Feature Tracking Utilizing a Maximum-Weight Independent Set Problem

Andrea Schnorr*
JARA – HPC
Visual Computing Institute,
RWTH Aachen University

Dirk N. Helmrich
JARA – HPC
Visual Computing Institute,
RWTH Aachen University

Hank Childs
University of Oregon

Torsten W. Kuhlen
Visual Computing Institute,
RWTH Aachen University
JARA – HPC

Bernd Hentschel
Visual Computing Institute,
RWTH Aachen University
JARA – HPC

ABSTRACT

Tracking the temporal evolution of features in time-varying data remains a combinatorially challenging problem. A recent method models event detection as a maximum-weight independent set problem on a graph representation of all possible explanations [35]. However, optimally solving this problem is NP-hard in the general case. Following the approach by Schnorr et al., we propose a new algorithm for event detection. Our algorithm exploits the model-specific structure of the independent set problem. Specifically, we show how to traverse potential explanations in such a way that a greedy assignment provides reliably good results. We demonstrate the effectiveness of our approach on synthetic and simulation data sets, the former of which include ground-truth tracking information which enable a quantitative evaluation. Our results are within 1% of the theoretical optimum and comparable to an approximate solution provided by a state-of-the-art optimization package. At the same time, our algorithm is significantly faster.

Keywords: Feature Tracking, Graph Optimization, Approximation, Flow Visualization, Dissipation Elements

1 INTRODUCTION

The analysis of time-varying phenomena is a key method in scientific visualization. The goal of feature tracking is to gain insight in the spatio-temporal evolution of objects by automatically identifying their respective correspondences in subsequent time steps. Classically, feature tracking approaches have focused on the tracking of sparse features, i.e. structures which cover only a small portion of the domain. Given a sufficiently high temporal resolution, these approaches provide a robust way to match corresponding features. In contrast, we are especially interested in the analysis of space-filling features, i.e. structures which partition the entire domain. This is largely motivated by our collaborators' recent research in turbulent fluid mechanics, which brought up the definition of *dissipation elements*, a space-filling structure that partitions a flow simulation's domain into regions of homogeneous behavior of a scalar field's gradient [33, 44].

Including space-filling structures complicates the tracking due to the high number of potentially contradicting candidate explanations. Because several feature objects might be involved in an event, finding an optimal explanation with respect to some similarity criterion involves searching the power set of all candidate objects. Thus, the number of available explanations grows exponentially with the number of eligible objects. Several of these candidates will regularly contradict each other because they assign the same objects in different ways, which adds to the overall challenge.

*e-mail: schnorr@vr.rwth-aachen.de

Recently, we introduced an approach that addresses these challenges by solving two successive graph optimization problems [35]. The first step solves a maximum-weight, bi-partite matching problem in order to identify linear one-to-one assignments (cf. Figure 1 a)). Based on this initial solution, the second step models the event detection stage as a maximum-weight independent set problem. While we use the initial matching stage to limit the search for event explanations in the subsequent event detection stage, an optimal solution to the latter remains NP-hard.

In this paper, we focus on that second stage. We propose to exploit the specific structure of the independent set graph which results from our way of modeling the event detection problem. In a nutshell, this graph consists of a set of cliques of mutually contradicting event explanations plus a number of cross-clique connections. Figure 1 b) gives a schematic example. A valid, contradiction-free solution to the event detection problem is equivalent to a subset of nodes which contains exactly one node per clique, i.e. will select one of the available explanations, such that no two selected nodes are connected by an edge; this is the independent set property. Our main idea is to align the optimization process along these cliques in order to create a greedy assignment strategy. The details of this construction are given in Section 3.

In Section 4 we analyze the performance of our approach and show that despite its relatively straightforward greedy assignment it provides reliably good solutions for a number of benchmark problems. Specifically, we use both synthetic and simulation data sets for the evaluation. The former were specifically generated for evaluation purposes, because they enable a comparison to ground truth data. The latter result from direct numerical simulation of homogeneous, isotropic turbulence. They are provided by our collaborators from the Institute for Combustion Technology at RWTH Aachen University, who – as outlined above – also provide the main motivation for this research. We compare the event detection results and runtime performance to our earlier approach, which relies on the *CBC* solver of the *COIN-OR* project [1]. In Section 5, we discuss our approach with an eye on known limitations and more general observations, before concluding in Section 6.

In summary, we make the following contributions. First, we propose an optimized event detection method for feature tracking, which is based on a careful analysis of the specific structure of the underlying, model-specific independent set problem. Second, we demonstrate the effectiveness and efficiency of this approach by applying it to different synthetic and simulation data sets and comparing the results to the solution found using the external *CBC* solver.

2 RELATED WORK

Feature-based visualization is a core concept in scientific visualization. It has particularly been used in the context of flow visualization [27]. In general, features are loosely defined as *objects of interest* (cf., e.g., [37]). Before tracking their evolution, however, features have to be extracted. A full review of feature definitions

IEEE LDAV 2019
21 October, Vancouver, BC, Canada
978-1-7281-2020-1/19/$31.00 ©2019 IEEE

and extractions is beyond the scope of this paper. Typical examples include vortices [2, 15, 16, 41, 45], shocks [20, 24, 25], or topological structures [10, 12–14, 34]. These examples share the property that they are sparse with respect to the input domain, i.e. they cover a relatively small portion of the data domain. This fact has been highlighted as a key advantage, because it substantially reduces the amount of data that is required for subsequent analysis [43]. In contrast, our approach targets a specific feature definition: *dissipation elements* [44] which are space-filling by definition.

Feature tracking algorithms aim at identifying the temporal evolution of features in a series of raw data snapshots $\{S_t\}_{1\leq t\leq T}$. Generally, it is assumed that the snapshots are sampled densely enough in the temporal dimension to allow for a faithful reconstruction of temporal relations.

However, finding a globally optimal assignment – i.e. one that maximizes the feature similarity – over all T time steps is NP-complete for $T \geq 3$ [18, 26] and remains a challenge in scientific visualization [17]. In scientific visualization, approaches are often based on the idea of extending a feature path from time step t_i to t_{i+1}. This idea was introduced by Samtaney et al. [31]. They define correspondence criteria for the canonical event explanations of continuation, split, merge, birth, and death. In order to determine the evolution of features, they propose a greedy strategy, which implicitly resolves conflicts by removing feature objects from the search space once a correspondence is found. Silver and Wang utilize the normalized volume difference of objects in order to establish correspondences [39]. This method implicitly builds a bi-partite graph w.r.t. the overlap between feature objects from subsequent time steps. Event detection is included by comparing a single object from one time step to all combinations of overlapping feature objects in the neighboring time step. The used overlap metric has later been extended to unstructured meshes [40], and AMR data [6], and has been implemented on distributed memory parallel machines [5].

