Tracking Features in Embedded Surfaces: Understanding Extinction in Turbulent Combustion

Wathsala Widanagamaachchi[1*] Pavol Klacansky[1*] Hemanth Kolla[2†] Ankit Bhagatwala[2†]
Jackie Chen[2†] Valerio Pascucci[1*] Peer-Timo Bremer[1,3‡]

[1]SCI Institute, University of Utah [2]Sandia National Laboratory [3]Lawrence Livermore National Laboratory

Abstract

Understanding the temporal evolution of features of interest requires the ability to: (i) extract features from each snapshot; (ii) correlate them over time; and (iii) understand the resulting *tracking graph*. This paper provides new solutions for the last two challenges in the context of large-scale turbulent combustion simulations. In particular, we present a simple and general algorithm to correlate hierarchical features, embedded in time-dependent surfaces. This, for the first time, provides a parameter independent approach to track embedded features. Furthermore, we provide a new technique to adaptively change feature parameters over time to both: alleviate artifacts due to insufficient temporal resolution as well as to simplify the resulting tracking graphs to promote new scientific insights. Our solutions are integrated into a general and flexible analysis environment that allows users to interactively explore the spatio-temporal behavior of large-scale simulations. We demonstrate the results using the analysis of *extinction holes* in turbulent combustion as primary case study and a number of other applications to illustrate the generality of the approach.

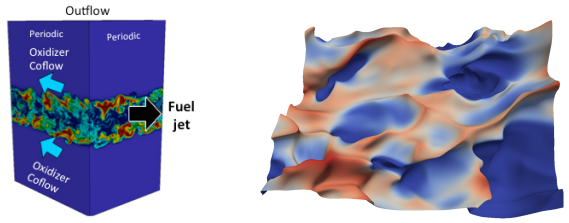
Index Terms: E.1 [DATA STRUCTURES]: Graphs and Networks—; J.2 [PHYSICAL SCIENCES AND ENGINEERING]: Engineering—;

1 Introduction

One of the most common analysis tasks in many scientific applications is the need to understand the evolution of time-varying features. Often, there exists some notion of a feature of interest at each moment in time, e.g. extinction regions in flames, eddies in the ocean, and these features evolve over time. Exploring and analyzing the behavior of these features with respect to changes in parameters and in time is of significant interest. Often, the complex spatio-temporal relationships of these features are represented using *tracking graphs* that capture the evolution of features across time as a collection of *tracks*. In practice, one is usually given a number of time steps from a simulation or an experiment and understanding the feature evolution proceeds in three steps: First, one needs to define the feature of interest within each time step; Second, features should be correlated, i.e. tracked, between successive time steps; and Third, the resulting feature tracks need to be analyzed.

Here, we are interested in understanding the evolution of extinction regions in turbulent flames. More specifically, we are given a large-scale, direct numerical simulation (DNS) of a turbulent jet flame undergoing extinction and re-ignition (see Figure 1). Scientists define the flame as an isosurface of mixture fraction – the ratio of unburnt fuel versus combustion products – as shown in Figure 1(b). However, not all of this surface is considered burning.

*e-mail: {wathsy, klacansky, pascucci}@sci.utah.edu
†e-mail: {hnkolla, abhagat, jhchen}@sandia.gov
‡e-mail: bremer5@llnl.gov



(a) (b)

Figure 1: (a) Schematic of the DNS domain for the turbulent jet flame. Image curtsey of [4]. (b) The jet flame isosurface of mixture fraction with interpolated OH concentration. The 'blue' regions represent extinction regions and 'red' indicates an active chemical process. The isosurface is complex and can fold on itself making feature tracking challenging.

Instead, a second indicator, the OH concentration, is used to classify the flame into burning regions and *extinction holes*. A high OH level indicates an active chemical process while a low concentration marks an extinct portion of the flame. As the simulation proceeds, more extinction holes develop, grow, and ultimately heal again and disappear. Tracking this process over time on a per-hole basis may lead to new insights into why these holes are forming and what effects contribute to their closing.

A number of factors make this use case especially challenging for existing techniques. First, the exact OH threshold is not known and in fact exploring how the evolution differs for different thresholds is of significant interest. However, given the size of the source data (≈1500GB in total), a repeated analysis is practically infeasible especially as part of an interactive exploration. Furthermore, most existing tracking techniques are based on spatial overlap or motion prediction [23, 16, 28]. The former does not apply as the flame surface itself moves quite significantly and flame surfaces of successive time steps rarely overlap. Predictor-based approaches [22, 21, 25] work well for sparse sets of well separated features in which ambiguities can be accurately resolved. Unfortunately, as shown in Figure 1b, the flame surfaces become highly convoluted especially in later time steps, and features are quite densely present throughout the flame. This makes predictor-based schemes unreliable and difficult to tune. Weber et al. [26] propose to use a Reeb graph of a space-time surface to track embedded features. However, as discussed later, this technique is quite involved, as well as, expensive and is only applicable to a fixed OH threshold. Instead, our approach is simple to implement and produces equivalent results for variable thresholds. Finally, the resulting tracking graphs are large, highly complex, and difficult to comprehend (see Figure 2). This is partially due to artifacts from the lack of temporal resolution and unavoidable instabilities in the parameter choices. In principle, the former could be avoided through repeated execution with more temporal resolution and/or in-situ processing. However, in practice re-running even parts of the simulation is infeasible.

In this paper, we address the challenges discussed above by combining a new algorithm to correlate embedded features with a new

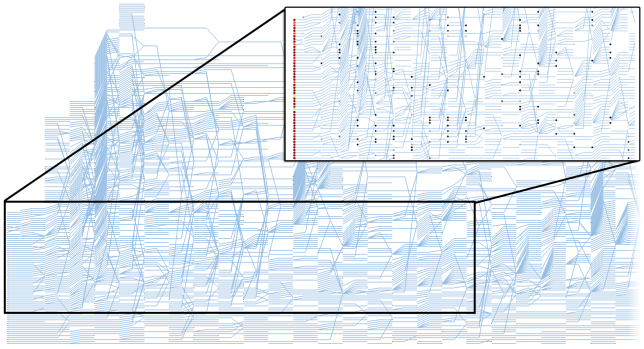


Figure 2: The tracking graph for the first 26 of the 101 time steps in the turbulent jet flame data set. As discussed below we display an intermediate step between each consecutive pair so a total of 51 steps are displayed. The graph contains only 1802 nodes, still it is nearly incomprehensible due to the complex interactions.

adaptive parameter selection to allow an interactive and parameter independent exploration of embedded features in large-scale simulations. In particular, we use merge trees [7, 28] to define extinction holes in a parameter independent manner for each time step. We use the space-time isosurfacing of Bhaniramka et al. [5] and show how merge trees of different subsets of a space-time isovolume can be combined with a simple overlap tracking to correlate and thus track parameter independent families of embedded features. Finally, we exploit the flexibility in defining temporally and spatially varying parameters to simplify the resulting tracking graphs by removing artifacts caused by temporal undersampling and parameter instabilities. Our contributions in detail are:

- A new algorithm to track families of embedded features by computing merge trees of various subsets of space-time isovolumes;
- An approach to compensate for the insufficient temporal resolution by locally adapting the OH parameter in time;
- A progressive, three-pass algorithm to locally adapt thresholds within a given error bound in order to produce more temporally cohesive and thus easier to comprehend tracking graphs; and
- An interactive system to explore the temporal evolution of embedded features with applications to several multi TeraByte combustion simulations.

