Multi-field Pattern Matching based on Sparse Feature Sampling

Zhongjie Wang, Hans-Peter Seidel, and Tino Weinkauf

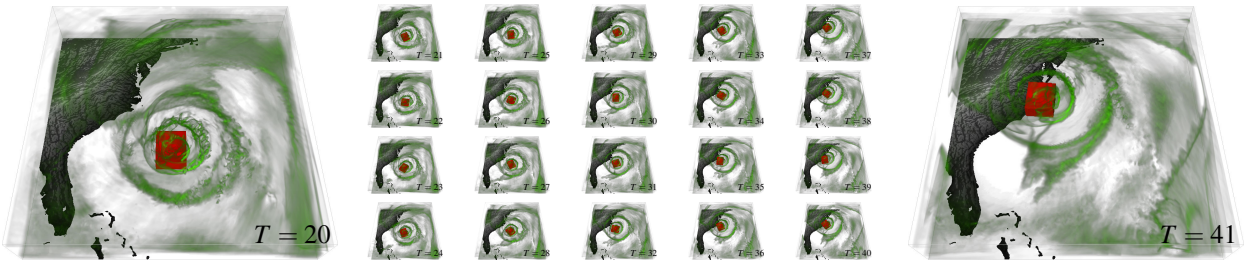


Figure 1. Our method allows pattern matching in multi-fields such as the *Hurricane Isabel* data set. The user selects the eye of the hurricane at $T = 20$ using a red box. This is the input pattern. Our algorithm uses the 3D SIFT features of 11 scalar fields simultaneously to find matching patterns in the following time steps. This amounts to a tracking of the eye of the hurricane.

Abstract—We present an approach to pattern matching in 3D multi-field scalar data. Existing pattern matching algorithms work on single scalar or vector fields only, yet many numerical simulations output multi-field data where only a joint analysis of multiple fields describes the underlying phenomenon fully. Our method takes this into account by bundling information from multiple fields into the description of a pattern. First, we extract a sparse set of features for each 3D scalar field using the 3D SIFT algorithm (Scale-Invariant Feature Transform). This allows for a memory-saving description of prominent features in the data with invariance to translation, rotation, and scaling. Second, the user defines a pattern as a set of SIFT features in multiple fields by e.g. brushing a region of interest. Third, we locate and rank matching patterns in the entire data set. Experiments show that our algorithm is efficient in terms of required memory and computational efforts.

Index Terms—Pattern matching, multi-field visualization.

1 Introduction

Pattern matching algorithms have proven useful for scalar [10, 12, 22] and vector fields [3, 9, 13, 33]. The general idea is to compute the similarity between a user-supplied pattern and every location in the data set. A pattern can be given in different forms such as a small subset of the domain or a selection of stream lines. A desired property for pattern matching is invariance to translation, rotation, and scaling, i.e., a translated, rotated and scaled copy of the pattern can be found in the data despite these transformations. To achieve this, some existing algorithms require substantial computation time. Furthermore, existing algorithms work for a single field only.

To the best of our knowledge, we present the first pattern matching method in the context of multi-field visualization. Our approach offers fast responses to the user by performing a large part of the pattern search using a sparse set of feature points. Features are computed using the 3D SIFT algorithm [7, 29]. A 3D SIFT feature is located at a point and describes the behavior of the scalar field in its neighborhood with invariance to translation, rotation, and scaling. We compute SIFT features for scalar fields. We deal with vector fields indirectly by means of computing SIFT features for derived scalar fields such as the vorticity magnitude of a flow. Scalar fields for which we compute SIFT features are called *trait fields*.

Pattern matching using our method works as follows: a user selects a search pattern as a region of interest in the domain (a small box). For each SIFT feature within this region, we find similar ones in the entire data set. This is a very fast procedure and yields candidate transformations of the search pattern. In other words, instead of testing the search pattern against *every* other location in the domain (as well as all possible rotations and scale factors), we test it only against a sparse and sufficient set of candidates. The actual similarity value is then computed using a weighted L2-norm over all considered trait fields. The final result is a scalar field that indicates regions in the multi-field data set where all trait fields show a similar behavior as in the user-selected region.