Inspired by a computer vision approach [36], Reinders et al. perform similarity computation based on abstract *attribute sets* [28]. They describe features by a finite number of attributes, extrapolate these attributes to the target time step, and establish correspondences by comparing the extrapolation to the attributes of candidate objects. In order to facilitate interactive feature tracking of individual features, Muelder and Ma combine the idea of extrapolation with the overlap metric [21].

Clyne et al. develop a physical model to establish feature correspondences utilizing the underlying governing equations [7]. In order to identify a match in a subsequent time step, they use a pathline starting at the point of minimal dissipation. Sauer et al. also make use of particle paths to determine feature correspondence between time steps, they propose an approach which uses pre-computed particle data in order to match features over several time steps [32].

Theisel et al. formulate the tracking problem by means of streamline integration in the *feature flow field* [42] and show the effectiveness by tracking critical points in a 2D vector field. This approach was later refined w.r.t. (numerical) stability by Weinkauf et al. [46]. Reininghaus et al. refrain from numerical computations altogether by introducing combinatorial feature flow fields [29]. Garth et al. propose an approach which is capable to track vector field singularities by explicitly detecting structural changes in the vector field in the form of bifurcations [9].

Tracking algorithms open up a number of possibilities for a follow-up analysis of the data sets under consideration. Ozer et al. propose an approach to investigate the group behavior of multiple features [23]. Additionally, they introduced an approach which uses Petri Nets in order to detect inter-feature interactions [22]. Griffith et al. discuss the tracking and subsequent analysis on simulated cloud formations [11], while Doraiswamy et al. base their approach on 2D satellite imagery [8]. Laney et al. track surface segments in turbulent mixing processes with a method related to the one by Samtaney et al. They use topological concepts in order to segment the interface surface [19]. Similarly, Bremer et al. propose an approach in order to analyze the flame surface in a simulation of turbulent combustion [4].

Recently, Saikia and Weinkauf proposed a method which also utilizes ideas from graph optimization in order to solve the tracking problem [30]. In contrast to our approach, they aim at maximizing the similarity over all time steps for a single feature. In order to do so, they establish a directed acyclic graph which represents the tracking problem over all time steps. On this graph, they use Dijkstra's single source shortest path algorithm to find the temporal evolution of a single feature.

3 METHOD

As stated in the introduction, we follow up on the approach by Schnorr et al. [35]. They model the tracking problem between subsequent time steps by two successive graph optimization problems. The first phase uses a maximum-weight maximum-cardinality matching on a bi-partite graph to establish a set of linear continuations. These make up the vast majority of explanations – an assumption that is backed by previous work [39] and domain experts [44]. The linear assignments designated by the matching either directly represent a continuation or identify the largest component participating in a merge or split event.

In the second step, events are detected by augmenting the matching edges with additional connections which results in $1:n$ or $n:1$ relationships. This is done by constructing a graph that contains all non-trivial, potential event explanations – i.e., splits and merges – for all feature objects which have not been covered by the matching in the first phase. The construction of this graph is informed by the matching and, thus, the connections identified by the matching are part of any valid explanation. While nodes in this graph model potential explanations, edges signify mutual contradictions between explanations. These contradictions arise from the fact that two explanations assign at least one specific feature object to an event in different ways. Hence, not both these events can be valid at the same time. A valid, i.e. conflict-free assignment thus forms an independent set on this graph. By adding weights to each node, i.e. the similarity score of that specific explanation, this is turned into a maximum-weight independent set problem. Finally, all nodes that are not assigned after both optimization problems are assumed to be involved in a birth or death event.

In order to find an approximate solution to the NP-hard independent set problem, Schnorr et al. rely on the *CBC* solver of the $COIN-OR$ project. They suggest to find a solution within 99% of the optimal score. In contrast, we propose a new algorithm to solve the independent set problem by exploiting the specific structure of the underlying graph. In the following we will first detail this structure and then describe our greedy assignment strategy which eventually leads to a faster solution process of comparable solution quality.

3.1 Graph Construction

The maximum-weight independent set problem is defined over a graph $G = (V, E, w)$ on nodes V, edges E and a weight function w which assigns a weight *to each node*.

The nodes model potential explanations. We construct these event explanations by traversing the matching edges resulting from the initial phase. A schematic example of the matching resulting from the initial phase is given in Figure 1 a). Matching edges are depicted with thick edges. All other edges are edges with a positive similarity value and, hence, valid but not part of the matching solution. First, all possible split explanations are constructed for each edge (o_s, o_t) covered by the matching. A split is a $1:n$ assignment and, thus, contains one node from the node set in time step T_i and a set of nodes from T_{i+1}. In order to find all valid explanations, we enumerate all

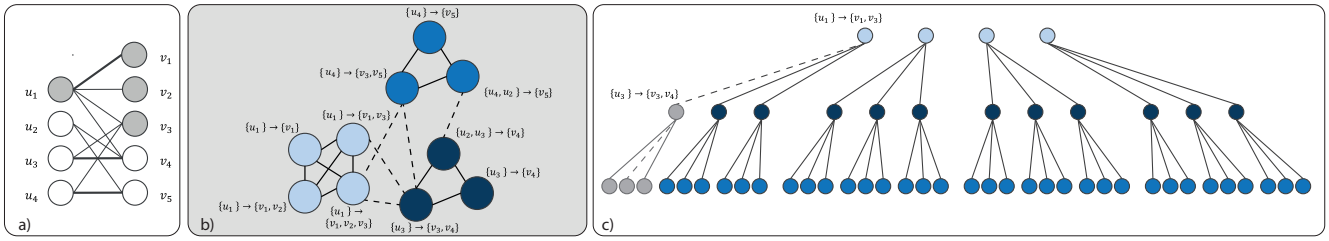

Figure 1: Illustration of our graph construction and the resulting search tree. a) The bi-partite matching resulting from the initial phase of our algorithm. One-to-one assignments found by the matching are depicted as thick lines. All other valid edges are depicted as thin lines. Additionally, all nodes considered as participating in a split with the node u_1 are highlighted in gray. b) For each matching edge a set of explanations is constructed to augment the initial matching solution in order to detect events. This results in a graph containing all non-trivial, potential event explanations. Nodes in this graph resemble potential explanations; edges model mutual contradictions between these explanations. All nodes resulting from the same matching edge are conflicting and, hence, form a clique in the graph. Thus, we can choose at most one node in each clique. Since each clique has a trivial choice given by the continuation (only the matching edge with no additional edges) which has no edge to any other clique, we can choose *exactly one* edge per clique. c) The search space can thus be modeled as a search tree over the set of cliques and the selection is essentially constructed by a depth-first search over this search tree taking the cross-clique conflicts into account. A cross-clique conflict for the nodes $u_1 \rightarrow v_1, v_3$ and $u_3 \rightarrow v_3, v_4$ is depicted by a dashed line between the nodes. Since this path does not resemble a valid solution, the subtree is grayed out.

currently unmatched objects $\tilde{o} \in T_{i+1}$ which are connected by a valid edge – i.e. an edge in the bi-partite graph with a positive similarity score – to the matched node $o_s \in T_i$ given by the matching edge. In Figure 1 a) the node u_1 is connected to v_1 by a matching edge. Additional candidates which are connected to u_1 are v_2 and v_3. The power set of all these candidates is computed and for each element of the power set, we construct one event explanation. This explanation consists of the two nodes given by the matching edge and the additional nodes given by one element of the power set. Thus, each element of the power set results in one possible explanation.