2 Related Work

This section briefly recaps relevant related work in the areas of feature extraction, correlation, and tracking graphs.

Feature Extraction. There exist a large variety of feature definitions and even a cursory overview is beyond the scope of this paper. Here we are predominantly interested in threshold-based features of a scalar field, i.e. OH. Traditionally, such features are extracted using isosurfaces [20] or interval volumes [13]. However, as mentioned above, there exist no a priori known threshold and repeated processing is too costly. Instead, we need techniques able to extract information for all or a large range of thresholds in a single pass. Recently, topological based techniques such as the Morse-Smale complex [15], contour trees [9], or merge trees [7, 3] have been used very effectively to capture flexible feature hierarchies. Here, we use merge trees as the simplest structure that encodes extinction holes. To remove artifacts and simplify the tracking graphs we use localized thresholds to define features. This idea is similar to the flexible isosurfaces of Carr et al. [10] but based on merge rather than contour trees and thus somewhat easier to compute and manipulate.

Feature Correlation. Given a set of features the next step is to correlate them and analyze their evolution over time. In particular, scientists are interested in events such as the birth/death or the merging/splitting of features as well as in understanding the evolution of a single feature over time, e.g. how its size or chemical composition changes. One simple form of tracking is through spatial overlap: defining features as correlated if they intersect in space [23, 24, 28]. However, in the case of extinction holes, features are defined on a moving surface and two consecutive snapshots of the mixture fraction isosurface may not overlap at all. This makes overlap based tracking not applicable.

Another set of techniques considers isosurfaces of time varying functions and tracks their components by either directly computing the topology of the time-varying function [12] or using various acceleration structures [1, 17]. However, these techniques only consider isosurfaces of a single scalar function not the restriction of one function (OH concentration) on the isosurface of another. This is a crucial limitation for our application as 3D extinction regions are space-filling. As shown in Figure 4, isosurfaces of OH at the relevant thresholds do not form isolated features but a "swiss-cheese" like structure with internal voids of burning material. As a result tracking extinction regions only becomes meaningful once restricted to the mixture fraction surfaces.

Weber at al. [26] track embedded features – in their case fuel consumption restricted to a temperature isosurface – by first extracting a space-time isovolume and then thresholding it to extract a space-time isosurface. Alternatively, this surface can be thought of as the boundaries of the burning regions swept through time. The Reeb graph of time as a function on this surface creates the tracking graph. However, this approach is quite involved and also does not allow the fuel consumption rate (the equivalent of OH in our case) to vary. Instead of large-scale Reeb graphs our approach requires only much simpler to compute merge trees yet provides full flexibility in choosing the OH threshold.

There also exist a number of approaches to track features through flow fields by directly integrating their paths in the flow [14, 27]. However, extinction regions are not purely advected but also "move" as the flame consumes fuel. Thus, their paths are not necessarily well described by the surrounding flow. Furthermore, it is unclear how well the turbulent flow present at the interface layer is described by the temporally sparse sampling of the velocity field.

Tracking Graphs. A tracking graph is one of the most commonly used representations for visualizing feature evolution over time [21, 19, 7, 28]. However, as the data sets grow in size, the corresponding tracking graphs can be convoluted and contain many spurious events and difficult to comprehend crossings. In general, finding an optimal layout for these graphs is NP-hard [2]. Kasten et al. [18] consider the simpler problem of a tracking tree which can always be displayed without crossings. Widanagamaachchi et al. [28] use a greedy heuristic to minimize edge crossings by iteratively growing the graph starting at some focus time step. Here we employ a similar strategy but instead of only optimizing the layout we allow users to specify a small range of valid thresholds. Subsequently, we use the additional flexibility to simplify the graph on-the-fly by removing spurious events and artifacts.

3 Turbulent Combustion

Non-premixed combustion is employed in diverse areas of application ranging from diesel engines to gas turbines for power generation and aviation. One of the main challenges to using this mode of combustion for energy generation is the flame stability. The high Reynolds numbers found in most industrial applications generate significant flame-turbulence interactions which may result in a partial or complete blow off, decreasing efficiency and increasing unburned hydrocarbon emissions. It is therefore important to understand the details of this process. This requires a carefully designed

simulation that involves partial flame extinction due to strong turbulence. With this objective, a three dimensional DNS of a turbulent jet flame with di-methyl ether (DME) as fuel, undergoing extinction and re-ignition are performed [4]. The grid resolution is $920 \times 1400 \times 720$ with 101 snapshots stored for post-processing totaling ≈1500GB of data.

The turbulent jet flame is initialized as two shear layers with a stream-wise velocity $\Delta U/2$ in the center of the jet (fuel stream) and $-\Delta U/2$ outside it (oxidizer coflow), as shown in Figure 1(a). The fuel jet is comprised of 12% DME, 18% H_2, and 70% N_2 by volume while the oxygen enriched oxidizer stream is comprised of 31% O_2 and 69% N_2. The simulation parameters with respect to turbulence and chemistry are chosen such that significant local extinction and subsequent re-ignition is observed. These simulations represent the first time DME has been incorporated and the highest Reynolds number ever achieved in a fully resolved reacting direct numerical simulation. In trying to understand the mechanism of extinction and re-ignition in this flame, the focus is on a single surface termed the "stoichiometric mixture fraction isosurface", corresponding to the region of the flame where fuel and oxidizer are present in exactly the right amounts required to consume each other completely. We track various reacting scalar quantities on this surface as the flow field evolves in time to understand the details of extinction/re-ignition. Being able to track specific sub-regions on this reacting isosurface is crucial towards understanding the details of the physical process and thus tracking graphs of these regions are crucial for any subsequent analysis.