We give the following technical contributions:

- We introduce a novel pattern matching method for 3D multi-field data sets, which bundles the information from different fields into the description of a pattern.
- We achieve response times for pattern matching of less than a second even for large data sets by working with a sparse set of prominent features.
- In contrast to the two previous versions of the 3D SIFT algorithm [7, 29], we achieve full invariance to 3D rotation by finding a robust local coordinate system. This increases the accuracy of pattern matching.

The next section provides an overview of related work. Section 3 recapitulates the 3D SIFT algorithm and explains our improvement to it. Section 4 presents our pattern search algorithm for multi-fields. We evaluate and discuss our method in Section 5. Results are shown in Section 6.

2 Related Work

Revealing the complexity of a multi-field data set is a challenging task. Showing all fields within the same visualization leads to massive occlusions, while individual visualizations fail to communicate the commonalities. Several approaches have been proposed to deal with

- *Zhongjie Wang and Hans-Peter Seidel are with MPI for Informatics, Saarbrücken, Germany. E-mail: {zwang, hpseidel}@mpi-inf.mpg.de.*
- *Tino Weinkauf is with KTH Royal Institute of Technology, Stockholm, Sweden. E-mail: weinkauf@kth.se.*

Manuscript received 31 Mar. 2015; accepted 1 Aug. 2015; date of publication xx Aug. 2015; date of current version 25 Oct. 2015.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

these issues such as multiple coordinated views [8], correlation analysis [27], or the rather recent topological approaches of Joint Contour Nets [6] and Pareto Optimality [14]. In all these approaches, finding similarities between different parts of the data is essentially a manual process.

Pattern matching automates the process of finding similarities in a data set. *Region-based* algorithms define a pattern as a compact spatial region and aim at finding similar regions within a scalar or vector field. The challenges are two-fold. First, it is necessary to detect a similar region even if it is a translated, scaled and rotated version of the pattern. Invariance to other transformations, such as small distortions or brightness changes, is desirable. Second, the computation times are often very long, which hinders interactive approaches. Ebling et al. [9] and Heiberg et al. [13] independently introduced pattern matching for vector fields to the visualization community. Both approaches use a convolution to compute the similarity between a pattern and a location in the field. This requires to sample all possible rotations, translations and scales, and typically leads to high running times. Moment-invariant descriptors are used by Schlemmer et al. [28] and Bujack et al. [3] to achieve pattern invariance with respect to translation, rotation, and scaling. These approaches are fast, but treat 2D vector fields only.

Feature-based approaches aim at finding similar structures by comparing features with each other. Several approaches use topology to compare structures in scalar fields. Thomas et al. [31, 32] concentrate on finding all symmetric structures in a scalar field using the contour tree. Saikia et al. [26] perform a similarity search for any structurally significant region as given by a subtree in the merge tree.

A number of methods for vector fields deal with the comparison of stream lines. Li et al. [19] uses the bag-of-features approach to describe the characteristics of stream lines. This leads to a clustering of line fields. Several approaches [17, 25] use the spatial distance between pairwise closest points as the similarity measure for clustering stream lines. Wang et al. [33] define patterns using segmented stream lines and compute the similarity of a pattern to the entirety of the data set. Tao et al. [30] extract shape invariant features and compare them using a string-based algorithm.

Pattern matching for scalar fields has mainly be developed in the computer vision community and found a large number of applications. One of the most successful approaches is *SIFT*, which stands for *Scale-Invariant Feature Transform*. It was introduced by David Lowe [20] in 1999 and refined in 2004 [22] to describe local feature points in photos (2D images) in a translation-, rotation-, and scale-invariant manner. Since then, it has been applied to a number of domains including image registration [36], object recognition [21], image stitching [2], and video tracking [1].

Generalizations to higher dimensions have been proposed by Scovanner et al. [29] and Cheung et al. [7] independently. The former provided a 3D version of the SIFT algorithm, the latter a generalization to n-dimensional domains. For 3D domains, they both boil down to the same algorithm. As we will show in the next section, these existing approaches fail to capture rotation invariance in the sense that they only determine one axis of the 3D local coordinate system. The results are sensitive to rotations around this axis.

Our algorithm is a hybrid method combining aspects of feature-based and region-based pattern matching. We apply feature-based matching using SIFT to obtain a set of candidate patterns, which are then confirmed using a region-based approach. This yields fast computation times, dense results and a high accuracy at the same time.