The original continuation represented by the matching edge is included by using the empty set as extension, i.e. not adding additional nodes. In Figure1 b) the resulting nodes which model the potential explanations are illustrated for the three matching edges given in the schematic example. All possible merge explanations based on the node o_t of the matching edge are constructed analogously. Since all possible explanations are stored and not chosen greedily while their construction, the order of traversing the matching edges will not affect the results.

A node's weight $w(n)$ is given by the similarity benefit which is given by the union overlap normalized by the union volume of all corresponding feature objects.

Edges in the independent set graph model contradictions between any two explanations. These arise when two explanations use at least one common feature object either from T_i or T_{i+1}: they provide different explanations for the temporal evolution of the same feature object. We observe that all explanations which result from the same matching edge are conflicting by definition. Thus, these *form a clique* in the independent set graph. In Figure1 b) all nodes which are part of the same clique since they result from the same matching edge are connected by thick edges. Cross-clique connections are indicated by dashed lines. Additionally, all explanations which are part of the same clique are highlighted in the same color.

In practice, we note that the power set can become prohibitively large. We thus restrict the node set's size by keeping only explanations that provide a higher weight than the initial linear assignment. In addition, we introduce a user-defined cutoff, which we typically set to 256 potential explanations per matching edge, a value that we empirically found to be sufficient for all our experiments. For more details, we refer to [35].

A valid solution, i.e. a conflict-free selection of explanations, which ensures the global maximization of feature similarity is equivalent to a maximum-weight independent set on the graph of all explanations. This means a set of mutually non-adjacent nodes with a maximum overall weight corresponding to non-conflicting explanations – splits, merges, and continuations – which describe the relation of feature objects in two subsequent time steps. All feature objects which are not assigned after the second step are assumed to be the result of a birth or death event.

3.2 Greedy Selection Event Detection

We utilize the specific structure of the independent set graph as the basis for a greedy assignment strategy. As stated above, the graph consists of cliques, where each clique is spawned by all explanations which are constructed based on the same matching edge, and additional inter-clique edges. These cliques have a minimum size of two nodes because a clique is just spawned if there is at least one explanation increasing the overall weight compared to the matching. The maximum number of nodes per clique is determined by the cutoff value of 256.

Since all nodes in a clique are connected by an edge and, thus, conflicting we can choose at most one node in each clique for the independent set. Because we want to maximize the overall weight, we aim to choose at least one node per clique. Thus, a valid solution will pick *exactly one* explanation from each clique. Hence, the search space for a solution can be modeled as a search tree over the set of cliques as depicted in Figure 1 c). The trivial choice in each clique is always the node representing the continuation, i.e. only the matching edge, because this node has no edge to any other clique by definition. Thus, we know that there always is a valid choice in each clique. Due to the fact, that each choice may affect other choices in the subsequent search, a solution has to be constructed in an inherently incremental fashion. This leads to a sequential solution process.

We implement this process by a greedy selection strategy. The idea is to iterate over all cliques and choose the best non-conflicting explanation that remains available. Thus, the selection is constructed by a depth-first traversal of the search tree which respects potential cross-clique conflicts. In order to arrive at a good solution, we want to ensure that important choices, i.e. the ones where the difference between the best and the second-best explanation are large, get resolved first. Thus, we sort the list of cliques according to decreasing weight difference between their two highest-weight nodes. Then, we iterate linearly over all cliques. In each step, we pick the highest-weight node that is not incident to a previously picked one. If the graph instance did not have any inter-clique edges, this would lead

Table 1: Summary of data set sizes for the data sets used in our evaluation.

Data Set	Spheres	Voronois	Vortices	Dissipation Elements (small)	Dissipation Elements (large)
Dimensions	256^3	256^3	256^3	256^3	512^3
Size per field[MB]	64	64	64	64	512
Number of time steps T	200	200	2,000	2,000	100
Avg. #features per TS	189	141	1,135	28,979	109,893

to the optimal solution, i.e. the one that features the highest-weight node for each clique. By making the high-difference choices first, we ensure that the penalty for choosing the second-best option steadily decreases.

We note that our selection strategy mimics the behavior of earlier feature tracking algorithms [38] by choosing the best available option in a greedy fashion. We still keep the advantage of considering both merge and split events at the same time, while the approach by Silver et al. always prefers splits over merges due to the implicit order in which these events are analyzed by the algorithm. In more complex cases however, specifically for space-filling features, our approach retains the flexibility to look into other options and thus returns a satisfactory solution whereas the implicit ordering in Silver's method precludes this opportunity as demonstrated in [35].

Eventually, the greedy depth-first search leads to a conflict-free assignment which solves the event detection problem. The complexity of this process is linear in the number of cliques times the maximum size of a clique. The latter is bound by the cutoff value introduced above leading to a linear solution process with respect to the number of features.

The quality of the eventual solution is largely determined by the initial ordering of the cliques. We considered other criteria, including decreasing difference between largest weight and smallest weight in a clique and decreasing standard deviation of the weights within a clique. But experiments quickly revealed that these choices led to uniformly inferior results. Therefore, we do not report detailed results in the next section.

4 Results

In this section we analyze the performance of our approach by applying it to different synthetic and simulation data sets containing both sparse and space-filling features. Additionally, we present a comparison of our approach to the method by Schnorr et al. [35] which uses a 0.99-approximation by an external solver that does not harness the specific problem structure for solving the independent set. First, we test the approach on synthetic data sets which enable a quantitative evaluation due to the availability of ground truth information. Even though synthetic data sets do not fully emulate the behavior of the actual simulation data, they facilitate measuring the approach's performance in a controlled scenario. Second, we test our approach on different simulation data sets and perform a phenomenological analysis by manual visual inspection of the results. The characteristics of the synthetic and the simulation data sets are given in Table 1.

We investigated the tracking results for a large number of features. Afterward, we selected representative cases from this overall set, and only show the actual features participating in the event under consideration. In order to ease the distinction of different features over time they are color coded by randomly assigned colors. If a feature continues, it keeps its color. If a split occurs, only the largest part keeps the color and all other participating structures receive a new randomly assigned color. Similarly, a merged feature receives the color of its largest participant.

In order to compare our results to the method by Schnorr et al. [35], we first compare the results regarding the ground truth information for the synthetic data sets. Second, we compare the runtimes and the weights regarding the independent set problem for both approaches on the synthetic and the simulation data sets. Additionally, we compare the resulting weights to the theoretical maximum which is given by choosing the highest weighted node in each clique without respecting conflicts. We detail these steps in the following sections.

