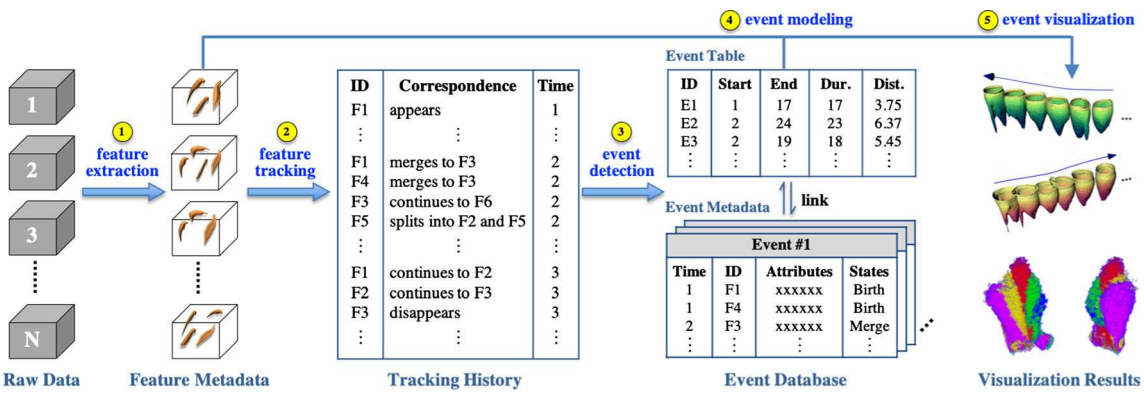

Fig. 2 An overview of the proposed framework of visualizing events from time-varying scientific data. The framework can be divided into two major parts: feature-based event computation and event visualization. The feature-based event computation includes four steps: feature extraction, feature tracking, event detection, and event modeling

4 Feature-based event computation

4.1 Feature extraction

Feature extraction (or feature segmentation) involves isolating regions-of-interest from the background region as independent objects. Given a threshold value, a region-growing algorithm starts with a seed node (local maximum value) and iteratively scans each voxel of data. Each seed point is assigned a unique object ID. The algorithm continues to recursively search all adjacent voxels of the chosen seed point, and those with intensity greater than the threshold are labeled as belonging to the current feature. All searched voxels are marked as visited. If the dataset contains more than one variable, the "non-primary variables" (i.e., the variables that do not take part in the region-growing process) are written at every voxel of the features. The current feature stops growing when voxels with values less than the threshold are encountered in all growing directions. For each extracted feature, a set of attributes of each extracted object, such as volume, centroid, and bounding box, is calculated for each resulting plume object. To reduce the rendering cost of visualization, the Marching Cubes algorithm (Lorensen and Cline 1987) is applied to compute an iso-value surface (i.e., isosurface) of each feature. Using isosurfaces, a volume object can be rendered by a simple polygonal. The region-growing process is completed when all voxels in the data are visited. The results of the feature extraction step, which is called feature data, include the set of voxels, isosurfaces, and attribute of the extracted features at each timestep.

4.2 Feature tracking

After the feature extraction step has been performed, it is not yet known which features are the same object in different instances of time. Thus, there is no description of motion and evolution of features (Reinders et al. 2001). Feature tracking is used to correlate features in each timestep over time to determine the occurrences of the same features in the consecutive timesteps. Here, we adopt an overlap-based feature tracking algorithm (Silver and Wang 1997). In the life cycle, a feature may go through five states (or instances): *birth* a feature in the current timestep does not correspond to any feature in the previous timestep; *continuation* a feature from the previous timestep corresponds to just one feature in the current timestep; *merge* two features from the previous timestep correspond to one feature in the current timestep; *split* a feature from the previous timestep corresponds to two features in the current timestep; *death* a feature from the previous timestep does not correspond to any features in the current timestep. Once the feature tracking has been completed, a correspondence list (tracking history) is created to record the evolution of features in every timestep as shown in Fig. 2.

4.3 Event detection

Tracking is only the first step in studying the motion and evolution of features in a time-varying dataset. With numerous features spanning dozens or even hundreds of timesteps, a method of event detection is needed to isolate the evolution process of every individual feature from the tracking history. An event can be thought of as a sequence of timesteps during which the features and/or their interactions pass through a series of states (or actions). The algorithm scans the tracking history and searches the "death" label in every timestep. Whenever a "death" label is found or a scan reaches the final timestep of the data, the algorithm will iteratively trace the corresponding feature back to the previous timesteps until a "birth" label is found for this feature. The output of the event detection is an event database including an event table and a set of event metadata files (as shown in Fig. 2). Each event item listed in the event database is linked to an event metadata file.

4.4 Event modeling

With an event database saving all evolution processes of features, it is tempting to query the database in multiple ways and specify a criterion by which keyframes (or key timesteps) can be chosen to represent the stages of an event or the motion of a feature in space or the different occurrences of events. In this work, we refer to this process as event modeling. Seven database operations are provided in the framework to serve the purpose of event modeling.

Search by conditions The searching operation involves scanning all of the events in the database and finding out those events that match the defined conditions. A condition is usually a logic expression or an inequation. Users can define four types of conditions in searching an event: (1) temporal relationships of features (e.g., a feature expands in its volume for at least five timesteps); (2) a combination of instances (e.g., merge + five continuations + split); (3) feature attributes (e.g., the volumes and heights of features); and (4) event attributes (e.g., the duration of events).

Filter by conditions Filter is the opposite operation of search and involves filtering the events that match the defined conditions. Similarly, four types of conditions can be defined to perform this operation.

Redefinition of states This operation combines several existing states into one new state or classifies an existing state into different states based on the attributes of the feature such as size, volume, and physical properties. After the operation, the old state label(s) will be replaced by the new one(s). For example, the "continued" hurricane feature can be classified into different "categories of hurricane" based on the average wind speed of the hurricane.

Grouping by states This operation first scans all events in the event database and categorizes the features at every timestep of the events into different groups based on the state labels of these features. For each state, a file is created to save the information of the contained features (e.g., index, timestep, attributes).

Terminating an event This operation ends an event in the middle when a predefined condition is matched. For instance, a user can terminate an event when the size or volume of the involved feature is below some threshold.

Segmenting an event This operation segments an event into a set of new events by iteratively truncating the event whenever a predefined condition (e.g., the number of timesteps or some action occurs on the feature) is matched. The timesteps of a truncated part are saved as a new event.

Save current database This operation saves users' operations and current status of the event database. The output is a new database that contains a subset of the detected events.