4 Background

As discussed above, we are interested in tracking regions of low OH concentration on the evolving mixture fraction isosurface. To correlate the isosurfaces over time we assume a piecewise linear interpolation in time and use the algorithm of [6] to compute a space-time isovolume. Furthermore, we use merge trees to encode extinction holes for variable thresholds to allow a flexible parameter exploration of the resulting tracks. Here, we briefly recap these two algorithms before describing our contributions in the later sections.

4.1 4D Isovolumes

Since the mixture fraction isosurfaces become highly complex over time we propose to create an explicit mesh that connects consecutive time steps rather than relying on a geometric heuristic to correlate features. In particular, our input data is given on a regular grid and for analysis purposes it is usually interpolated using trilinear interpolation in space. Consequently, we assume linear interpolation in time and use the marching hypercubes algorithm of Bhaniramka et al. [6] to extract a space-time mixture fraction, i.e. *isovolume*. Conceptually, one can think of this volume as the region of space (and time) swept by the moving isosurface of mixture fraction. In practice, an isovolume between each pair of consecutive time steps is defined as a tetrahedral mesh with three spatial and one time coordinate. We then interpolate the corresponding OH values at each vertex to produce a mesh defining the evolution of the extinction regions. Selecting all the triangles from the isovolume mesh whose vertices are entirely within one time step (rather than at an interpolated time between) results in the traditional isosurface of mixture fraction at the corresponding time step.

4.2 Merge Tree

While the mixture fraction isovalue is fairly standard and well established there exists no universal OH threshold value to indicate extinction and/or burning. Consequently, the goal is to explore different thresholds ideally without re-processing the data. The merge tree, a topological structure encoding the hierarchy of connected components of superlevel sets, has proved highly effective in encoding threshold-based features like the extinction holes in a parameter independent manner [9, 7, 3].

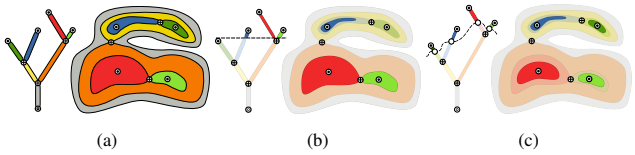


Figure 3: (a) The merge tree encodes the nesting relationships of superlevel sets. Each branch corresponds to a connected component of a superlevel set, and each vertex to a critical point where the connectivity changes. (b) Traditionally the tree is cut at a fixed threshold (horizontal line) resulting in a set of subtrees each representing one feature. (c) Localized thresholds create an arbitrary cut allowing a more flexible feature extraction. Images curtsey of [3].

Given a function f a *level set* $L(h) = \{v \in M | f(v) = h\}$ on a mesh M, $f : M \mapsto R$, is a set of points with the same isovalue h. A connected component of level set is a *contour*. A superlevel set is a union of level sets $S(h) = \cup_{h' \geq h} L(h')$. As the function value changes, contours nest inside each other and define a feature hierarchy which can be captured by a merge tree [9]. The merge tree is a rooted tree defined by a set of edges corresponding to equivalence classes of contours and a set of vertices corresponding to critical points where connectivity changes, as shown in Figure 3a. The root of tree is either global minimum (merge tree) or maximum (split tree). To extract components of a superlevel set the tree is cut at a selected threshold h, creating a forest of subtrees, as shown in Figure 3b. Each subtree corresponds to a connected component and thus to a connected region in the domain. As shown in Figure 3c, any cut that intersects each path from a leaf to the root at most once results in a valid segmentation. In particular, as discussed below one can use localized, per-feature thresholds for a more flexible feature definition. Here, we use the streaming algorithm of [8] to extract split trees of the OH concentration and store the results to interactively extract extinction regions at localized thresholds.

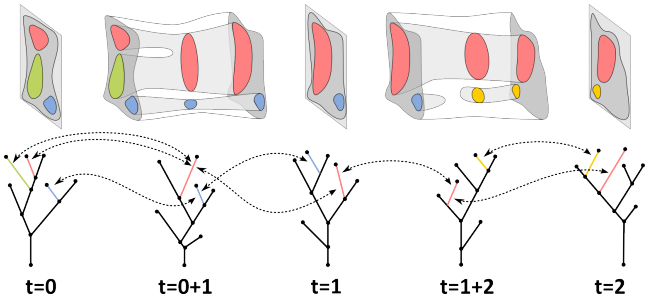
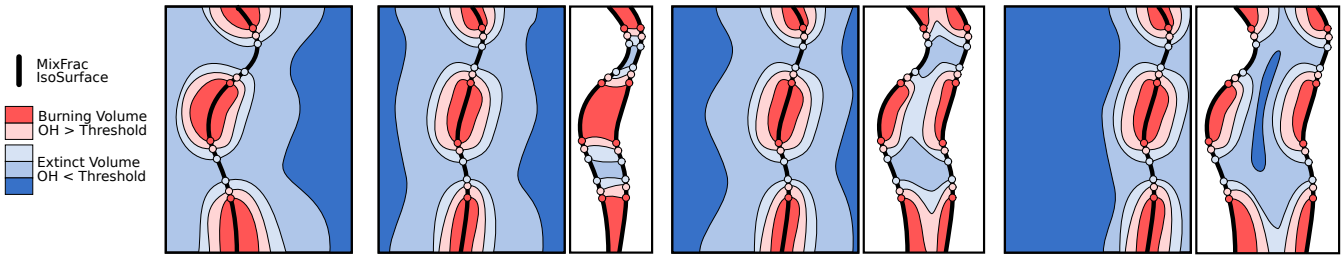


Figure 4: Parameter independent tracking of embedded features. Isosurfaces (shown as planes) of three consecutive time steps are shown with the merge tree of the second function indicated below. The space-time isovolume between time steps contains the surfaces at either end as well as vertices between steps. For illustration purposes some super-contours in the isovolume are shown in grey. Two branches in the trees of the original time steps and the isovolume that share vertices are correlated. Combined these correlations form the complete tracking information between all branches.