4.1 Synthetic Data Sets

A comprehensive, formal evaluation would require the access to ground truth information. For simulation data sets this is usually not available. In order to enable the quantification of our algorithm's tracking performance, we used two different types of synthetic data sets including all types of event explanations as well as the corresponding ground truth information. The objects and their temporal evolution are generated to cover different aspects of the simulation data sets under consideration.

In order to assess the tracking performance, we tracked both data sets using our approach and the reference algorithm. We then compared the respective results to the ground truth information and the approaches regarding their accuracy w.r.t. the ground truth. In the following sections, the data set generation and the tracking results are discussed in more detail.

4.1.1 Sparse Sphere Data Set

The first synthetic data set represents a sparse setting containing randomly distributed spheres which differ in their diameter. Time variance is simulated by moving each object in the data set independently along a local velocity field of a simulation of homogeneous, isotropic turbulence. Each connected component of mutually intersecting spheres in the data set is considered as a single feature object obtaining a unique ID. A continuation is recorded if a connected component contains the same defining spheres after the temporal displacement. In the case that objects overlap or break apart due to the temporal displacement of their constituent spheres, a merge or split event is stored, respectively. To enable birth and death events, spheres either enter or leave the domain, respectively. Additionally, birth events in the inner part of the domain are are included by searching for empty regions and seeding new small spheres which grow in the following time steps. Analogously, death events are enabled by searching for non-connected spheres whose size is decreased until they disappear. Furthermore, the combination of splits and merges including the same object in one time step is excluded since such a combination does not reflect the evolution in experimental data sets for a sufficiently high temporal resolution. This is done by artificially splitting such components in a subsequent step and assigning different IDs.

4.1.2 Space-Filling Voronoi Data Set

The Voronoi data set is defined by the Voronoi diagram of a randomly distributed 3D point set. Each cell serves as a single feature object. Due to the definition of Voronoi cells, this setting is space-filling. All features are moved across the domain by a single translation vector to simulate time variance. The boundaries are handled periodically such that seed points which leave the domain re-enter on the opposite side.

Additionally, split and merge events are included in this data set. In order to create a split, a single seed point is duplicated on its position. In the following time steps, both seed points are moved

Table 2: Ground truth and tracking results for the sphere and the Voronoi data set regarding the individual events and the sum of all events.

Event Type	Sphere data set				Voronoi data set			
	#Events Ground Truth	#Detected Events	#Correctly Detected	% Correctly Detected	#Events Ground Truth	#Detected Events	#Correctly Detected	% Correctly Detected
Continuation	32,272	32,258	32,186	99.73	27,304	27,336	27,300	99.99
Split	1,744	1,713	1,696	97.25	293	279	275	93.86
Merge	1,926	1,979	1,847	95.90	186	177	177	95.16
Birth	810	919	801	98.89	0	3	0	-
Death	577	717	560	97.05	0	0	0	-
Sum	37,329	37,586	37,090	99.36	27,783	27,795	27,752	99.89

in opposing directions in addition to the overall movement of the whole data set. As the two points diverge, the element is split and the respective event is stored. For a merge event, one randomly selected seed point and its nearest neighbor are moved towards each other and merged to a single point if their distance is below a certain threshold. This movement is independent from the overall movement of the data set; the respective event is stored in the ground truth. Please note that birth and death events are not included in this data set. The rationale behind this exclusion is the absence of background in the data set which would be necessary for birth and death events.

4.1.3 Performance on Synthetic Data Sets

In order to measure our algorithm's performance and to compare it to the reference solution we tracked 200 time steps of both data sets – the sphere data set and the Voronoi data set – with both algorithms. Table 2 shows the results with respect to the ground truth information. Listed are the absolute number of detected events, the number of correctly detected events, and the percentage of the correctly detected events per event type. Additionally, the sum of all events is listed.

Please note that both approaches provide exactly the same results in both cases. Furthermore, not just the raw numbers regarding the ground truth but the complete tracking result is identical. Thus, the resulting values in the table are just listed once in order to show that the greedy solution for the independent set provides correct tracking results for more than 90% for all types of event explanations regarding both data sets. This suggests that our greedy approach is able to provide reliable tracking results for sparse and space-filling data sets.

4.2 Simulation Data Sets

To evaluate our approach on real world data sets, we analyze data sets from turbulence research phenomenologically based on manual visual inspection of the resulting feature paths obtained by our algorithm. To this end, we take different data sets into account which contain sparse and space-filling features. All considered data sets result from a direct numerical simulation of homogeneous, isotropic turbulence inside a periodic box. The first data sets comprises vortices, the two other data sets both contain extracted dissipation elements (DEs) as features while they differ in size and complexity.

The vortices have been extracted by thresholding for high vorticity magnitude resulting in features which are sparse with respect to the overall data domain. We include this case in our evaluation since similar settings have been used in order to evaluate tracking algorithms [28, 31, 38, 39].

The two other data sets containing DEs are included to evaluate the performance of our approach for space-filling features. They directly result from our collaborators' research. DEs describe the geometric structure of small-scale turbulence; their extraction is performed by a validated code provided by our collaborators [3]. Both data sets have a relatively high temporal resolution; every tenth simulation time step has been written to disk. They have a spatial resolution of 256^3 voxels and 512^3 voxels, respectively.

4.2.1 Vortices in Isotropic Turbulence

In our analysis, we investigated the tracking results for a number of vortices regarding their plausibility. This phenomenological assessment is in line with evaluations in previous work [21, 31, 32, 39]. In Figure 2 an exemplary tracking result of our approach for a single vortex is shown. For the sake of clarity, we additionally included the corresponding tracking graph. The vortex splits into two smaller vortices in the third time step – a larger pink one and a smaller orange one. These two vortices move independently over the next few time steps until the smaller one splits again in the last time step. Similar settings and evolutions have been evaluated in this context. In summary, based on the inspected tracking results, we found that our algorithm is able to provide plausible results for this data set.

4.2.2 Dissipation Elements

Analogously to the vortex data set, we inspected several evolutions in the two DE data sets for their plausibility. Figure 3 illustrates the exemplary evolution of a DE in the 256^3 case involving one merge and two split events and the corresponding tracking graph. First, the feature splits in the second time step into the larger orange component and the smaller yellow one. These features do not move apart due to the space-filling setting. Second, these two features merge again in the fifth time step resulting in the large orange DE. Finally, a smaller green component splits apart from the large orange feature in the sixth time step. The manually inspected results are comparable to the results by Schnorr et al. [35] and backed by our collaborators. In conclusion, we argue that our greedy approach also provides plausible results in case of a space-filling setting.