3 Scale-Invariant Feature Transform

In the following, we recapitulate the basics of the 3D SIFT algorithm based on the work of Scovanner et al. [29] and Cheung et al. [7] as well as the original 2D work of Lowe [22]. Especially the original work comes with a lot of background information both on a theoretical and practical level, which serves as justification for the individual steps of the algorithm and for the parameter choices. In this paper, we follow these choices and refer the interested reader to [22] for more background information.

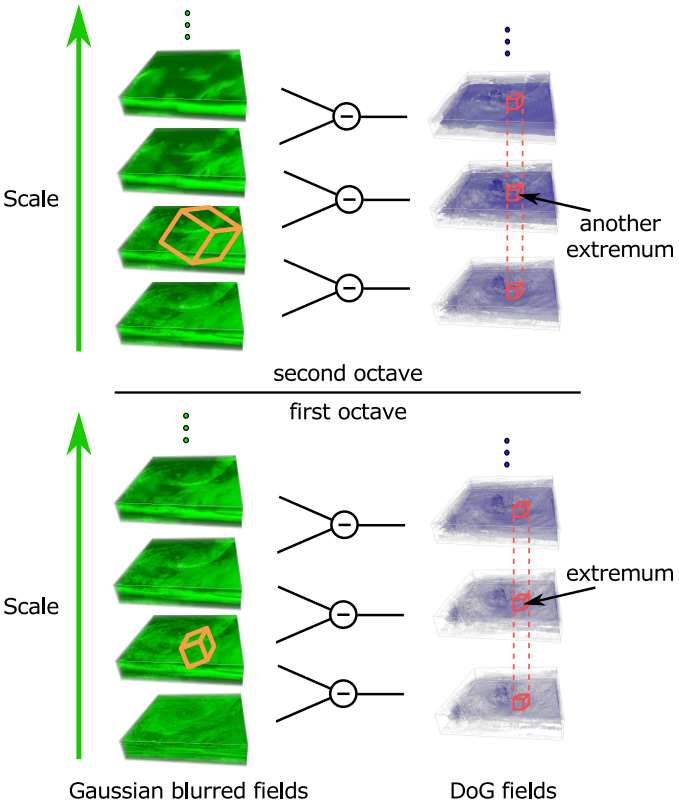


Figure 2. Localization of SIFT features in different scales. A sequence of Gaussian blurred fields with increasing σ is generated. DoG (Difference-of-Gaussian) fields are computed by subtracting neighboring blurred fields. The extrema in the DoG fields denote the SIFT feature locations. For a given location, the neighborhood in the blurred field of the corresponding scale (orange box) serves as a description of the SIFT feature (see also Figure 3).

3.1 Background on 3D SIFT

There are two aspects about a SIFT feature: its *location* and its *description*. The computation of these aspects is related, as we shall see in the following.

Consider a 3D scalar field $S(x,y,z)$ on a 3D uniform grid. The scale space of S is constructed using a convolution with a Gaussian blur $G(x,y,z,\sigma)$

$$L(x,y,z,\sigma) = G(x,y,z,\sigma) * S(x,y,z) \tag{1}$$

where σ refers to the standard deviation of the Gaussian distribution. Note that σ spans the new scale-dimension and larger σ correspond to blurrier versions of the scalar field. The location of the SIFT features is computed from a derived space, namely the convolution of the scalar field S with the *Difference-of-Gaussian* function

$$D(x,y,z,\sigma) = (G(x,y,z,k\sigma) - G(x,y,z,\sigma)) * S(x,y,z), \tag{2}$$

where $k > 1$. This can be computed as the difference between two neighboring scales

$$D(x,y,z,\sigma) = L(x,y,z,k\sigma) - L(x,y,z,\sigma). \tag{3}$$

The locations of the SIFT features are the extrema of $D(x,y,z,\sigma)$.