The database operations introduced above can be incorporated to a Petri Net graph to provide a graphical way of event modeling. A Petri Net graph (Fig. 3a) consists of tokens (solid dots), places (circles), transitions (rectangles), and directed arcs (arrows): Tokens are iconic representations of features undergoing the modeled event; places are containers of tokens and "holds" features of certain states (note: features of different states can stay in one place); transitions sit between places and contain one or a combination of database operations; directed arcs define the connections between places and transitions and thus indicate the directions of tokens. The location of a token indicates the current status of a model event, and an instance of the event ends whenever a token reaches the final place. In Fig. 3a, a Petri Net graph illustrates a model for searching the events that contain the pattern "merge and split" from the event database of the simulation of turbulent vortex. Three transitions (T1, T2, and T3) are designed to run database operations: T1 searches the "merged" features (or tokens) in the place P1 and moves them to the place P2; T2 filters out the "split" features in P2 and moves any other features back to P1; T3 searches the "split" features in the P2 and terminates an event by moving a "split" feature to P3. The tokens in the Petri Net graph reflect the status of the modeled event at the 66th timestep of the data (Fig. 3b).

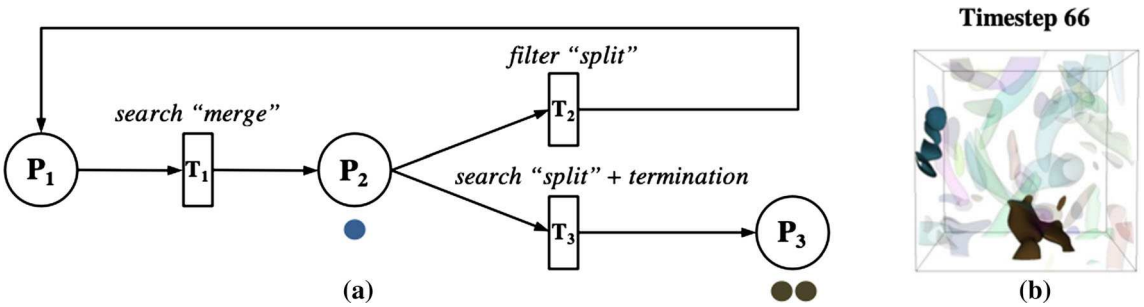

Fig. 3 **a** A Petri Net graph models the "merge-and-split" event. The graph consists of places (circles), transitions (rectangles), directed arcs (arrows), and tokens (solid dots with unique color below places). The tokens in **a** represent the features of the 66th timestep of the simulation of turbulent vortex in **b**

5 Event visualization

A straightforward way of visualizing an event is to view each timestep in a succession of frames, preferably with the involved features highlighted. (An example is shown in Fig. 1.) As the numbers of timesteps and features increase, this method becomes inefficient. Since the event database provides explicit access to all of the events and the involved features, we query the database in multiple ways which leads to three event visualization algorithms: (1) *event temporal visualization* adopts a temporal-centric design strategy which juxtaposes the features that are involved in an event in a consecutive sequence; (2) *event spatial visualization* adopts a spatial-centric design strategy which places events and auxiliary glyphs (e.g., arrow, line, geographic map) in a spatial context to highlight the locations and motions of events and the interactions between the features and events; (3) *event summary visualization* which presents an average view of a bunch of events in a certain state and gives the general pattern of all events of a dataset in a concise summary view. Table 1 lists all of these visualizations and the accompanying example figures.

5.1 Event temporal visualization

An *event temporal visualization* focuses on a single event timestep by timestep and intends to provide a visual answer to queries to the event database about when an event happens, how long the event lasts, and how the involved features look like in the key timesteps of the event. This visualization reads the event metadata of a chosen event through the event table and presents this event as a consecutive sequence of features (isolated from the feature metadata) in a temporal axis. For an event that has long duration, visualizing features in each timestep is no longer efficient if not overwhelming. Instead, scientists prefer to see a bunch of key or representative timesteps that briefly summarize the observed long-lasting event. In this case, the first and last timesteps and those timesteps which change the stage in the event (e.g., a feature splits into several features) are selected as key timesteps. Compared with visual inspection of changes and manual selection of key timesteps, this method achieves similar result automatically and efficiently.

5.2 Event spatial visualization

An *event spatial visualization* intends to provide a visual answer to queries to the event database about where does an event occur and how the involved features evolve and interact in the evolution process. The visualization reads the event metadata of one or a couple of chosen events and places the features of the chosen events in a spatial space to give a principal attention to the spatial attributes of the visualized events. For each visualized event, a collision detection algorithm based on bounding box (Palmer and Grimsdale 1995) is applied to avoid conflictions between the visualized features. In the visualization, the temporal information in an event is depicted using spatial bandwidth of the display and is overlaid on top of the spatial dimensions. By placing multiple events in a spatial context, the visualization presents some overall patterns of the presented events and helps users observe the interactivities among these events. Additional visual elements (e.g., 3D map, coordinate axis, speed line, arrow) are used to highlight the orientations, positions, sizes, and motions of events in the visualization and strengthens the awareness of motion from the visualizations.

5.3 Event summary visualization

An *event summary visualization* is aimed to create a set of averages or targeted summaries of events so that scientists can ask the following questions: For a particular stage of an event, what does it typically look like? May I see an average view of all events in a dataset? Using the database operation "grouping by state" operation (see Sect. 4.4) to gather all of the features that pass through a particular state (e.g., expanding, merged, spinning clockwise) in the event database, we can actually compute an "average" view of the data

Table 1 Event visualization and the example figures

Visualization name	Description	Example figure
Event temporal visualization	Juxtapose every timestep of an event in a sequence	Figures 4c, 5a, 8b
Event spatial visualization	Present one or multiple events in a spatial context (i.e., 3D map)	Figures 5b, 6, 8a, 10
Event summary visualization	Summarize all events in a data by the states of event	Figures 7b, c, 9

at a certain state and even go a step further to ask for a summary picture of the entire database of events. The *event summary visualization* is built around the concept of occupancy map and is suitable for data with one or a couple of features. In this work, two types of *event summary visualizations* can be created by using different occupancy maps.

In the Type 1 *event summary visualization*, an occupancy map is computed for each event state within a single event or across the whole event database, creating a sequence of images that illustrate the typical nature of the data. The visualization counts the occupation frequency of features over all of the timesteps at every voxel of the data frame and uses these computed frequencies as voxel values. The voxel value (i.e., occupation frequency) at the coordinate (i, j, k) can be calculated as follows:

$$O(i,j,k) = \sum_{t=1}^{t=N} V(t)_{i,j,k}$$

where $O(i,j,k)$ is the occupancy value at voxel location (i, j, k), t is a random timestep, and N is the number of timesteps in the dataset. $V(t)$ equals to 1 if the plume object occupies the voxel (i, j, k) at the timestep t or 0 otherwise. Computing the typical shape of a moving feature (whether during an event or for a given state) involves instead registering the feature location by its centroid before computing the desired occupancy map.