5 Correlating Embedded Features

This section presents a simple approach for parameter-independent tracking of embedded features with variable thresholds. Conceptually, we follow the approach of Widanagamaachchi et al. [28] which connects branches of merge trees over time to store the parameter independent tracking information using overlap tracking. However, for embedded features, i.e. regions on moving surfaces, overlap tracking does not apply directly since, in general, two consecutive isosurfaces do not share any vertices. As discussed in Sec-



(a) OH field at T_0 (b) OH field at T_1 and iso-volume T_0+T_1 (c) OH field at T_2 and iso-volume T_0+T_2 (d) OH field at T_3 and iso-volume T_0+T_3

Figure 5: Effects of decreasing temporal resolution. (a) An illustration of the neighborhood around a mixture fraction iso-surface. Small pockets of burning (high OH) fuel are separated by extinction holes (low OH). (b) A slow moving mixture fraction surface – T_0 to T_1 – creates easily tractable *tunnels* of low OH. (c) A moderately fast moving surface – T_0 to T_2 – artificially connects extinction holes. However, lowering the extinction threshold to the middle shade of blue can compensate for the artifact. (d) A fast moving surface – T_0 to T_3 – can create a region of artificially low OH (dark blue) connecting extinction holes that cannot be split by manipulating the threshold.

tion 4.1, we build an explicit space-time isovolume of mixture fraction between pairs of consecutive time steps and interpolate the corresponding OH values at all its vertices, i.e. $t = 0+1$ in Figure 4. Subsequently, we extract the isosurfaces within each time step and extract their merge trees, i.e. $t = 0$ and $t = 1$ in Figure 4. In principle, one can connect branches between the isosurfaces, i.e. $t = 0$ to $t = 1$, by traversing the connecting isovolume to determine whether two branches or rather their corresponding features are part of the same super-contour as indicated in light gray. However, this would require an explicit mesh data structure for the space-time isovolume and repeated costly traversals for all branches. Weber et al. [26] circumvent this problem by concentrating on a single OH threshold (a single super-levelset) and extracting its boundary (equivalent to the boundary of the light gray region in Figure 4). The Reeb graph of this surface describes the tracking graph for this single threshold. Instead, we propose a simpler approach that exploits the fact that all vertices in either time step are also part of the isovolume.

Given two branches b_i^0 and b_j^1, in the merge trees of time step 0 and 1 respectively, the question one has to answer is whether their corresponding features share the same super-contour in the space-time isovolume. If they do then there exists a path in the isovolume entirely contained in this super-contour that connects b_i^0 with b_j^1 and the two should be considered correlated in time. However, instead of tracing such a path directly this query can be answered by simply computing an additional merge tree of OH on the isovolume. This merge tree by design encodes the nesting and segmentation of all super-contours of OH. Therefore, for all vertices of all branches in $t = 0$ we determine their respective branches in the merge tree of the isovolume $t = 0+1$. Symmetrically, we can do the same for all branches in $t = 1$. Following the algorithm in [28] we then accumulate the tracking information towards the root to compute a per-feature (per-subtree) rather then a per-branch tracking. Effectively, we track all features in $t = 0$ forward and all features in $t = 1$ backward to an intermediate representation $t = 0+1$. Combined these provide the complete tracking information between both time steps. This approach replaces each step of the fixed threshold Reeb graph based tracking of [26] with two steps of simple overlap tracking (as vertices are shared between the surfaces and the volume) at the cost of an additional merge tree computation but providing complete flexibility in varying the threshold.

This scheme is simple to implement and very general. All that is required is a mesh that contains all the vertices of two consecutive time steps as well as an arbitrary number of additional vertices to form the connections. For each two time steps, three merge trees are computed, the complete one and the two restricted to the starting and ending time steps, then overlap tracking is used for computing the relevant tracking results.

6 Temporal Artifact Reduction

Once the merge trees and tracking information are computed, we have the capability to explore the evolution of extinction holes for a range of OH threshold values. For each time step or intermediate tree we select a threshold, extract the corresponding features, and connect features across time using the pre-computed tracking information. However, the resulting tracking graph shows the evolution of features assuming linear interpolation in time. Depending on the temporal resolution of the original data this may be incorrect. Consider the example shown in Figure 5(a)-(c). For a slow moving isosurface – T_0 to T_1 – the interpolated OH field on the space-time isovolume shows essentially the same structure as on the individual surfaces. Thus, the two independent extinction holes (blue sections of the surfaces) remain isolated and can easily be tracked. However, for a slightly faster moving isosurface – T_0 to T_2 – the temporal interpolation creates artifacts. In particular, the burning regions (red) are typically pancake like structures surrounding the flame surface. If the surfaces moves further than the width of the burning structures, the interpolation on the isovolume will create a single connect extinction region. This will result in the two extinction holes erroneously merging in the intermediate step before splitting again.

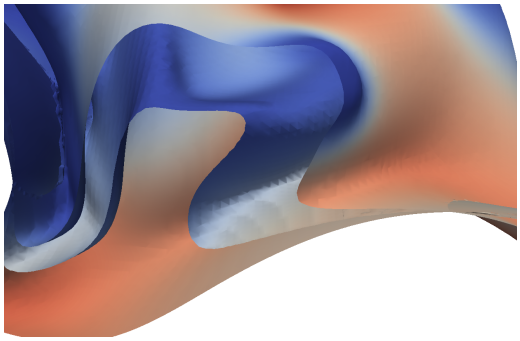
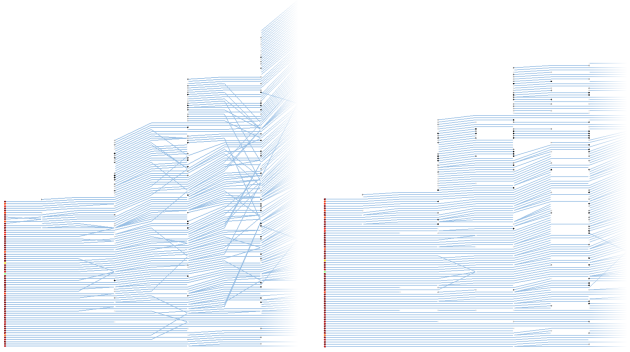


Figure 6: Interpolation artifacts due to insufficient temporal resolution: Detailed view of two mixture fraction isosurfaces colored by OH concentration. Both the bottom surface at $t = 0$ as well as the top surfaces at $t = 1$ is burning (red). However, the middle surface computed through linear interpolation at $t = 0.5$ is extinguished as illustrated in Figure 5d.

Clearly, this is an artifact of an insufficient temporal resolution, as shown in Figure 6 and should ideally be addressed through more frequent simulation snapshots. Unfortunately, for most large-scale simulations including the one considered here, the temporal resolution available for post-processing is strictly limited by the high cost of file I/O as well as storage space restrictions. Therefore, even if

the simulation could be repeated, which in itself is too costly to consider in practice, it would be infeasible to save significantly more data. In principle, tracking could be performed in-situ at the required temporal resolution. Unfortunately, this would require keeping at least two copies of the simulation state in memory which is typically not possible. Furthermore, any such processing would not be allowed to unduly increase the simulation time and/or file IO making it not viable at this point.