4.3 Comparison Regarding the Independent Set

As mentioned in the introduction we compare the event detection results and the runtime performance of our greedy assignment strategy to our earlier approach which uses the *CBC* solver of the *COIN-OR* project. To this end all available time steps of all data sets have been tracked and their runtimes regarding the independent set graph setup as well as the runtimes for solving the independent set problem have been measured. Additionally, we reported the resulting weight for the independent set w.r.t. both algorithms as well as the theoretical maximum given by the sum of the maximum values in each clique. All measurements have been performed on a dual socket server featuring two Intel® XEON® E5-2695 v3 CPUs (28 cores @ 2.3 GHz each) and 512 GB of RAM.

Figure 4 summarizes the results of our measurements. Each row represents the results for one of the data sets – spheres, voronois, vortices, and DEs (256^3 and 512^3). The column on the left shows a box-plot for the resulting independent set weights. For both approaches all time steps are inspected and the results are normalized by the theoretical maximum. Regarding the synthetic data sets we see equivalent values for both approaches which is backed by the comparison w.r.t. the ground truth information. For the vortex data set, we see similar results as for the synthetic data set. In case of the two DE data sets, the median of the normalized quality is slightly higher for the approach using the *CBC* solver. However, the results

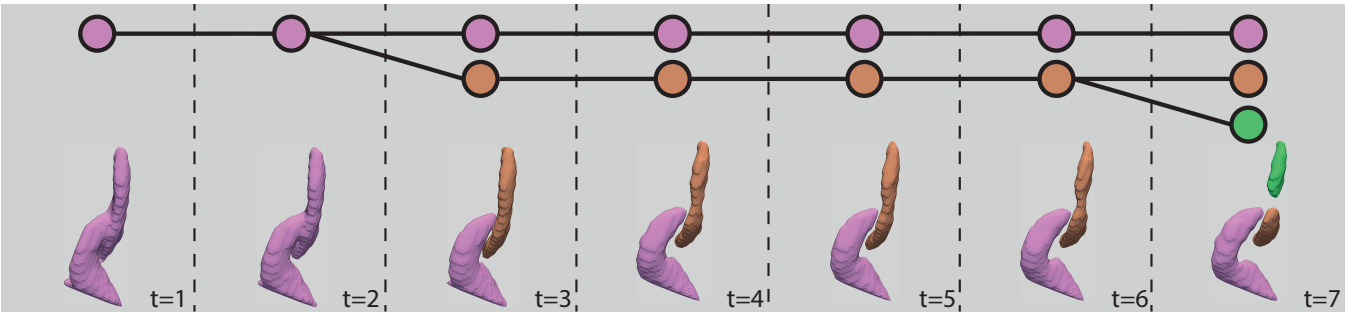

Figure 2: Depiction of the evolution of a vortex (bottom row) including the corresponding tracking graph (top row). The selected vortex splits into two components in the third time step. Additionally, the smaller component splits again in the seventh time step.

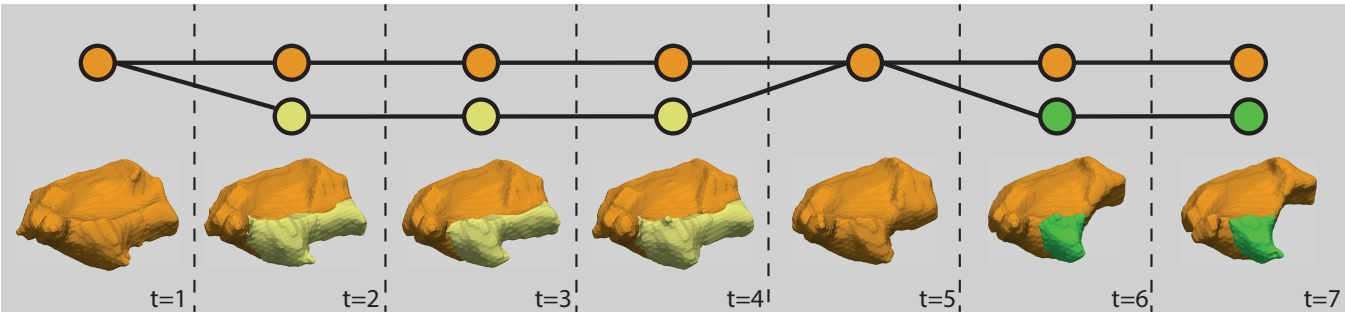

Figure 3: Illustration of the evolution of a DE and the corresponding tracking graph. The DE is involved in several events. First a smaller component splits apart in the second time step, Second, the component merges again with the larger one in the fifth time step. Finally, in the sixth time step a smaller part of the DE breaks apart again.

are comparable since the values for the greedy strategy are also above 99% of the theoretical maximum which was the bound used for the approximate solution by the solver.

In addition to this comparison regarding the weights, we inspected several feature evolutions based on differences between the two approaches. One of these differences is depicted in Figure 5. The images show the features within the former time step, the overlap of these features with the feature in the latter time step which is assigned to two different events, and the different evolutions presented by the two approaches. The overlap for the orange feature with the two other features in the former time step seems similar by manual visual inspection. Thus, the assignment of the orange feature is ambiguous. This setting is demonstrative for a number of similar situations we investigated. For all these cases, the assignment solely based on the two time steps is ambiguous by manual inspection. One option to solve such problems is an assignment which is additionally weighted based on the path length of the corresponding feature as proposed by Reinders et al. [28]. However, this additional weighting might have implications on other parts of the solution and is, hence, an idea we plan to investigate in future work.

In addition to the weights of the independent set, we investigated the runtimes of both approaches. The middle and the right column in Figure 4 show the runtimes for both approaches for all data sets for the first 100 or 50 time steps, respectively. The middle column compares the runtimes for the graph setup, while the right column shows the comparison for the runtimes for solving the independent set problem. Furthermore, Table 3 lists the average runtimes over all time steps for the setup and the solution for both algorithms as well as the corresponding speedup. For both time measurements we see a speedup regarding all data sets – in case of the synthetic data sets the setup decreases to 0 on average. Especially, the speedup regarding the solution of the independent set is mentionable. Moreover, the graphs in Figure 4 reveal that apart from single outliers the runtime for solving the independent set seems to be more stable in contrast to the solver. This observation is justified by the linear complexity which was mentioned in Section 3.

Beyond being uniformly faster across all data sets, there is another interesting observation to be gleaned from the performance data: the speedup decreases with increasing complexity. We offer the following explanation. For simple cases, specifically sparse features which lead to very few cross-clique connections, the greedy selection encounters little choice: the highest-weight feature in each clique will almost always be pickable and the optimization degenerates into a linear path traversal. This is naturally very fast. In contrast, the general COIN solver cannot exploit the specific problem structure and thus has to run a full optimization process, which is comparably costly. The gap closes as problem size and complexity increase; the relative speedup of our solution decreases. This is evident for both the synthetic data (spheres vs. voronoi) and the simulation data sets (vortices vs. dissipation elements). In case of the synthetic data set the speedup decreases from 276.5 to 210.0; the speedup of 6112.0 regarding the vortex data set decreases to 96.52 and 33.34 for the 256^3 and the 512^3 DE case, respectively (cf. Table 3). The space-filling cases will feature frequent inter-clique edges and thus the greedy assignment will have to execute more comparisons and will have to frequently select nodes other than the highest-weight one. This process is costly, largely because in involves a re-travesal into already seen parts of the independent set graph. In comparison, the solution stratgey of the COIN solver is not susceptible to this phenomenon. Thus, our algorithm still finds a comparable solution much faster, yet the gap becomes smaller for more complex problems.