On the other hand, a Type 2 *event summary visualization* is intended to give an overview of the entire event dataset or the whole dataset with a single figure. It highlights different stages of a modeled event and their spatial relationships by assigning voxel values with indices of stages. The voxel value at the coordinate (i, j, k) is equivalent to the index of a stage that is dominant at this voxel over all timesteps:

$$O(i,j,k) = p(1 \le p \le M), \quad if \sum_{t=1}^{t=N} S_p(t)_{i,j,k} = Max\left\{\sum_{t=1}^{t=N} S_1(t)_{i,j,k} \sum_{t=1}^{t=N} S_2(t)_{i,j,k} \ldots \sum_{t=1}^{t=N} S_M(t)_{i,j,k}\right\}$$

where p is the index of a random stage in the modeled event, M is the number of stages, and $S_p(t)$ is a binary value and equals to 1 if a feature falls into the stage p and the voxel (i, j, k) belongs to the feature. The voxel value is within the range of 1 to M.

5.4 GUI

All of the proposed event visualizations are incorporated with a graphic user interface (GUI) using Qt with VTK and written with standard C++ language. The GUI takes an event database as input and supports graphically modeling an event with a Petri Net graph and querying the loaded event database for event visualizations. The implementation was designed to be interactive so that scientists can refine their Petri Net models as their hypothesis about the data changes in response to the information conveyed in the visualizations. As shown in Fig. 4a, the upper part of the GUI shows the Petri Net model of the "merge-and-split" event discussed in Sect. 4.4. One of the timesteps (the figure shows the 33rd timestep) is chosen and shown on the lower part of the GUI. Two events are selected from the event table (Fig. 4b) and are presented as *event temporal visualizations* in Fig. 4c.

6 Case studies

In this section, we apply the proposed visualization framework to three representative time-varying scientific datasets as listed in Table 2. Each dataset is different in its number of features, number of events, whether more than one event happens simultaneously, and so forth. Thus, not every visualization is suited to every dataset and vice versa; but as a group, these case studies are used to illustrate the usability and effectiveness of the proposed event visualizations. In each case study in this section, we also discuss the scientific findings that arose from the visualization results. For the first two case studies, we also discuss the comments and feedbacks from the domain experts who cooperated with us.

6.1 Ocean simulation

This dataset is a numerical simulation of the northwest Atlantic Ocean off the coast of North America (Kang and Curchitser 2013) generated by the Regional Ocean Modeling System (ROMS). The simulation domain

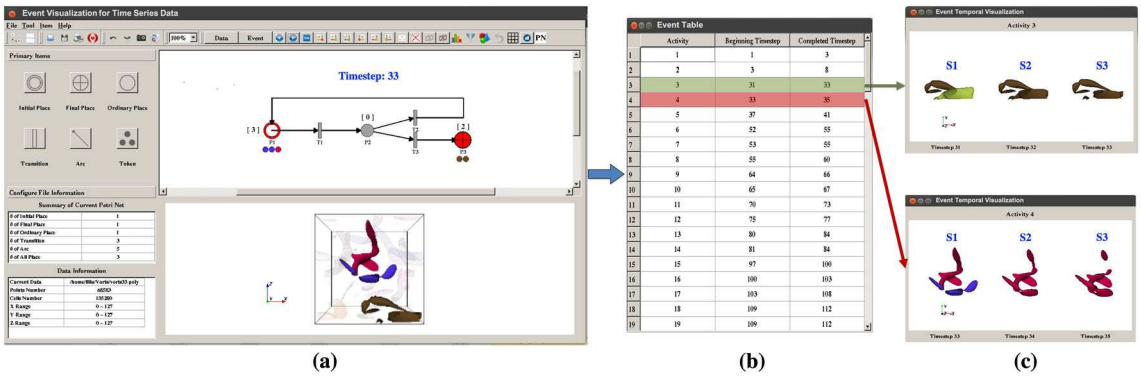

Fig. 4 The event modeling and event visualizations are implemented in a GUI. **a** The upper part of the GUI shows the Petri Net model of the "merge-and-split" event. On the lower part of the GUI, a timestep (the 33rd) in the data is displayed with the "active" features highlighted in solid colors, **b** an event table lists all of the instances of the modeled event, and **c** two events are isolated from the event table and are visualized as event temporal visualizations

Table 2 Application datasets and data information

Datasets	Variables	Dimensions	Duration	Events-of-interest
Ocean simulation	*Okubo–Weiss*, speed, salinity temperature	761 × 361 × 40	1095	Eddy path which is defined as the migration process of an eddy in its life cycle
Underwater plume	*Backscatter intensity*	161 × 121 × 181	479	The general pattern of plume bending and the process of slosh back and forth
Hurricane Isabel	*Precipitation*, speed, temperature, pressure	500 × 500 × 100	48	The tropical cyclone intensifies and weakens in its life cycle

The variables in italics are used for segmenting features from data frames (i.e., feature extraction)

covers most of the path of the Gulf Stream in the northwest Atlantic Ocean (258.34° to 315.19° in longitude, 8.78° to 53.83° in latitude, and 0 to 5500 m in altitude). The implementation in this study covers the most recent 2 years (2006–2007) in the simulation with 730 daily NetCDF files. Each data frame of this dataset is a structured grid with a horizontal spacing of 7 km (722 × 362 grid points) and 40 vertical terrain-following levels stretched toward the surface. The vertical coordinate and stretching functions follow the ROMS conventions (Kang and Curchitser 2013).

In this case study, the event-of-interests is the migration processes of ocean eddies over their life cycles which are referred to as eddy migration paths. The ocean eddies were identified by using the canonical methods: the Okubo–Weiss (OW) parameter (Okubo 1970; Weiss 1991), and were segmented from every data frame of the ROMS data as features. Each voxel in the eddy features contains three variables: horizontal speed, salinity, and temperature. In an eddy migration path, an eddy may go through five states from its birth to death: birth, propagate, merge, split, and death. After performing the feature tracking and event detection, all eddy migration paths were detected from the dataset and are saved in an event database. A "filter by condition" database operation was applied to the event database to keep only the eddy migration paths that last at least 50 days. In Fig. 5a, an eddy migration path is presented in the style of *event temporal visualization* in which the first and the last timesteps and those state-shifting timesteps in the events were automatic selected as key timesteps. The visualized timesteps were color-coded by temperature and were represented as wireframes to highlight the variations of topological structures (Liu et al. 2019). In Fig. 5b, the same eddy migration path is presented as *event spatial visualizations* with speed lines depicting the motions of the eddies. In Fig. 6 (Liu et al. 2017), all eddy migration paths in the event database that last at least 50 days are placed in a 3D map to give an overview of the motion and distribution of long-lived ocean eddies (i.e., eddies that last at least 50 days). Timesteps of each visualized event were automatically chosen to avoid conflictions. The 3D map was generated by combining ocean basin bathymetry with land surface topography for the east coast of the USA and creating a terrain elevation surface using Delaunay Triangulation (Ruppert 1995). The terrain surface generation uses a high-resolution elevation dataset from National Oceanic and Atmospheric Administration (NOAA, http://www.noaa.gov/).