In practice, the fields are constructed as shown in Figure 2. One starts with an initial blur; Lowe [22] suggests $\sigma_0 = 1.6$. The following scales are computed using a repeated convolution with $G(x,y,z,k)$. The value of k is set such that a doubling of σ is achieved after a certain number s of steps. Doubling σ is referred to as an *octave*. Lowe [22] suggests $k = 2^{1/s}$ with $s = 3$. The Difference-of-Gaussian (DoG) fields

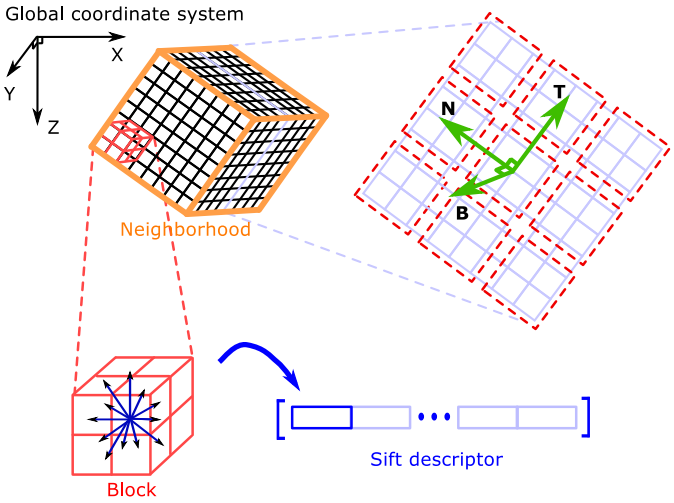


Figure 3. Computation of the SIFT descriptor. Based on a proper orientation (green arrows), a neighborhood of size $9 \times 9 \times 9$ is sampled in the blurred scalar field (see also Figure 2). It is split into 27 blocks. For each block, a 3D histogram of the gradient is computed and stored in the SIFT descriptor.

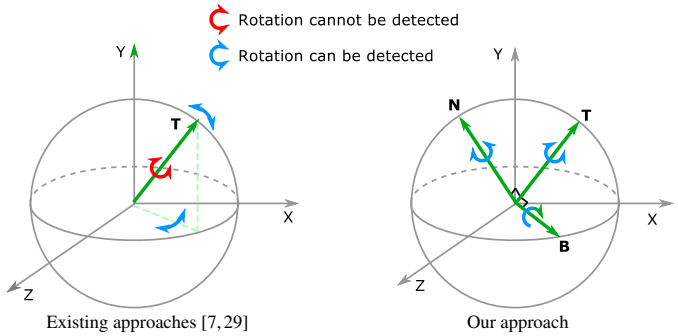


Figure 4. Rotation invariance of the SIFT descriptor can only be achieved, if the 3D orientation of the neighborhood is properly fixed. See also Figure 3. Previous approaches attempt this using a single vector encoded in spherical coordinates. The result is sensitive to rotation *around* this vector itself. We compute a stable local coordinate system using (10), which makes the 3D SIFT descriptor properly rotation invariant.

are computed from two neighboring scales as shown in Figure 2. Note that the total number of blurred scalar fields needs to be $s+3$ such that the SIFT feature locations can later be computed from a full octave. The next octave starts with $2\sigma_0$ and a halved resolution. We use a total of three octaves throughout the paper.

The *locations* of the SIFT features are found as extrema in the DoG fields. More precisely, a grid vertex in the DoG field is marked as an extremum if all its grid neighbors in the previous, current and next scale are smaller/larger, i.e., a $3 \times 3 \times 3 \times 3$ neighborhood is checked. See the right part of Figure 2 for an illustration. It does not matter whether an extremum is a minimum or a maximum.

The *description* of a SIFT feature is derived from a neighborhood in the blurred scalar fields. This is illustrated in Figure 2 as orange boxes over the blurred fields. Note that the effective physical size of these boxes is larger in higher octaves due to the changed resolution.

The computation of the SIFT descriptor is illustrated in Figure 3. The key here is to achieve rotation invariance by choosing a consistent orientation for the neighborhood box based on local properties of the data. For 2D SIFT features, Lowe [22] uses the local image gradient, i.e., a 2D vector, which uniquely defines the orientation of a 2D neighborhood. For the 3D case, the existing methods by Scovanner et al. [29] and Cheung et al. [7] use the 3D gradient vector as orientation. More precisely, they parameterize it to the spherical coordinate system and compute a 3D rotation matrix from that. We will show in the next section that this does *not* define the orientation of a 3D neighborhood in a unique manner.