In summary, the measurements demonstrate that the results provided by our greedy assignment strategy are comparable to the 0.99%-approximation of the optimal solution provided by an external solver while the runtime is decreased substantially.

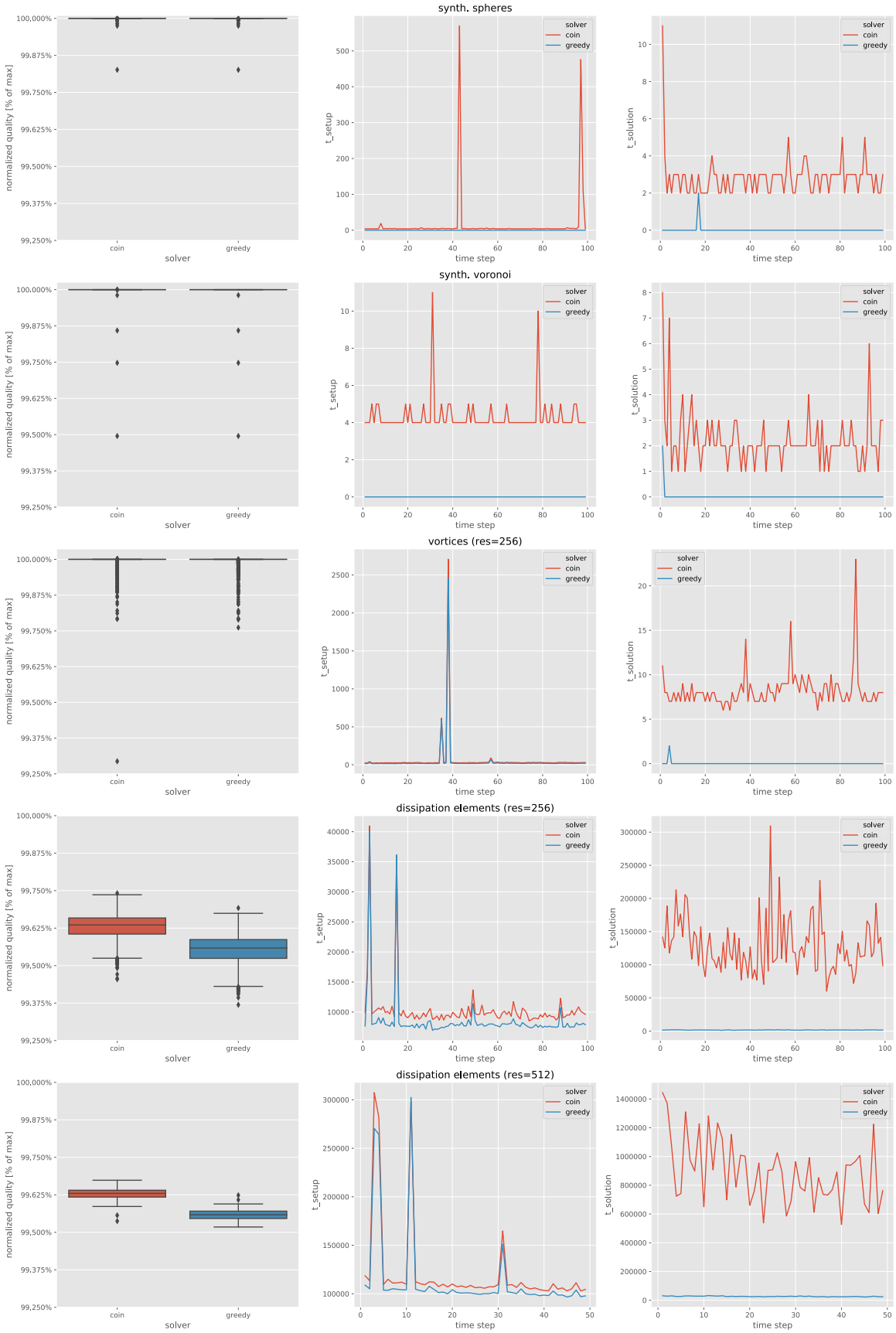

Figure 4: Summary of the measurements regarding our greedy approach and the *CBC* solver. Each row depicts the results for one of the data sets. The graphs show a box-plot for the independent set weights normalized by the theoretical maximum (left), the distribution of the runtimes regarding the setup for the independent set graph (middle), and the distribution of the runtimes w.r.t. the solution of the independent set (right). For the sake of readability, the graphs comparing the runtimes only include the first 100 and 50 time steps, respectively.

Table 3: Summary of the average runtimes reagrding the independent set graph setup and the solution for the greedy approach and the reference approach using the *CBC* solver.

Data set	Avg. setup COIN [ms]	Avg. setup greedy [ms]	Avg. speedup	Avg. solution COIN [ms]	Avg. solution greedy [ms]	Avg. speedup
synth. spheres	12.774	0.000	—	2.779	0.010	276.50
synth. Voronois	8.166	0.000	—	2.111	0.010	210.00
vortices (256^3)	234.604	199.917	1.17	6.115	0.001	6112.00
diss. elements (256^3)	9799.532	7949.669	1.23	88678.282	918.759	96.52
diss. elements (512^3)	121018.959	113456.714	1.07	895430.714	26856.694	33.34

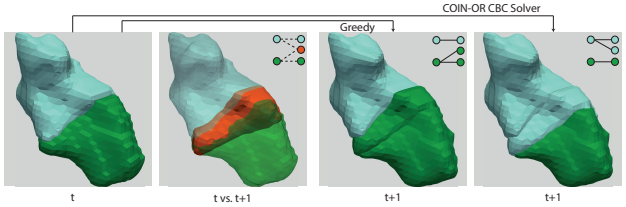

Figure 5: Depiction of different assignments detected by our approach and the reference solution. The images show the former time step (left), the overlap of the differently assigned (orange) feature with the features in the former time step (middle left), and the resulting assignments found by our greedy approach (middle right) and the *CBC* solver (right).

5 Discussion

The main goal of the method proposed in this paper has been to exploit the specific structure of the independent set graph in order to accelerate the event detection stage of feature tracking while maintaining good-quality results. In summary, our results demonstrate that the approach provides reliable results which are comparable to those found by a state-of-the-art optimization tool. In this section, we discuss our approach with regard to design decisions, known limitations, and more general implications.