Three domain experts at Rutgers University (one from the Department of Environmental Sciences, and two from the Department of Marine and Coastal Sciences) were invited to evaluate the visualizations in this

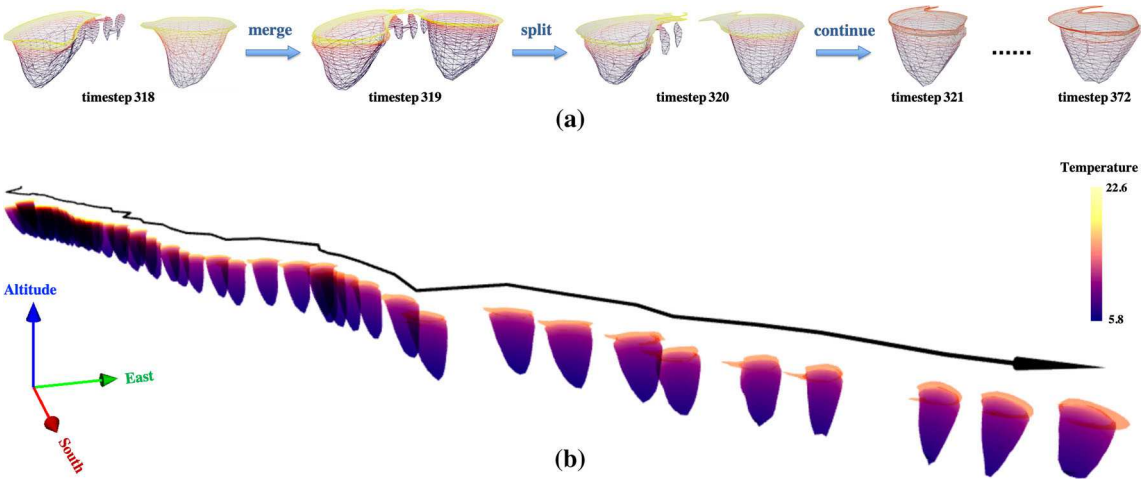

Fig. 5 **a** Key timesteps of a chosen eddy migration path are visualized as an event temporal visualization. Each timestep is color-coded by the temperature variation and is plotted as a wireframe to highlight the topological structures of eddy features, **b** the same eddy migration path is presented as an event spatial visualization. A directional line on the top of the visualization is used to give a sense of motion

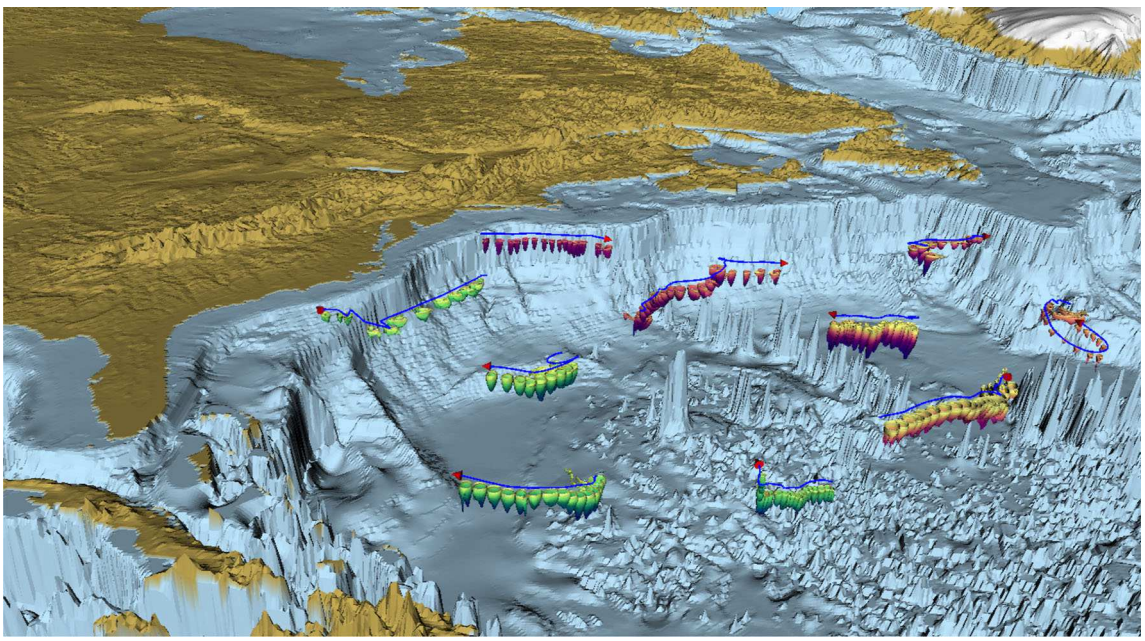

Fig. 6 This event spatial visualization showcases the eddy migration paths that last at least 50 days in 2006 and 2007 in a 3D map of North Atlantic Ocean and American Continent. Directional lines above eddy migration paths highlight the motion. Two colormaps are used to differentiate eddies by their spin directions and convey variations in ocean temperature

case study. The experts appreciate using an eddy migration path to represent the evolution process of an eddy (i.e., Fig. 5), as one expert commented "it illustrates trajectories of the long-lived cyclonic and anticyclonic eddies in 3D. Visualizing an eddy migration path as a series of 3D entities provides more details than traditional 2D vertical profile view and tracking lines. The color-coded isosurface clearly conveys the temperature variation on eddy surface." The experts were amazed to see that the eddies almost invariably tapered to a point at depth: The tendency to narrow with the depth has implications about the geostrophic and shear conditions of the ocean and how they change with depth. They also thought placing the *event spatial visualization* of eddy migration paths in the context of a geographic map (Fig. 6) helped them understand the different life histories of anti-cyclonic and cyclonic eddies and how these are affected by interactions with the ocean basin: "provide insights into the eddies' on-shore and cross-stream transport

of the momentum and energy and both figures tell a story of what happens to Gulf Stream eddies." The experts were surprised at how few eddies actually approach the shelf and that most anti-cyclonic eddies move away from the shelf.

6.2 Underwater plume

This dataset consists of 479 acoustic scans of an underwater hydrothermal plume that is located above a cluster of black smokers on Grotto mound, Main Endeavour Field, Juan de Fuca Ridge (Bemis et al. 2015). Each data frame (dimensions $161 \times 121 \times 181$) maps backscatter intensity, which increases from the metallic sulfides and temperature fluctuations within the plume, over a 3D volume. Theoretically, tidal currents push a plume in directions that vary in an elongated ellipse such that the primary motion appears as sloshing back and forth along an elliptical path. At current stage, only the two main plumes are observed for study due to sidelobe interference by the Grotto edifice. In studying the behaviors and characteristics of hydrothermal plumes which have been the subject of much scientific interest, scientists always want to ask: "What is the general pattern of plume bending? How do the plumes look like on the seafloor? How the two plumes vary in their bending orientations, sizes, and shapes and interact with each other in time?" These questions are difficult to answer with traditional two-dimensional plots (e.g., scatter plots, line plots and bar charts) of physical properties, tables, and photographs that are often used in oceanographic publications. The goal of this case study is to develop scientific insight into these questions by using the proposed event visualizations.