However, after assigning an orientation, a neighborhood with size $9 \times 9 \times 9$ is considered around the extrema position. It is split into 27 small blocks with size $3 \times 3 \times 3$ each. For each block, the gradient of the blurred scalar field is sampled at the vertices and recorded in a 3D orientation histogram. It has 12 bins as defined by a Platonic icosahedron. The histogram records the magnitude of the gradient. Each histogram comprises a part of the SIFT descriptor. The complete SIFT descriptor has $12 \times 27 = 324$ elements. The descriptor is considered a vector and normalized to have unit length. The process of computing a SIFT descriptor is shown in Figure 3.

Euclidean distance is used to measure the cost (dissimilarity) between two SIFT descriptors $\mathbf{k}$ and $\mathbf{m}$:

$$\text{cost}(\mathbf{k},\mathbf{m}) = ||\mathbf{k}-\mathbf{m}||. \tag{4}$$

This direct and fast comparison is possible, since we transformed them already into a common space w.r.t. rotation and scale. Two SIFT descriptors are said to be *matching*, if their cost is below a certain threshold. This parameter is quite unproblematic in our setting since a larger set of matching features will just have a slight impact on running time. We fixed it to 0.2 in our implementation, where 0 means that the two descriptors are identical, and 2 is the largest possible distance between two diametrically opposed descriptors.[1]

3.2 Obtaining Rotation Invariance for 3D SIFT

A single vector does not suffice to fix the orientation of a 3D neighborhood. Figure 4 shows how the existing methods by Scovanner et al. [29] and Cheung et al. [7] use the gradient direction $\mathbf{T}$ to fix two out of three possible rotations. Specifically, any rotation around $\mathbf{T}$ cannot be detected with their SIFT descriptors.

We fix this issue by computing the Frenet-Serret frame in the gradient field as the base coordinate system. This gives us a local, orthogonal coordinate system. We use it to assign a unique orientation to the neighborhood box before computing the SIFT descriptor.

The Frenet-Serret frame has three orthogonal axes, i.e., tangent $\mathbf{T}$, normal $\mathbf{N}$, and bi-normal $\mathbf{B}$. They are computed from local derivatives, but conceptually they refer to the tangent, normal and bi-normal of the gradient curve $\mathbf{r}$ passing through a given point:

$$\mathbf{T} = \frac{d\mathbf{r}}{ds} \tag{5}$$

$$\mathbf{N} = \frac{d\mathbf{T}}{ds} \tag{6}$$

$$\mathbf{B} = \mathbf{T} \times \mathbf{N}. \tag{7}$$

To make the frame more reliable, we compute the averages of the tangent, normal and bi-normal of all the Frenet-Serret frames within a $3 \times 3 \times 3$ neighborhood. Let us denote them with $\overline{\mathbf{T}}$, $\overline{\mathbf{N}}$, and $\overline{\mathbf{B}}$. To obtain a orthogonal coordinate system, we apply the Gram-Schmidt process, which can be written as

$$\mathbf{T}' = \overline{\mathbf{T}} \tag{8}$$

$$\mathbf{N}' = \overline{\mathbf{N}} - \frac{\overline{\mathbf{N}} \cdot \mathbf{T}'}{\mathbf{T}' \cdot \mathbf{T}'}\mathbf{T}' \tag{9}$$

$$\mathbf{B}' = \overline{\mathbf{B}} - \frac{\overline{\mathbf{B}} \cdot \mathbf{T}'}{\mathbf{T}' \cdot \mathbf{T}'}\mathbf{T}' - \frac{\overline{\mathbf{B}} \cdot \mathbf{N}'}{\mathbf{N}' \cdot \mathbf{N}'}\mathbf{N}', \tag{10}$$

where $\mathbf{T}'$, $\mathbf{N}'$, and $\mathbf{B}'$ are further normalized to have unit lengths.

This coordinate system gives us a reliable basis for a rotation-invariant SIFT descriptor. Figure 4 gives an illustration. We tested the rotation invariance in a practical setting, see Section 5.5.

[1] Since the descriptors are normalized to unit length, two descriptors $\mathbf{k}$ and $\mathbf{m}$ have a cost of 2, iff $\mathbf{k} = -\mathbf{m}$.