This work was mainly motivated by the idea to increase the performance of the workflow outlined in [35]. Thus, we originally set out to devise a parallel algorithm for feature tracking. In order to do so, we focused on the stage that comprised the largest part of the computation: the event detection. The initial analysis of the search space suggested that the resulting search tree would be amenable to a parallel branch and bound strategy. To this end, we implemented a *task-based strategy* which traverses the search tree while simultaneously pruning suboptimal paths w.r.t. the current best solution. This scheme had to construct the search tree incrementally because a full, up-front construction would lead to a prohibitively large structure. In addition, a branch and bound scheme needs a quick first guess at a good solution in order to have bound to prune against. Our experiments revealed that this first bound – chosen to be a greedily selected, valid path in the search tree – was almost always very close to the achievable optimum. Hence, a sequential depth-first search showed very good results in practice; our parallel branch-and-bound algorithm did not significantly improve on the results while still consuming more compute time. Thus, we currently resort to the sequential greedy solution. Despite this finding, the matching algorithm and the setup for the independent set graph – which now become the most time consuming parts of the overall computation – do not yet run in parallel. We plan to address these in order to scale the tracking to larger cases.

One important aspect for feature tracking in general is the chosen similarity criterion. Currently, we use the normalized overlap in both phases of our tracking algorithm: we compute the intersection of the two feature objects in question and normalize it by the volume of the larger feature. This may result in a bias towards larger feature objects. However, our approach is oblivious to a switch of the applied similarity metric. As mentioned in Section 4.3, we found several cases where the two approaches for computing the independent set provided different explanations which are ambiguous based on manual visual inspection solely based on the respective time steps. We thus plan to investigate other similarity metrics, e.g. abstract attribute sets as proposed by Reinders et al. [28] or particle data as used by Sauer et al. [32], in order to resolve these issues.

Changing the similarity metric might have effects on the search space of the independent set problem. Thus, we plan to investigate different ordering strategies for the node traversal in our search tree in the context of different similarity metrics. We already considered different strategies for the current version of our approach and initial experiments revealed that ordering the cliques by decreasing weight difference between the two highest-weight nodes led to the best results. However, with a switch of the similarity criterion, this choice might have to be re-considered.

Finally, the cumulative results of our recent paper [35] and the above analysis give rise to a more general suggestion w.r.t. feature tracking. We initially started our research by arguing that feature tracking becomes significantly more challenging in a dense, i.e. space-filling setting. Indeed, we observe that sparse data sets result in smaller cliques than their space-filling counterparts. In addition, sparse data sets lead to fewer cross-clique connections. In terms of our model, this leads to the observation that the independent set graphs of sparse problem instances *are* less complicated to solve than those for space-filling data. Therefore, we did find significant advantages of the original two-stage optimization algorithm over a reference tracking algorithm for sparse features. However, the current study add more detail to the discussion. Finding an optimal assignment remains an NP-hard problem in general, regardless of sparse or dense input data. However, our results suggest that real world problem instances for the tracking problem usually are far from worst case scenarios. Specifically, in a sparse feature setting the search for a conflict-free, similarity-maximizing solution will often be straightforward. Thus, greedy strategies such as the one proposed by Silver et al. [38] or the one discussed above will lead to reasonably good results. For dense settings the problem becomes more complex. Here, a straightforward greedy approach is susceptible to ordering artifacts, as shown in [35]. However, our greedy search, which relies on a limited view of the overall search space, still provides results on par with state of the art optimization packages as shown in this paper. This ability to abide by a greedy method avoids the need for a general, uninformed optimization run and thus enables us to provide good solutions at a fraction of the runtime cost.

6 Conclusion

In this work, we have presented an optimized event detection method for feature tracking which exploits the specific structure of the independent set graph containing all possible event explanations. Our approach is based on the idea to align the optimization process for the independent set along the cliques which are spawned by the mutually contradicting event explanations in order to create a greedy assignment strategy. To demonstrate this capability, we applied the

approach to a variety of synthetic and simulation data sets. In addition to a comparison with the ground truth result for the synthetic data sets and a manual visual inspection backed by our collaborators for the simulation data sets, we compared the results and runtimes for the independent set problem to the approach by Schnorr et al. [35].

Several aspects for future work have been discussed in the previous section. In particular, we plan to investigate additional similarity metrics in order to reliably solve ambiguous assignments in the event detection. In addition, we plan to implement and assess other ordering criteria to align the optimization process in order to increase the overall weight of the independent set based on the chosen similarity metric.

In summary, the event detection method proposed in this work provides reliable results which are comparable to the 99%-approximation of the theoretical optimum provided by an external solver while the overall runtime is significantly decreased.

Acknowledgments

This work received support from the DFG under KU 1132/10-1, HE 6406/1-1, and the German Excellence Initiative via JARA–HPC. In addition, we thank our collaborators from the Institute for Combustion Technology, RWTH Aachen University, particularly Dominik Denker for their valuable input to this work.

References

[1] CBC: An LP-based branch-and-cut library. https://github.com/coin-or/Cbc, last access: 2019-08-16.

[2] D. C. Banks and B. A. Singer. Vortex Tubes in Turbulent Flows: Identification, Representation, Reconstruction. In *Proceedings of IEEE Visualization*, pp. 132–139, 1994.

[3] N. Berr, D. Schmidl, J. H. Göbbert, S. Lankes, D. an Mey, T. Bemmerl, and C. H. Bischof. Trajectory-Search on ScaleMP's vSMP Architecture. In *PARCO*, pp. 227–234, 2011.

[4] P.-T. Bremer, G. H. Weber, V. Pascucci, M. Day, and J. B. Bell. Analyzing and Tracking Burning Structures in Lean Premixed Hydrogen Flames. *IEEE Transactions on Visualization and Computer Graphics*, 16(2):248–260, 2010.

[5] J. Chen, Y. Kusurkar, and D. E. Silver. Distributed Feature Extraction. In *Proceedings of SPIE 4665 Visualization and Data Analysis*, vol. 4665, pp. 189–195, 2002.

[6] J. Chen, D. Silver, and L. Jiang. The Feature Tree: Visualizing Feature Tracking in Distributed AMR Datasets. In *Proceedings of the IEEE Symposium on Parallel and Large Data Visualization and Graphics*, pp. 103–110, 2003.

[7] J. Clyne and P. M. A. Norton. Physically-Based Feature Tracking for CFD Data. *IEEE Transactions on Visualization and Computer Graphics*, 19(6):1020–1033, 2013.

[8] H. Doraiswamy, V. Natarajan, and R. S. Nanjundiah. An Exploration Framework to Identify and Track Movement of Cloud Systems. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2896–2905, 2013.

[9] C. Garth, X. Tricoche, and G. Scheuermann. Tracking of Vector Field Singularities in Unstructured 3D Time-Dependent Datasets. In *Proceedings of IEEE Visualization*, pp. 329–336, 2004.

[10] A. Globus, C. Levit, and T. Lasinki. A Tool for Visualizing the Topology of Three-Dimensional Vector Fields. In *Proceedings of IEEE Visualization '91*, pp. 33–40, 1991.