After performing the feature-based event computation, the sloshing processes of the two plumes were detected from the data and were saved in an event database. Both of the plumes never disappeared and went through two states in the motion: birth and continuation. To reflect the different bending states of a plume, we applied a "redefinition of the states" database operation and classified the bending motion of a plume into five bending states (Fig. 7a): S1: extreme bending to the north; S2: mild bending to the north; S3: little to no bending; S4: mild bending to the south; and S5: extreme bending to the north. The general patterns of the plume bending were summarized through grouping the large plume at different timesteps by the defined bending states and creating two types of *event summary visualizations* (Fig. 7). In Fig. 7b, five occupancy maps of the large plume are rendered with the frequency of plume occurrence in each voxel color-coded by a rainbow colormap in which the color varies from cold to hot as the frequency increases. As can be found from the figure, the "averaged plumes" reveal a clear asymmetry of the plume bending response and thus the driving force of tidal currents: The plume bends much more strongly to the north (S1 and S2) than to the south (S4 and S5). Furthermore, although the S5 appears similar to S4, the slight shortness of S5 indicates that much of the bending in S5 is out of the typical plane defined by tidal currents. We also find that the plume gets thicker (or has a bigger radius) as it bends to the two sides and it remains the thinnest in the vertical state (S3). In Fig. 7c, a Type 2 *event summary visualization* of the large plume is presented using five unique colors to indicate the occupancies of the five bending states. The front view of the visualization shows that the areal extent across the vertical plane of bending derives from the systematic bending response to ocean currents. From the side view and top view, we can find that the bending motion of the plume is not in an ideal plane but closer to an ellipse as the plume model in the laboratory predicts. At the extremes of bending in the state S1 (pink) and the state S5 (dark blue), the plume leans out of the plane of bending defined by the inner stages (states S2–S4). This out-of-plane bending could be induced by ocean currents and reflects the complexity of the ocean currents. The incidences of plume at each bending state, which is reported in the bar chart on the left of Fig. 7c, also show the asymmetry of plume bending.

To study the temporal and spatial properties of the plume bending and the interaction between the two plumes, we segmented the two events by days (i.e., every 24 timesteps) and created a series of "daily plume bending processes." In Fig. 9a, the daily bending processes of both of the large and small plumes on October 9, 2010, are presented as an *event spatial visualization* and are placed in the context of the seafloor (bathymetry), which showcases the scenario where the two plumes slosh around the Grotto Vents and gives a reference of the sizes and positions of plumes. Careful color-coding with visual effectiveness enhancement makes the scenario more real and explainable. As can be found from the figure, the large plume formed similar to a pillar and located in the west, while the small plume had a bifurcate shape and located in the east. In Fig. 9b, the two "daily plume bending processes" are presented as an *event temporal visualization* and are put side by side. In each timestep, a rectangular plate and the centerlines of plumes (a concise representation of the bending orientation) were added to enhance the visual effectiveness: The plate highlighted the visual connections between the plumes; the centerlines in the center of each plate facilitated

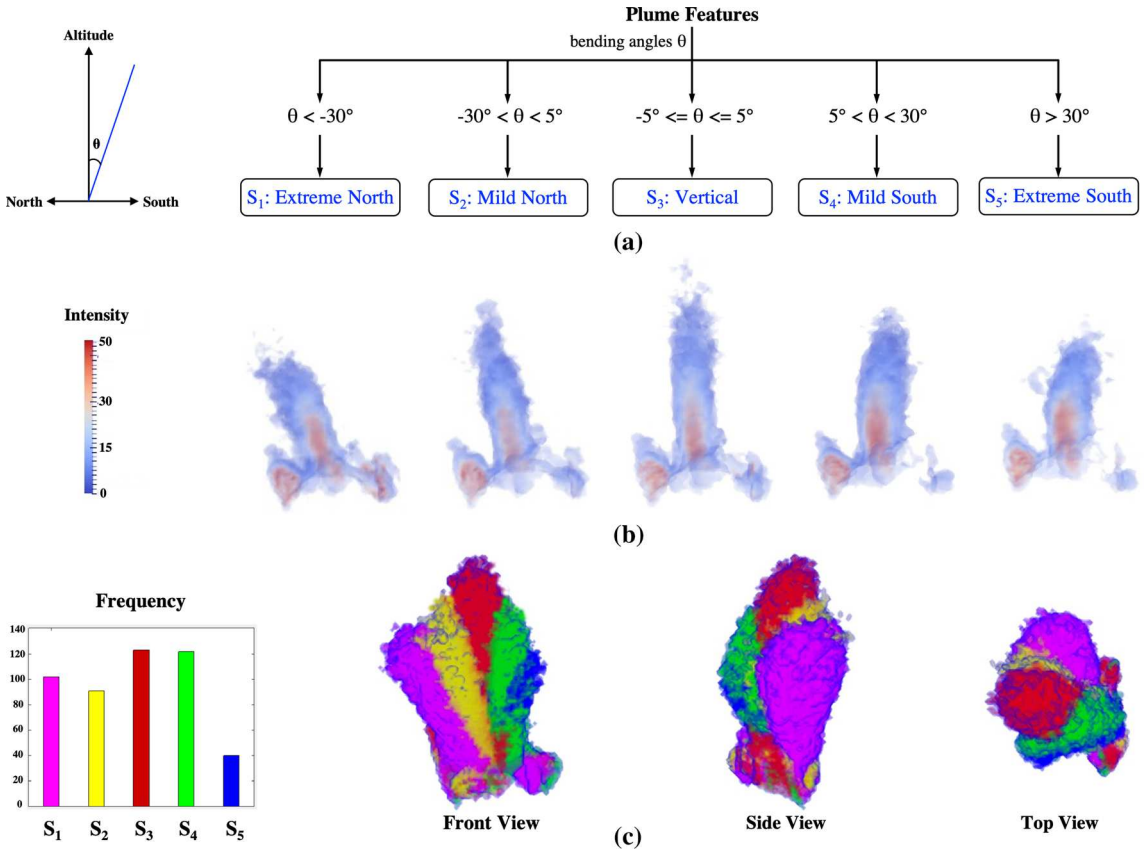

Fig. 7 **a** An event model classifies the bending motion of a plume into five bending states based on the bending angles, **b** a type 1 event summary visualization summarizes the five bending states of the large plume: five occupancy maps of the large plume are volume rendered with the frequency of plume occurrence in each voxel color-coded by a rainbow colormap, **c** a type 2 event summary visualization gives a general summary of the bending motion of the large plume across the whole dataset by using five unique colors to indicate the occupancies of the five bending states

a comparison of the plume motions; the projections of the centerlines to the plates remove the possible visual ambiguity. The visualization suggests a strong semi-diurnal periodicity (2 cycles/day) for the bending motions of both the large plume (cycle 1: 1–8 timesteps; cycle 2: 9–24 timesteps) and the small plume (cycle 1: 1–10 timesteps; cycle 2: 11–24 timesteps), which is consistent with the hypothesis about the periodicity of plume motion and was confirmed by many spectral analyses (Bemis et al. 2015). Besides the bending periodicity, we also find that the small plume always tends to bend toward the large plume. This bending tendency suggests the interaction between the two plumes can probably be simplified as the entrainment field of the large plume.