Instead, we propose to alleviate some of the artifacts by exploiting the flexibility to use localized thresholds. In particular, consider the case of Figure 5c: The interpolation artifact connects the otherwise separate extinction holes by a strip of relatively high, yet still extinct, OH values. To address this problem we selectively lower the OH threshold for features in the intermediate time step to effectively re-classify the light blue region as burning. This will separate the extinction holes across time and result in the correct tracking information. Unfortunately, while this approach alleviates many problems some artifacts are too severe to be corrected in this manner. As shown in Figure 5d, once the isosurface moves too fast a region of artificially low OH values, lower than those on the surface itself, is created in the isovolume. This artificial connection cannot be cut by lowering the threshold as the low (dark blue) valley would always exist.



(a) (b)

Figure 7: (a) Early time steps of the original tracking graph of extinction holes with a large number of artifacts causing artificial merge-split events. (b) The graph of (a) after artifact removal.

In practice, we first create the tracking graph for a single threshold as discussed above. We then search all intermediate time steps for high valence, merge-split nodes indicating a potential artifact. For each of these we progressively lower the local threshold until it either splits into individual nodes, resolving the artifacts, or a further lowering would increase the number of non-valence two nodes in the tracking graph by disconnecting holes entirely. Note that, at any point the features used in the tracking are valid extinction regions though potentially using slightly different thresholds. Only the tracking information in the intermediate time step is directly affected and only if it decreases the number of non-valence two nodes. Nevertheless, with the ground truth tracking information unavailable and the linear interpolation containing known artifacts this approach remains a heuristic albeit a very effective one. Figure 7 shows an example of the first couple of time steps of the tracking graph of all extinction regions is shown before (a) and after (b) simplification. The approach described above is a very simple and yet effective approach to compensate for the insufficient temporal resolution and produce much simpler graphs that can more intuitively express the fundamental trends in the data.

7 Tracking Graph Simplification

Even assuming a sufficiently high temporal resolution, tracking graphs often contain another class of spurious events. Virtually all feature definitions are slightly unstable in the sense that a feature may hover right at the threshold. Over time, depending on whether the threshold is slightly above or below, these features can cause repeated merges and splits resulting in spurious merges and splits in the tracking graph. Typically, these events convey little information of interest but clutter the view and make post-processing of the graphs unnecessarily difficult. Furthermore, in most applications the specific choice of threshold is rather arbitrary and any value within some range is acceptable to the scientists. Therefore, we propose to exploit the flexibility in choosing a threshold to simplify the graph within a tightly controlled error bound. In particular, we introduce a new progressive three-pass layout algorithm to simplify the tracking graphs by locally adapting the feature thresholds within a user defined range. In practice, these changes correspond to adjusting the cut-points within the merge tree to create an arbitrary cut rather than a horizontal one. The resulting graphs are more stable, easier to interpret, and better at representing the fundamental trends within the data. Conceptually, this is similar to the artifact removal of Section 6 but instead of only addressing specific configurations in the intermediate time steps we make the threshold somewhat malleable to simplify the graph globally.

The first step is to define an objective function with respect to which the graph should be optimized. For both post-process analysis and visualization we prefer long tracks of valence two nodes or in other words features that evolve in isolation. Therefore, we focus on reducing the total number of non-valence two nodes which includes both splits and merges with valences greater than two as well as births and deaths with valence one. In general, computing the optimal graph with the least number of non-valence two nodes within a given threshold range is NP-hard. Instead, we introduce a greedy heuristic that is both efficient to implement and has been shown to be highly effective in practice.

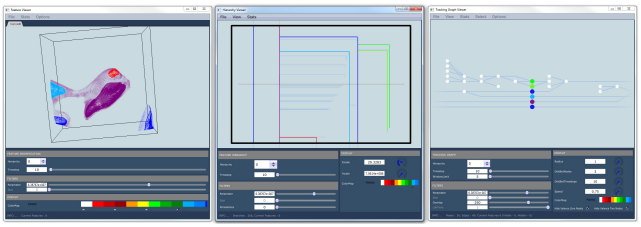
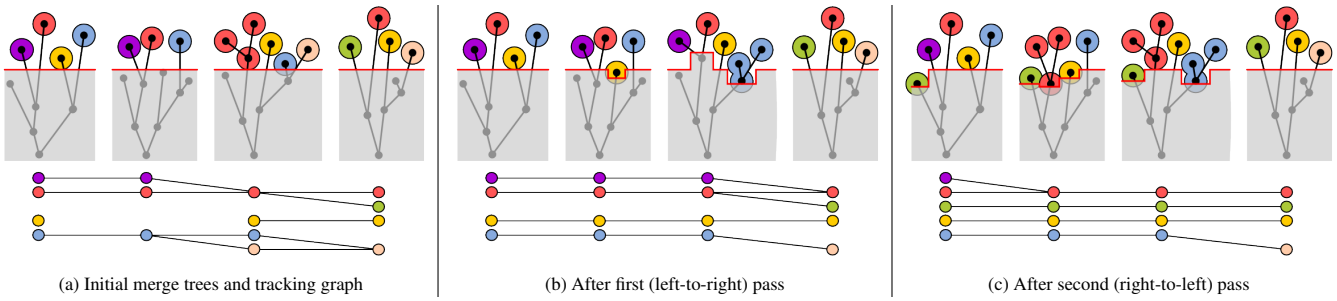


Figure 9: Our interactive linked-view system consists of multiple views, from left to right: feature embedding view, feature hierarchy view, and feature evolution view. The range of potential outcomes for the user-defined threshold range within the current time step are displayed within both feature hierarchy and feature embedding view. Here, the counterflow flame combustion data set has been used and the user-defined fuzzy region is outlined in ‘black’.