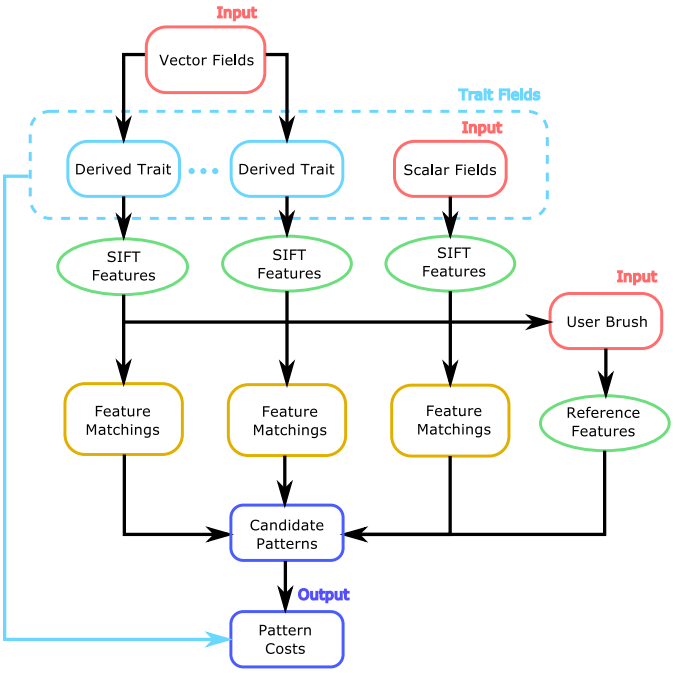


Figure 5. The pipeline of the proposed algorithm. Firstly, vector fields and scalar fields are all converted to trait fields. Later, SIFT features are extracted from each trait field independently. We select SIFT features which have intersection with the user brushed box as the reference features. For each reference feature, each of its matchers within a cost threshold determines a location of the candidate pattern regarding to the transformation between features. The actual pattern cost for each candidate pattern is the weighted sum of the costs in all trait fields.

4 Pattern Matching in Multi-Fields

We define a *pattern* as a user-defined 3D box in the data. It is characterized by its orientation, extent, location, and, most importantly, by the data values of the individual fields in the multi-field data set within this box. To match this pattern means to find locations in the domain where the individual fields of the multi-field data set attain similar values. Figure 5 depicts the algorithmic pipeline.

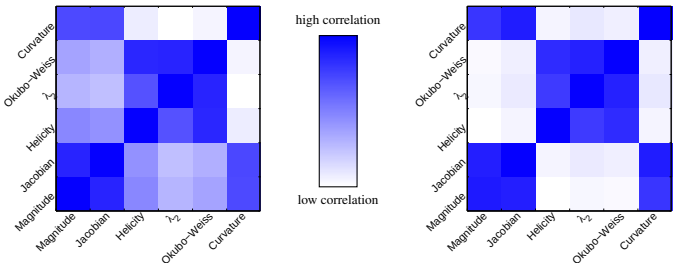
In the following, we show how we use the SIFT features from the previous section to effectively and efficiently match such patterns. Section 4.2 discusses the first stage of our algorithm, which is feature-based. Section 4.3 deals with the subsequent region-based part that leads to a dense output: a scalar field giving the similarity between the pattern and any other location. Before we come to that, we will first discuss how to incorporate vector fields into our setup in the next section.

4.1 Trait fields

SIFT features are defined for scalar fields. We are not aware of an extension to vector fields. For the purpose of this paper, we choose to incorporate vector fields indirectly by computing SIFT features for derived scalar fields. These fields can be directly incorporated into our multi-field approach. In general, all scalar fields for which we compute SIFT features will be called *trait fields* in the following, as they bear a characteristic trait of the underlying multi-field.

For a 3D vector field $\mathbf{v} = (u, v, w)^T$ with its Jacobian $\mathbf{J} = [\mathbf{v}_x \mathbf{v}_y \mathbf{v}_z]$, we consider the following trait fields, which reflect important characteristics of $\mathbf{v}$:

- magnitude $||\mathbf{v}||$
- norm of the Jacobian $||\frac{d\mathbf{v}}{dx}\frac{d\mathbf{v}}{dy}\frac{d\mathbf{v}}{dz}||$
- divergence $u_x + v_y + w_z$
- vorticity magnitude $||(w_y - v_z, u_z - w_x, v_x - u_y)^T||$



(a) Borromean data set. (b) Rayleigh-Bénard flow.