Two experts on the observation of underwater plumes (one from Woods Hole Oceanographic Institution and the other from University of Washington at Seattle) provided feedback on the use of the event visualizations in this case study. One expert commented the *event summary visualization* (Fig. 7) "provides an effective and succinct way of summarizing the motion of the plume within a long sequence" and the *event temporal visualization* (Fig. 8b) "by aligning two sequences of plume on a time axis, the interactions between the plumes and the occurrence of the bending anomalies stands out." The other expert commented "event sequence is more evocative of the actual bending than the typical feather plot of oceanographic illustrations." Both experts agreed that the *event spatial visualization* in Fig. 8a "is good to mark geographic directions of plume bending instead of just reporting left and right." As a general comment, they agreed that the event visualizations of the underwater plume are consistent with a long-time observation of the actual data and are effective in hypothesis testing.

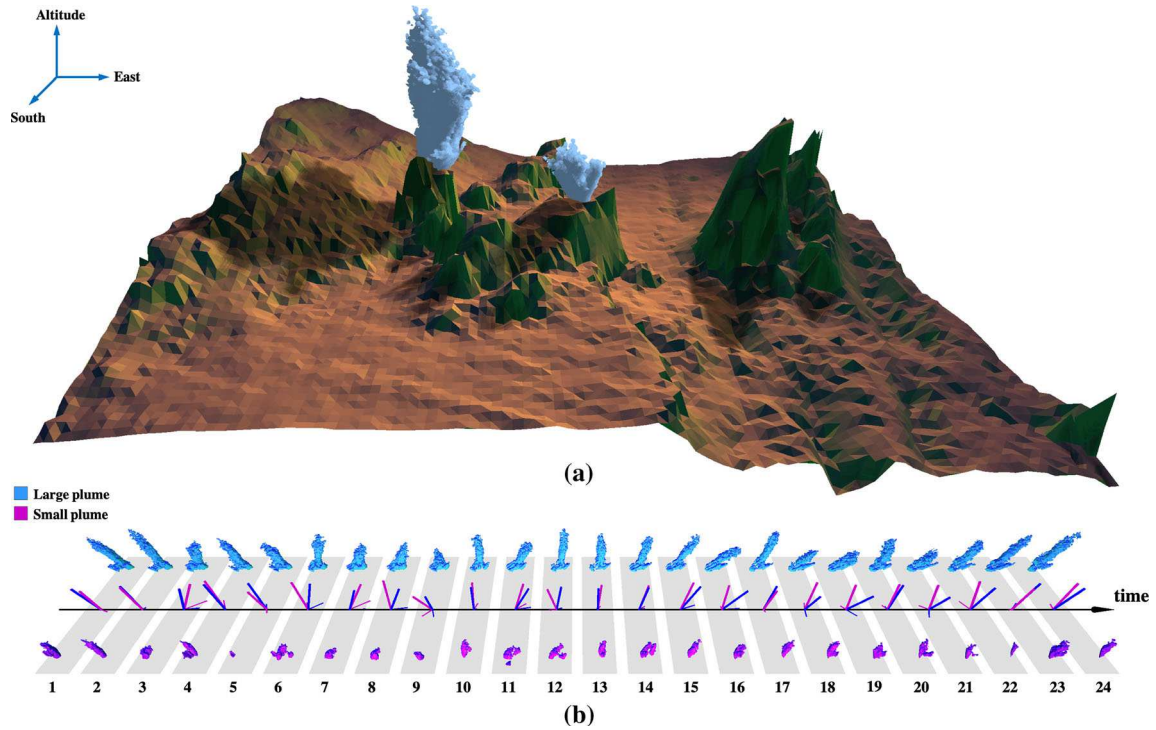

Fig. 8 **a** The bending processes of both of the large and small plumes on October 9, 2010, are presented as an event spatial visualization and are placed in the context of the seafloor (bathymetry), which gives a reference of the sizes and positions of plumes, **b** the two "daily plume bending processes" are presented as an event temporal visualization and are put side by side along a timeline for comparison and temporal analysis

6.3 Hurricane Isabel

This is a simulation of Hurricane Isabel from IEEE Visualization 2004 contest (http://vis.computer.org/vis2004contest/). The data have 48 timesteps with dimensions of 500 × 500 × 100. It consists of several time-varying scalar and vector variables (e.g., temperature, pressure) over large dynamic ranges. In studying the evolution of the Hurricane Isabel, scientists always need to track the intensification and weakening process of the hurricane and display the results in ways that are readily comparable to the observational data on hurricanes. The goal of this case study is to help scientists easily see whether the simulation reproduces the known behavior of hurricanes as well as to explore the details that cannot be easily observed in nature.

In this case study, the dataset contains only one feature (i.e., the Hurricane Isabel) at each timestep. Three variables were chosen for study and were saved at each voxel of the hurricane feature: horizontal speed, salinity, and temperature. Since the Hurricane never disappeared in the simulation, the detected event (i.e., the evolution process of the hurricane) only contains two states: birth and continuation. To analyze how wind speed affects the shapes and sizes of a hurricane in the evolution process, we applied a "redefinition of the states" database operation and classified the evolution process of the Hurricane Isabel into five states (as shown in Fig. 9a) based on the Saffir–Simpson Hurricane Wind Scale (Iacovelli 1991): S1 having the lowest wind speed (32–42 m/s) and S5 having the highest wind speed (over 70 m/s). In Fig. 9b, an *event summary visualization* is computed for each defined state of which the averaged hurricane (i.e., the occupancy map) was color-coded by the wind speed at every voxel. As can be clearly seen from the figure, the hurricane speed rose with the increasing warm colors and suggested a corresponding size increase from a Category-2 Hurricane (S2) to a Category-5 Hurricane (S5). In Fig. 10, an *event spatial visualization* is created to illustrate the intensification and weakening process of the hurricane. In the visualization, seven timesteps were automatically chosen from all of the 48 timesteps to avoid conflictions between neighboring timesteps and were placed in the same 3D map of the case study one (the American Continent and the North Atlantic). It can be learned from the figure that the Hurricane Isabel first intensified from a Category-2 tropic cyclone to a Category-5 full-fledged hurricane then gradually weakened to a Category-3 hurricane after landing the East Coast of USA.