Given a range of acceptable thresholds or alternatively an error bound around an ideal threshold we optimize the graph in three passes: left-to-right, right-to-left, and a final left-to-right. Conceptually, we always treat features at the “previous” time step (depending on the pass this could either be earlier or later in time) as fixed and optimize the current features accordingly. During the first two passes we adjust the graph greedily by delaying any event that causes a non-valence two node. More specifically, when encountering a split we merge the corresponding features, if it is possible within the parameter range. For example, in Figure 8a the blue feature temporarily appears to split before merging into the light pink one. A closer look at the merge trees shows that effectively the blue feature is slowly being replace by the pink one and the split-merge event is due to both existing simultaneous for one time step. As shown in Figure 8b, locally lowering the threshold delays the split and incidentally also resolves the later merge creating a single clean history. Similarly, when encountering a merge we attempt to



(a) Initial merge trees and tracking graph (b) After first (left-to-right) pass (c) After second (right-to-left) pass

Figure 8: Using localized threshold to simplify a tracking graph. (a) The initial tracking graph and corresponding merge trees with several nodes of the trees close to the fixed threshold. (b) After the first (left-to-right) pass the yellow death-birth event and the blue split-merge have been avoided by slightly lowering the threshold. The purple merge has moved through raising the threshold but ultimately cannot be corrected as the purple feature disappears. (c) After the second (right-to-left) pass the red-green split has been corrected by slightly lowering the threshold. The purple merge has moved to the left but still cannot be avoided as the saddle necessary to effect a merge is outside the acceptable range. The third pass (not shown) would move the purple merge back to its original position to minimize the deviation from the original threshold.

adjust the local parameter to split the features. For example, in the first pass shown in Figure 8b the purple merge has been delayed by one time step even though in this case it ultimately could not be resolved. Finally, births and deaths will be delayed, i.e. the yellow feature in Figure 8b. Figure 8c shows the situation after the second (right-to-left) pass which has resolved the green split and move the purple one further to the left.

The third pass is slightly different and not designed to further simplify the graph but rather to minimize the deviation in threshold. The first two passes simplify the graph by canceling merges with splits (the blue feature) or birth with deaths (the yellow feature). However, as a side-effect they also move unavoidable merges and splits first as far right and then as far left as possible (the purple feature). While these changes are valid within the error bound they do not structurally improve the graph and thus needlessly distort the threshold. The final left-to-right pass corrects this problem by moving these event back to their original location. For example, in Figure 8 the red-purple merge would move back to its original spot of Figure 8a.

In practice, this simplification process is driven by the user selecting a focus time step and a threshold range. As shown in Figure 9, our system shows the geometry of all features at the focus time steps at both ends of the threshold range as well as a dendrogram like display of the corresponding merge tree to help select the range. Furthermore, the system interactively updates the tracking graph as parameters change. Finally, we support two modes depending on whether the focus time step should be modified or not. In the first case we process the entire graph simultaneously in the three passes described above. However, given that the user chose a specific threshold at the focus time step it is reasonably to assume that these features are "correctly" selected and should be taken as ground truths. In this case, the simplification proceeds simultaneously both forward and backward in time keeping the focus time step unmodified. Not that in this case we only require two passes as any event that has not been resolved in the first pass cannot be resolved at all. For example, in Figure 8 the red-green split could not be resolved if the first time step is kept fixed. Figure 10 shows a practical example in another combustion use case tracking extinction holes in a flame burning in a cross-flow.

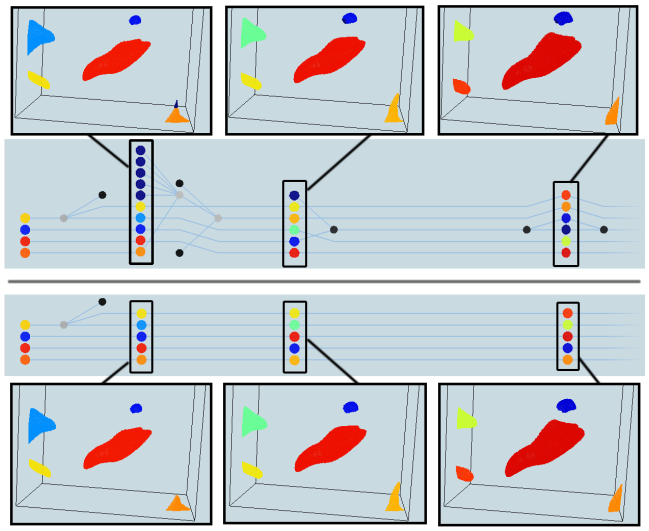


Figure 10: A example of a tracking graph simplification in the counterflow combustion simulation. Both the features and the tracking graphs before (top) and after (bottom) the simplification are shown. Note that most features that are simplified are too small to see at this scale and thus likely not significant. Yet without simplification these artifacts create a substantially more complex graph.

8 Results

We demonstrate the effectiveness of our tracking algorithm using a combustion data set as a primary case study. This combustion data set is from a three dimensional direct numerical simulation of a turbulent jet flame and contains 101 time steps at the resolution of $920 \times 1400 \times 720$ which even considering only the two scalar fields used here totals $\approx$1.5TB of data. For each pair of consecutive time steps we first extract the mixture fraction isovolumes, from these extract the corresponding isosurfaces and compute the necessary split trees. Finally, we extract the tracking information between consecutive pairs of time steps. All this is a one-time preprocessing that is embarrassingly parallel for each pair of consecutive time-steps. The analysis for the turbulent jet has been performed in parallel at the Oak Ridge Leadership Computing Facility on Rhea a Linux cluster with 16, 2.0 GHx Intel Xeon processors per node. The other two examples have been processed at NERSC on the Carver cluster with 16, 2.67 GHz Intel Xeon processors per node. Typically, processing each pair of time steps takes between 5 (AMR combustion) and 60 minutes (turbulent jet). The resulting data consisting of the trees as well as the corresponding segmentation and tracking information and for the turbulent jet reduces to about 6GB the vast majority of which is dedicated to the spatial information for rendering. The remaining processing is done on-the-fly on the workstation or laptop of the user. In this manner, our system, for the first time, allows an interactive exploration of the temporal evolution of features from several TeraBytes of data.

Our approach enables scientists to explore the evolution of ex-

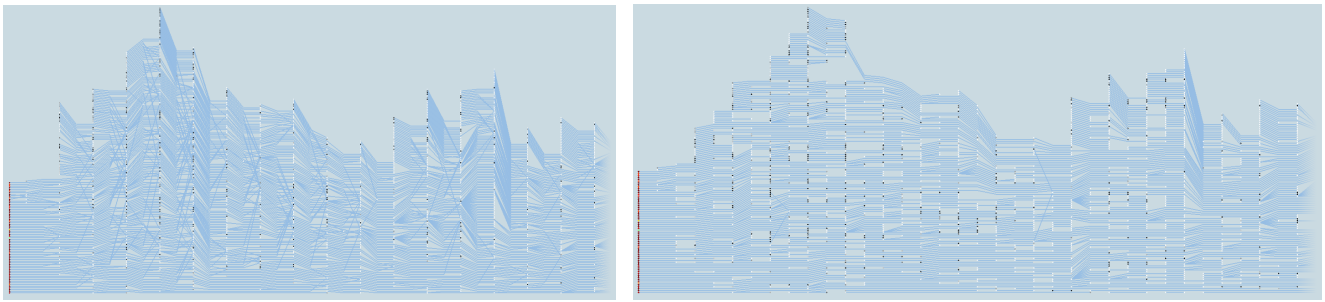


Figure 11: Tracking graph of all extinction holes in the turbulent jet flame for the first 35 time steps. (Left) Original graph; (Right) The graph of (a) after artifact removal and simplification.

tinction regions for a range of OH concentration values. More importantly the techniques introduced above allow us to remove temporal artifacts as well as simplify the graph. For example, figure 2 shows the evolution of extinction regions at the default OH concentration value of $9.77e^{-4}$ resulting in a complex, difficult to comprehend and process graph. Instead, our system interactively removes many of the temporal artifacts (Figure 7b) and allows users to simplify the graph within a strictly controlled error bound. The resulting graph shown in Figure 11 is significantly simpler, more intuitive, and easier to process.