Figure 6. Trait correlations. Some traits correlate strongly with each other (dark blue patches).

- helicity $u(w_y - v_z) + v(u_z - w_x) + w(v_x - u_y)$
- λ_2-criterion [16]
- Okubo-Weiss criterion [15]
- curvature of the stream lines [35].

Some of these trait fields will only make sense in some applications. As an example, most fluid simulations are executed under the assumption of a divergence-free fluid, i.e., the divergence field is constant zero. Other trait fields may be highly correlated and therefore redundant. For example, both the λ_2-criterion and the Okubo-Weiss criterion describe vortex structures in flows.

In essence, each application has to decide on the specific set of trait fields. To aid such a decision, we provide some guidance in Figure 6. Here, we computed the pairwise correlation between trait fields. We employ the *gradient similarity* measure (GSIM) with the correlation characteristic ac_N as discussed in Sauber et al. [27]. The correlation patterns are rather similar for the two examined vector fields in Figure 6. Helicity, λ_2, and Okubo-Weiss are highly correlated. A high correlation can also be observed between magnitude, the norm of the Jacobian, and the stream line curvature. In our experiments, we often choose just one member of such a correlation group for the actual pattern matching.

A similar approach can also be taken to incorporate tensor fields. We leave this for future work.

4.2 Feature-Based Search for Candidate Patterns

Consider a user-brushed 3D box as the reference pattern $\mathbf{P}$. We define its center $\mathbf{c_P}$ as the geometric center of the box, its orientation $\mathbf{O_P}$ is given by the world coordinate system, and the scale is given as $s_\mathbf{P} = 1$.

The following procedure is carried out individually for each trait field. We define the set of *reference SIFT features* $\mathbf{K}$ as those SIFT features whose supporting neighborhood fully or partly overlaps with the box of the reference pattern $\mathbf{P}$. A SIFT feature $\mathbf{k} \in \mathbf{K}$ comes with a position $\mathbf{c_k}$, an orientation of its neighborhood $\mathbf{O_k} = (\mathbf{T}', \mathbf{N}', \mathbf{B}')$, and a scale $s_\mathbf{k}$.

For each $\mathbf{k} \in \mathbf{K}$, we find the set of matching SIFT features $\mathbf{M_k}$ (cf. Section 3). Only SIFT features from the same trait field are compared with each other. Again, each SIFT feature $\mathbf{m} \in \mathbf{M_k}$ comes with a position $\mathbf{c_m}$, an orientation $\mathbf{O_m}$, and a scale $s_\mathbf{m}$.

Given $(\mathbf{c_k}, \mathbf{O_k}, s_\mathbf{k})$ and $(\mathbf{c_m}, \mathbf{O_m}, s_\mathbf{m})$, we can compute a linear transformation that maps the neighborhood box of $\mathbf{k}$ to the neighborhood box of $\mathbf{m}$. We apply this transformation to the user-defined reference pattern $\mathbf{P}$ and obtain a *candidate pattern* $\mathbf{P}'$ by computing its center $\mathbf{c_{P'}}$, orientation $\mathbf{O_{P'}}$, and scale $s_{\mathbf{P}'}$ as follows:

$$\mathbf{c_{P'}} = \mathbf{c_m} + \mathbf{O_m}(\mathbf{Iv}) \tag{11}$$

$$\mathbf{O_{P'}} = \mathbf{O_m}\mathbf{O_k}^{-1} \tag{12}$$

$$s_{\mathbf{P}'} = \frac{s_\mathbf{m}}{s_\mathbf{k}} \tag{13}$$

$$\text{with} \quad \mathbf{v} = s_{\mathbf{P}'}\,\mathbf{O_k}^{-1} \cdot (\mathbf{c_P} - \mathbf{c_k}), \tag{14}$$

where $\mathbf{I}$ is the identity matrix. Figure 7 depicts how we find candidate patterns from matching SIFT features.