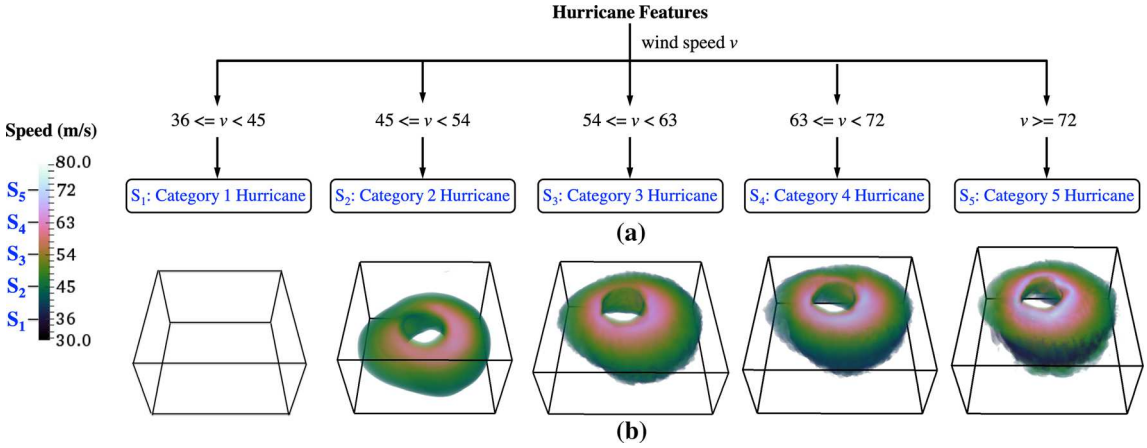

Fig. 9 **a** An event model classifies the hurricane motion into five states (S1–S5) based on the wind speed, **b** a type 1 event summary visualization summarizes the different states the Hurricane Isabel went through in the motion process. The increase in wind speed from S2 to S5 suggests a corresponding size increase in the hurricane

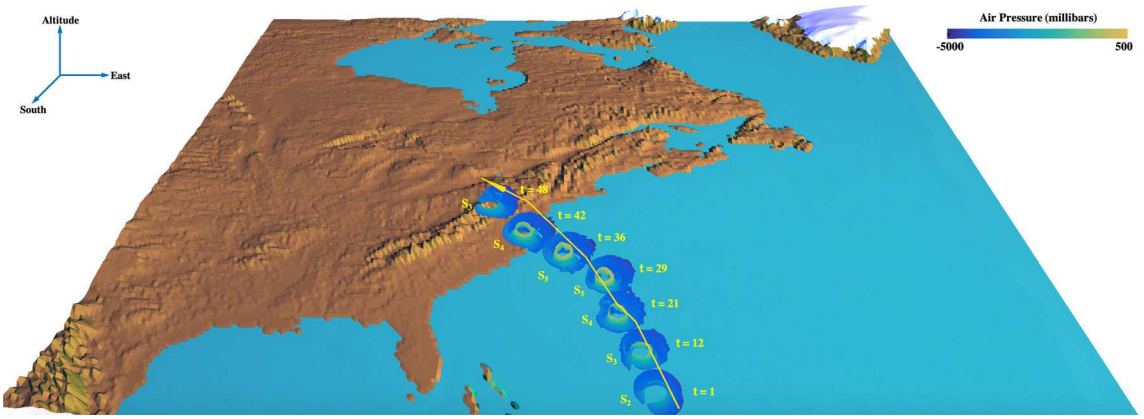

Fig. 10 The motion of Hurricane Isabel is illustrated by an event spatial visualization and is placed in the context of North Atlantic Ocean and North American Continent. The hurricane first intensifies from a level 2 tropic cyclone to a level 5 full-fledged hurricane then weakens to a level 3 hurricane in the last timestep. The hurricane feature at each timestep is color-coded by the variation of air pressure

7 Discussion and conclusion

Visualizing events from time-varying scientific datasets forms an important part of the analysis process and is crucial to understand the underlying scientific phenomena in the data. This study was prompted by the need for better tools to analyze, summarize, and illustrate events in the data. In most papers, the entire time series can never be displayed, but scientists will show anywhere from 4–8 timesteps selected from a sequence of thousands to hundreds of thousands of timesteps. These key timesteps are chosen either semi-randomly or manually. As the number of features and timesteps increases, it becomes impossible for scientists to view every single timestep to choose those of interest. Videos are only a partial solution (being still long and images fleeting). In this work, we propose a framework for the visualization of events in time-varying scientific datasets. Our method allows scientists to perform data reduction, and more importantly, enables hypothesis testing—determining whether the theoretical event matches the actual event. Using the feature metadata and the event database generated from the feature-based event computation, three event visualizations can be created to permit automated identification, highlight, and summarize features and events in time-varying scientific data. Each of these event visualizations may illustrate just one aspect of a dataset or the entire dataset at once, but the connection of the visualizations to an underlying conceptual model of the dataset permits ready refinement of the visualizations by updating the analysis process. The applications to the datasets Ocean Simulation, Hurricane Isabel, and Underwater Plume time series have

demonstrated that the proposed event visualizations can automatically and effectively illustrate events, their typical structure and occurrence, and feature motion in a concise, cogent, and meaningful way. Collaboration with domain experts has proven essential to determining effective visualizations that focus on questions of active scientific interest.

Despite these very successful examples, there still are a few limitations of our work at this point. The event modeling requires considerable understanding of the event detection and the event database operations, which results in much interaction between the domain scientist and the visualization expert. While the graphical event modeling with a Petri Net graph helped and conceptually domain scientists were able to create cartoon-like state graphs, creating a working Petri Net graph was a little more complicated. A few more extensions for this work can be developed in the future. First, a more formal usability study and systematic evaluation of the proposed visualizations by domain experts or other users will be conducted to improve our methods. Second, alternative methods to compute and illustrate a typical event with the incorporation of more statistical information and machine learning tools may replace current occupancy-based *event summary visualization*. Extending the event detection algorithm to handle the parallel action occurrence (e.g., a feature expands while propagating at the same timestep) could be another direction for our future work. Finally, the event detection could be combined with data mining techniques to enable a scientist to develop and test numerous different hypotheses even when there is no pre-existing conceptual model on which to base the event detection and modeling.

Acknowledgements We thank the Department of Environmental Sciences and the Department of Marine and Coastal Sciences at Rutgers University for providing the ocean simulation data and the underwater plume data. We also thank the domain experts who gave us feedback: Guangyu Xu, Darrell Jackson, Russ Light, Dujuan Kang, and Enrique Curchitser. This work has been funded by the US National Science Foundation (NSF) Grants 0825088 and OCE-1049088.