To demonstrate the generality and versatility of our approach we have applied it to a number of other large-scale simulations in the area of combustion. Figure 9 and 10 show the simulation of flame in a highly turbulent premixed counterflow. The original data has a resolution of $432 \times 640 \times 640$ at 1048 time steps totaling around 2.7TB of data. Similar to the first application the features of interest are extinction holes though they are defined slightly differently. unlike the turbulent jet flame the temporal resolution is quite high and the flame surface rather stationary. Therefore, one can use a simple distance based criterion to track features. This data contains far fewer features but due to the nature of the analysis more parameter instabilities. As shown in Figure 10 our system is able to extract clean graphs for visualization and analysis.

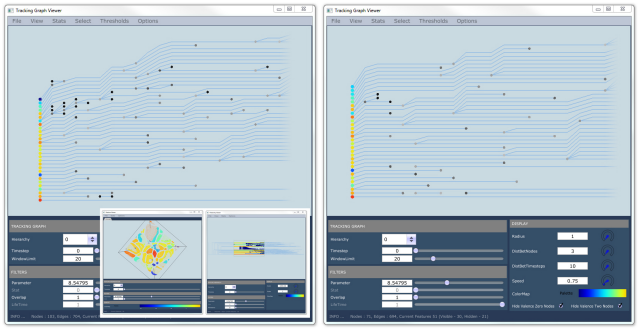


Figure 12: An example where (a) a tracking graph is adjusted within the user-defined fuzzy region, shown in bottom left and bottom right, to produce (b) a more temporally cohesive graph for the hydrogen flame combustion data set.

The final analysis describes an idealized premixed hydrogen flame [11] and contains 100 time steps of an adaptive mesh resolution (AMR) simulation at an effective resolution of $256 \times 256 \times 768$ and about 400GB of raw data. Here, the burning cells within the simulation, i.e. regions with high fuel consumption, are considered to be the features of interest and their correlation details are computed using spatial overlap. Figure 12 shows an example of simplification to reduce the complexity of the graph reducing it to show only the salient features. Figure 13 demonstrates some additional capabilities of our system namely the ability to interactively sub-select graphs based on space or a feature of interest. For more results on all data sets, we refer the reader to the accompanying supplementary images and video.

9 Limitations

Localized thresholds can significantly improve the overall tracking of embedded features and given some error tolerance can also simplify tracking graphs for easier interpretation. However, both approaches presented in this paper remain limited to the feature definition given by the merge tree. In particular, for the extinction holes we can break some artificial connections in the tracking graph by applying a more aggressive, i.e. lower threshold. However, this only applies in cases where the artificial features in the in-between time are weaker, i.e. more shallow, than the features in neighboring time steps. Unfortunately, this is not always the case, as shown in Figure 5d. Given that in most practical cases one cannot produce more data there might be other heuristics that could be applied to address these cases. For example, detecting when the intermediate feature has a significantly lower OH value than the surfaces would be reasonably straight forward. Then a more aggressive geometry based heuristic could be applied.

As demonstrated by our results the three-pass layout algorithm is very effective in simplifying a graph to minimize spurious events. However, due to the greedy nature of the algorithm each step in the sweep might be optimal but there exists a clear order dependency. Therefore, we cannot guarantee that the final graph is optimal, i.e. has the lowest number of non-valence two nodes. Furthermore, reducing the number of non-valence two nodes may not be the ideal metric in all applications. For example, the current metric favors fewer features in general and in certain cases might be prone to suppress features. Nevertheless, we have applied our system to a wide variety of case even beyond the results shown here with very positive results.

10 Conclusion

In this paper, we have presented a novel tracking algorithm for embedded features which requires only a merge tree as the underlying abstraction. The use of the merge trees provides our algorithm the additional flexibility to allow for arbitrary parameter selection interactively. Additionally, we demonstrate that previous approaches such as overlap tracking is a subset of the proposed algorithm and can be easily implemented using our approach without any ad-hoc tracking computation which further improves the generality of the algorithm.

For the first time, we allow tracking graphs to contain localized thresholds to enable removal of artifacts due to insufficient temporal resolution as well as from threshold instabilities. We further enable a new user interaction which allows selection of flexible feature definition parameters in tracking graphs and presents users with an intuitive rendering of the range of outcomes. These functionalities along with the capability to interactively define, select, and progressively layout subsets of large tracking graphs enables

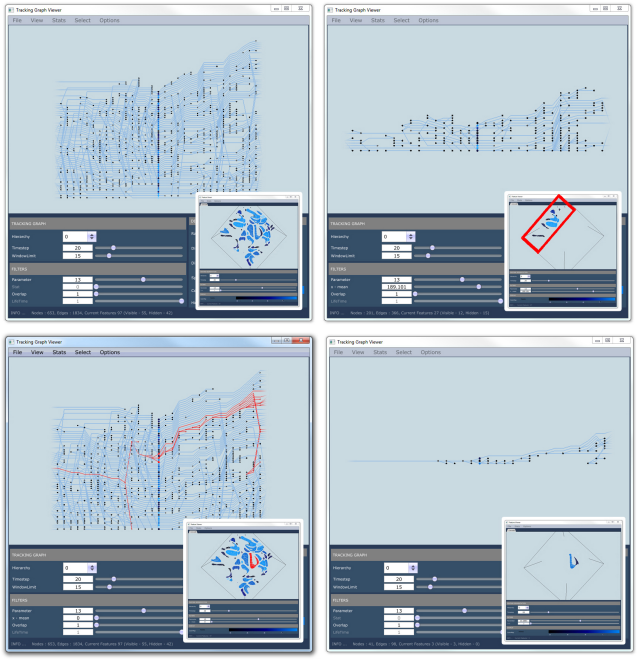


Figure 13: Extracting local graphs for a set of focus features which are defined in terms of space (top) or time (bottom) in the AMR combustion example. In each case, the tracking graphs are shown before (left) and after (right) the selection.

our framework to explore feature evolution and their feature selection parameters more flexibly than the traditional approaches. Finally, using several different data sets from combustion science we demonstrate the applicability and generality of our approaches.