[11] E. J. Griffith, F. H. Post, M. Koutek, T. Heus, and H. J. J. Jonker. Feature Tracking in VR for Cumulus Cloud Life-Cycle Studies. In *Proceedings of the EG Workshop on Virtual Environments (EGVE)*, pp. 121–128, 2005.

[12] A. Gyulassy, P.-T. Bremer, B. Hamann, and V. Pascucci. A Practical Approach to Morse-Smale Complex Computation: Scalability and Generality. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1619–1626, 2008.

[13] A. Gyulassy, N. Kotava, M. Kim, C. D. Hansen, H. Hagen, and V. Pascucci. Direct Feature Visualization Using Morse-Smale Complexes. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1549–1562, 2012.

[14] J. L. Helman and L. Hesselink. Visualizing Vector Field Topology in Fluid Flows. *IEEE Computer Graphics and Applications*, 11(3):36–46, 1991.

[15] J. Jeong and F. Hussain. On The Identification of a Vortex. *Journal of Fluid Mechanics*, 285:69–94, 1995.

[16] M. Jiang, R. Machiraju, and D. Thompson. Detection and Visualization of Vortices. In C. R. Johnson and C. D. Hansen, eds., *Visualization Handbook*, pp. 295–309. 2004.

[17] C. R. Johnson. Top Scientific Visualization Research Problems. *IEEE Computer Graphics and Applications*, 24(4):13–17, 2004.

[18] R. M. Karp. Reducibility among Combinatorial Problems. In R. E. Miller, J. W. Thatcher, and J. D. Bohlinger, eds., *Complexity of Computer Computations*, pp. 85–103. 1972.

[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 Transactions on Visualization and Computer Graphics*, 12(5):1053–1060, 2006.

[20] K.-L. Ma, J. V. Rosendale, and W. Vermeer. 3D Shock Wave Visualization on Unstructured Grids. In *Proceedings of the 1996 Symposium on Volume Visualization*, pp. 87–94, 1996.

[21] C. Muelder and K.-L. Ma. Interactive Feature Extraction and Tracking by Utilizing Region Coherency. In *Proceedings of IEEE Pacific Visualization*, pp. 17–24, 2009.

[22] S. Ozer, D. E. Silver, K. Bemis, and P. Martin. Activity Detection in Scientific Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):377–390, 2014.

[23] S. Ozer, J. Wei, D. E. Silver, K.-L. Ma, and P. Martin. Group Dynamics in Scientific Visualization. In *Proceedings of the IEEE Symposium on Large Data Analysis and Visualization*, pp. 97–104, 2012.

[24] H.-G. Pagendarm and B. Seitz. An Algorithm for Detection and Visualization of Discontinuities in Scientific Data Fields Applied to Flow Data with Shock Waves. In P. Palamidese, ed., *Scientific Visualization: Advanced Software Techniques*, pp. 161–177. 1993.

[25] H.-G. Pagendarm and B. Walter. Feature Detection from Vector Quantities in a Numerically Simulated Hypersonic Flow Field in Combination with Experimental Flow Visualization. In *Proceedings of IEEE Visualization1994*, pp. 117–123, CP12, 1994.

[26] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice Hall, 1982.

[27] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. The State of the Art in Flow Visualisation: Feature Extraction and Tracking. *Computer Graphics Forum*, 22(4):775–792, 2003.

[28] F. Reinders, F. H. Post, and H. J. Spoelder. Visualization of Time-Dependent Data with Feature Tracking and Event Detection. *The Visual Computer*, 17(1):55–71, 2001.

[29] J. Reininghaus, J. Kasten, T. Weinkauf, and I. Hotz. Efficient computation of combinatorial feature flow fields. *IEEE Transactions on Visualization and Computer Graphics*, 18:1563–1573, 2011.

[30] H. Saikia and T. Weinkauf. Global Feature Tracking and Similarity Estimation in Time-Dependent Scalar Fields. *Computer Graphics Forum*, 2017.

[31] R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing Features and Tracking Their Evolution. *IEEE Computer*, 27(7):20–27, 1994.

[32] F. Sauer, H. Yu, and K.-L. Ma. Trajectory-Based Flow Feature Tracking in Joint Particle/Volume Datasets. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2565–2574, 2014.

[33] L. Schäfer, J. H. Göbbert, and W. Schröder. Dissipation element analysis in experimental and numerical shear flow. *European Journal of Mechanics-B/Fluids*, 38:85–92, 2013.

[34] G. Scheuermann and X. Tricoche. Topological Methods for Flow Visualization. In C. R. Johnson and C. D. Hansen, eds., *Visualization Handbook*, pp. 341–356. 2004.

[35] A. Schnorr, D. N. Helmrich, D. Denker, T. W. Kuhlen, and B. Hentschel. Feature Tracking by Two-Step Optimization. *IEEE Transactions on Visualization and Computer Graphics*, preprint available online, 2018.

[36] I. K. Sethi, N. V. Patel, and J. H. Yoo. A General Approach for Token Correspondence. *Pattern Recognition*, 27(12):1775–1786, 1994.

[37] D. Silver. Object-Oriented Visualization. *IEEE Computer Graphics*

and Applications, 15(3):54–62, 1995.

[38] D. Silver and X. Wang. Volume Tracking. In *Proceedings of IEEE Visualization*, pp. 157 – 164, 1996.

[39] D. Silver and X. Wang. Tracking and Visualizing Turbulent 3D Features. *IEEE Transactions on Visualization and Computer Graphics*, 3(2):129–141, 1997.

[40] D. Silver and X. Wang. Tracking Scalar Features in Unstructured Data Sets. In *Proceedings of IEEE Visualization'98*, pp. 79–86, 1998.

[41] S. Stegmaier, U. Rist, and T. Ertl. Opening the Can of Worms: An Exploration Tool for Vortical Flows. In *Proceedings of IEEE Visualization'05*, pp. 463–470, 2005.

[42] H. Theisel and H.-P. Seidel. Feature Flow Fields. In *Proceedings of the Joint EG/IEEE TCVG Symposium on Visualization*, pp. 141–148, 2003.

[43] T. van Walsum, F. H. Post, D. Silver, and F. J. Post. Feature Extraction and Iconic Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2(2), 1996.

[44] L. Wang and N. Peters. The Length-Scale Distribution Function of the Distance Between Extremal Points in Passive Scalar Turbulence. *Journal of Fluid Mechanics*, 554:457–475, 2006.

[45] T. Weinkauf, J. Sahner, H. Theisel, and H.-C. Hege. Cores of Swirling Particle Motion in Unsteady Flows. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1759–1766, 2007.

[46] T. Weinkauf, H. Theisel, A. V. Gelder, and A. Pang. Stable Feature Flow Fields. *IEEE Transactions on Visualization and Computer Graphics*, 17(6):770–780, 2011.