References

Aggarwal JK, Ryoo MS (2011) Human activity analysis: a review. ACM Comput Surv (CSUR) 43(3):16

Albanese M, Chellappa R, Moscato V, Picariello A, Subrahmanian VS, Turaga P, Udrea O (2008) A constrained probabilistic petri net framework for human activity detection in video. IEEE Trans Multimed 10(8):1429–1443

Bemis KG, Silver D, Xu G, Light R, Jackson D, Jones C, Ozer S, Liu L (2015) The path to COVIS: a review of acoustic imaging of hydrothermal flow regimes. Deep Sea Res Part II Top Stud Oceanogr 121:159–176

Born S, Wiebel A, Friedrich J, Scheuermann G, Bartz D (2010) Illustrative stream surfaces. IEEE Trans Vis Comput Graph 16(6):1329–1338

Brambilla A, Carnecky R, Peikert R, Viola I, Hauser H (2012) Illustrative flow visualization: state of the art, trends and challenges. Vis Orient Vis Design Flow Illus. https://doi.org/10.2312/conf/EG2012/stars/075-094

Caban J, Joshi A, Rheingans P (2007) Texture-based feature tracking for effective time-varying data visualization. IEEE Trans Vis Comput Graph 13(6):1472–1479

Clyne J, Mininni P, Norton A (2013) Physically-based feature tracking for CFD data. IEEE Trans Vis Comput Graph 19(6):1020–1033

Ellsworth D, Chiang LJ, Shen HW (2000) Accelerating time-varying hardware volume rendering using tsp trees and color-based error metrics. In: 2000 IEEE symposium on volume visualization (VV 2000). IEEE, pp 119–128

Hauser H (2006) Generalizing focus+ context visualization. In: Scientific visualization: the visual extraction of knowledge from data. Springer, Berlin, Heidelberg, pp 305–327

Hsu WH, Mei J, Correa CD, Ma KL (2009) Depicting time evolving flow with illustrative visualization techniques. In: International conference on arts and technology. Springer, Berlin, pp 136–147

Iacovelli D (1991) The Saffir/Simpson hurricane scale: an interview with Dr. Robert Simpson

Jankun-Kelly TJ, Ma KL (2001) A study of transfer function generation for time-varying volume data. In: Volume graphics 2001. Springer, Vienna, pp 51–65

Ji G, Shen HW (2006) Feature tracking using earth mover's distance and global optimization. In: Pacific graphics, vol 2

Ji G, Shen HW, Wenger R (2003) Volume tracking using higher dimensional isosurfacing. In: Proceedings of the 14th IEEE visualization 2003 (VIS'03). IEEE Computer Society, p 28

Joshi A, Rheingans P (2005) Illustration-inspired techniques for visualizing time-varying data. In: VIS 05. IEEE visualization, 2005. IEEE, pp 679–686

Joshi A, Rheingans P (2008) Evaluation of illustration-inspired techniques for time-varying data visualization. Comput Graph Forum 27(3):999–1006

Joshi A, Caban J, Rheingans P, Sparling L (2009) Case study on visualizing hurricanes using illustration-inspired techniques. IEEE Trans Vis Comput Graph 15(5):709–718

Kang D, Curchitser EN (2013) Gulf Stream eddy characteristics in a high-resolution ocean model. J Geophys Res Oceans 118(9):4474–4487

Lavee G, Rivlin E, Rudzsky M (2009) Understanding video events: a survey of methods for automatic interpretation of semantic occurrences in video. IEEE Trans Syst Man Cybern Part C (Appl Rev) 39(5):489–504

Lee TY, Shen HW (2009) Visualizing time-varying features with tac-based distance fields. In: 2009 IEEE pacific visualization symposium. IEEE, pp 1–8

Liu L, Silver D, Bemis K, Kang D, Curchitser E (2017) Illustrative visualization of mesoscale ocean eddies. Comput Graph Forum 36(3):447–458
Liu L, Silver D, Bemis K (2019) Visualizing three-dimensional ocean eddies in web browsers. In: IEEE access, pp 44734–44747
Lorensen WE, Cline HE (1987) Marching cubes: a high resolution 3D surface construction algorithm. In: ACM siggraph computer graphics, vol 21, no 4. ACM, pp 163–169
Lu A, Shen HW (2008) Interactive storyboard for overall time-varying data visualization. In: 2008 IEEE pacific visualization symposium. IEEE, pp 143–150
Lum EB, Ma KL, Clyne J (2002) A hardware-assisted scalable solution for interactive volume rendering of time-varying data. IEEE Trans Vis Comput Graph 8(3):286–301
Ma KL (2003) Visualizing time-varying volume data. Comput Sci Eng 5(2):34
Ma KL, Shen HW (2000) Compression and accelerated rendering of time-varying volume data. In: Proceedings of the 2000 international computer symposium-workshop on computer graphics and virtual reality, pp 82–89
Ma KL, Liao I, Frazier J, Hauser H, Kostis HN (2012) Scientific storytelling using visualization. IEEE Comput Graph Appl 32(1):12–19
Muelder C, Ma KL (2009) Interactive feature extraction and tracking by utilizing region coherency. In: 2009 IEEE pacific visualization symposium. IEEE, pp 17–24
Okubo A (1970) Horizontal dispersion of floatable particles in the vicinity of velocity singularities such as convergences. Deep Sea Res Oceanogr Abstr 17(3):445–454
Ozer S, Silver D, Bemis K, Martin P (2014) Activity detection in scientific visualization. IEEE Trans Vis Comput Graph 20(3):377–390
Palmer IJ, Grimsdale RL (1995) Collision detection for animation using sphere-trees. Comput Graph Forum 14(2):105–116
Pfister H, Lorensen B, Bajaj C, Kindlmann G, Schroeder W, Avila LS, Raghu KM, Machiraju R, Lee J (2001) The transfer function bake-off. IEEE Comput Graph Appl 21(3):16–22
Poppe R (2010) A survey on vision-based human action recognition. Image Vision Comput 28(6):976–990
Rautek P, Bruckner S, Gröller E, Viola I (2008) Illustrative visualization: new technology or useless tautology? ACM SIGGRAPH Comput Graph 42(3):4
Reinders F, Post FH, Spoelder HJ (2001) Visualization of time-dependent data with feature tracking and event detection. Visual Comput 17(1):55–71
Rheingans P, Ebert D (2001) Volume illustration: nonphotorealistic rendering of volume models. IEEE Trans Vis Comput Graph 7(3):253–264
Ruppert J (1995) A Delaunay refinement algorithm for quality 2-dimensional mesh generation. Journal offocuses on a single event timestep by timestep algorithms 18(3):548–585
Samtaney R, Silver D, Zabusky N, Cao J (1994) Visualizing features and tracking their evolution. Computer 27(7):20–27
Shen HW, Chiang LJ, Ma KL (1999) A fast volume rendering algorithm for time-varying fields using a time-space partitioning (TSP) tree. In: Proceedings visualization'99 (Cat. No. 99CB37067). IEEE, pp 371–545
Silver D, Wang X (1997) Tracking and visualizing turbulent 3d features. IEEE Trans Vis Comput Graph 3(2):129–141
Viola I, Gröller ME (2005) Smart visibility in visualization. In: Computational aesthetics, pp 209–216
Viola I, Kanitsar A, Groller ME (2005) Importance-driven feature enhancement in volume visualization. IEEE Trans Vis Comput Graph 11(4):408–418
Weiss J (1991) The dynamics of enstrophy transfer in two-dimensional hydrodynamics. Phys D 48(2–3):273–294
Wohlfart M, Hauser H (2007) Story telling for presentation in volume visualization. In: Proceedings of the 9th Joint Eurographics/IEEE VGTC conference on visualization. Eurographics Association, pp 91–98