Acknowledgements

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) program. This work was also performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

References

[1] C. Bajaj, A. Shamir, and B.-S. Sohn. Progressive tracking of isosurfaces in time-varying scalar fields, 2002.

[2] O. Bastert and C. Matuszewski. Layered drawings of digraphs. In M. Kaufmann and D. Wagner, editors, *Drawing Graphs*, volume 2025 of *Lecture Notes in Computer Science*, pages 87–120. Springer Berlin Heidelberg, 2001.

[3] J. Bennett, V. Krishnamurthy, S. Liu, V. Pascucci, R. Grout, J. Chen, and P.-T. Bremer. Feature-based statistical analysis of combustion simulation data. *IEEE Trans. Vis. Comp. Graph.*, 17(12):1822–1831, 2011.

[4] A. Bhagatwala, Z. Luo, H. Shen, J. A. Sutton, T. Lu, and J. H. Chen. Numerical and experimental investigation of turbulent {DME} jet flames. *Proceedings of the Combustion Institute*, 35(2):1157 – 1166, 2015.

[5] P. Bhaniramka, R. Wenger, and R. Crawfis. Isosurfacing in higher dimensions. In *Proceedings of the Conference on Visualization '00*, VIS '00, pages 267–273, Los Alamitos, CA, USA, 2000. IEEE Computer Society Press.

[6] P. Bhaniramka, R. Wenger, and R. Crawfis. Isosurface construction in any dimension using convex hulls. *IEEE Trans. Vis. Comp. Graph.*, 10(2):130–141, Mar 2004.

[7] P.-T. Bremer, G. Weber, V. Pascucci, M. Day, and J. Bell. Analyzing and tracking burning structures in lean premixed hydrogen flames. *IEEE Trans. Vis. Comp. Graph.*, 16(2):248–260, Mar. 2010.

[8] P.-T. Bremer, G. Weber, J. Tierny, V. Pascucci, M. Day, and J. Bell. Interactive exploration and analysis of large-scale simulations using topology-based data segmentation. *IEEE Trans. Vis. Comp. Graph.*, 17(9):1307–1324, Sept 2011.

[9] H. Carr, J. Snoeyink, and U. Axen. Computing contour trees in all dimensions. *Computational Geometry*, 24(2):75–94, 2003.

[10] H. Carr, J. Snoeyink, and M. van de Panne. Simplifying flexible isosurfaces using local geometric measures. In *IEEE Visualization '04*, pages 497–504. IEEE Computer Society, 2004.

[11] M. Day, J. Bell, P.-T. Bremer, V. Pascucci, V. Beckner, and M. Lijewski. Turbulence effects on cellular burning structures in lean premixed hydrogen flames. *Combustion and Flame*, 156(5):1035 – 1045, 2009.

[12] H. Edelsbrunner, J. Harer, A. Mascarenhas, and V. Pascucci. Time-varying Reeb graphs for continuous space-time data. In *20th Symp. on Computational Geometry*, pages 366–372, New York, NY, USA, 2004. ACM, ACM Press.

[13] I. Fujishiro, Y. Maeda, and H. Sato. Interval volume: A solid fitting technique for volumetric data display and analysis. In *Proceedings of the 6th Conference on Visualization '95*, VIS '95, pages 151–, 1995.

[14] C. Garth, X. Tricoche, and G. Scheuermann. Tracking of vector field singularities in unstructured 3d time-dependent datasets. In *Visualization, 2004. IEEE*, pages 329–336, Oct 2004.

[15] A. Gyulassy, P.-T. Bremer, B. Hamann, and V. Pascucci. A practical approach to morse-smale complex computation: Scalability and generality. *IEEE Trans. Vis. Comp. Graph.*, 14(6):1619–1626, Nov. 2008.

[16] G. Ji and H.-W. Shen. Efficient isosurface tracking using precomputed correspondence table. In *Proceedings of the Sixth Joint Eurographics-IEEE TCVG conference on Visualization*, pages 283–292. Eurographics Association, 2004.

[17] G. Ji, H.-W. Shen, and R. Wenger. Volume tracking using higher dimensional isosurfacing. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, VIS '03, pages 28–, 2003.

[18] J. Kasten, I. Hotz, B. R. Noack, and H.-C. Hege. Vortex merge graphs in two-dimensional unsteady flow fields. In *EuroVis – Short Papers*, pages 1 – 5, 2012.

[19] D. Laney, P.-T. Bremer, A. Mascarenhas, P. Miller, and V. Pascucci. Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. *IEEE Trans. Visualization and Computer Graphics (TVCG) / Proc.of IEEE Visualization*, 12(5):1052–1060, 2006.

[20] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, Aug. 1987.

[21] F. Reinders, F. H. Post, and H. J. W. Spoelder. Visualization of time-dependent data using feature tracking and event detection. *The Visual Computer*, 17:55–71, 2001.

[22] R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing features and tracking their evolution. *Computer*, 27(7):20–27, July 1994.

[23] D. Silver and X. Wang. Tracking and visualizing turbulent 3d features. *IEEE Trans. Vis. Comp. Graph.*, 3(2):129 –141, apr-jun 1997.

[24] D. Silver and X. Wang. Tracking scalar features in unstructured data sets. In *Visualization '98. Proceedings*, pages 79–86, Oct 1998.

[25] F.-Y. Tzeng and K.-L. Ma. Intelligent feature extraction and tracking for visualizing large-scale 4d flow simulations. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, page 6. IEEE Computer Society, 2005.

[26] G. Weber, P.-T. Bremer, M. Day, J. Bell, and V. Pascucci. Feature tracking using reeb graphs. In *Topological Methods in Data Analysis and Visualization*, Mathematics and Visualization, pages 241–253. 2011.

[27] T. Weinkauf, H. Theisel, A. V. Gelder, and A. Pang. Stable Feature Flow Fields. *IEEE Transactions on Visualization and Computer Graphics*, 17(6), 2011.

[28] W. Widanagamaachchi, C. Christensen, P.-T. Bremer, and V. Pascucci. Interactive exploration of large-scale time-varying data using dynamic tracking graphs. In *IEEE Symposium on Large Data Analysis and Visualization (LDAV), 2012*, pages 9–17, Oct 2012.