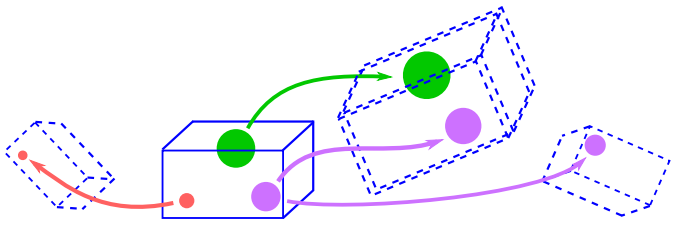


Figure 7. Finding candidate patterns from matching SIFT features. The solid blue box denotes the user-defined reference pattern **P**. The colored circles denote different types and scales of SIFT features. The SIFT features overlapping with **P** are called reference features. Matching them to other SIFT features in the data set leads to transforming the reference pattern **P** to several candidate patterns **P**′, shown as dashed boxes. A subsequent region-based cost computation yields the final result.

4.3 Region-Based Cost Computation

After matching the SIFT features, we have a number of candidate patterns. They are transformed versions of the reference pattern, i.e., translated, rotated, and scaled 3D boxes. Each candidate pattern has been computed using the SIFT features of a specific trait field. We disregard this information now. We are only interested in the boxes themselves. More precisely, we want to know whether the trait fields attain similar values in a candidate pattern versus the reference pattern.

We compute the cost between the reference pattern and a candidate pattern in each trait field using the L2-norm within these boxes. The final cost is then a weighted sum of the individual costs:

$$\text{cost}(\mathbf{P}') = \sum_t \sum_i w_t ||\mathbf{P}_{it} - \mathbf{P}'_{it}||^2, \tag{15}$$

where t denotes the trait field, i is the index of the grid vertices of $\mathbf{P}$, and w_t is the importance weight for trait field t.

The costs of all candidate patterns are combined in one global scalar field, which we use to visualize the matching result.

5 Evaluation and Discussion

5.1 Rationale Behind the Feature-Based Approach

A straightforward way of computing the cost scalar field without the help of SIFT features would be the following: We could sample the space of all possible rotations, translations, and scales that can be applied to the reference pattern. After transforming the reference pattern, we would evaluate (15) for each transformed box and combine all these costs in the global cost scalar field.

A simple example shows that this approach is more time-consuming than our matching using SIFT features. Consider a single scalar field such as the electrostatic potential of the Benzene molecule shown in Figure 11. It is sampled on a 257^3 grid. In order to shift the reference pattern to every grid vertex, we require just as many translations. Sampling 3D rotations is not as straightforward as many sampling schemes typically oversample the polar regions. Such matters are discussed in the Robotics community; Kuffner [18] shows how unit quaternions help in uniformly sampling rotations. Let us assume that 100 rotation samples allow for enough accuracy. Finally, let us use the same 18 scale samples we use in our approach (see Section 3). We end up with a total of over 30 billion transformations.

On the other hand, our approach uses sparse feature sampling to drastically decrease the number of considered pattern transformations. The numbers are shown in Table 1. For the Benzene data set, we have 38 SIFT features and the pattern overlaps with 8 of those. We have to compare those 8 features with the others, and create a transformed candidate pattern if they match. The entire pipeline including the cost computation is done in under 0.2 seconds. The detection of the SIFT features themselves takes about 80 seconds, but they can be reused for any pattern.

Our computation time depends on the total number of SIFT features n, and the number of selected SIFT features m, where $m \leq n$ but typically $m \ll n$. Every selected feature is compared against all others. Our current implementation does this straightforwardly using linear search with a computational complexity of $O(mn)$. For very large numbers of SIFT features, it may be beneficial to use a hashing function for a faster comparison. We refer the interested reader to the extensive discussions of this topic by Paulevé et al. [24] as well as Muja and Lowe [23]. The latter comes with a public domain software library for this purpose.

For very small numbers of SIFT features, one may have the issue that parts of the domain are not covered and therefore unavailable for defining patterns. We did not encounter this issue in our experiments, and deem it rather negligible for two reasons:

- Consider a region in a scalar field with uniform behavior, i.e., constant scalar value or no monotony breaks. Such a region is typically of low interest to a user and will also not have SIFT features. On the contrary, SIFT features indicate regions with non-uniform behavior in a scalar field, which are typically the regions of highest interest in an analysis.

- Our method works with multi-fields. A lack or sparsity of SIFT features in one trait field is compensated by the SIFT features of the other trait fields. In fact, our method needs only a single SIFT feature in the user-defined reference pattern **P**.

Table 1 shows that we find a high number of SIFT features in all our experiments. Figure 12 shows a number of trait fields with their SIFT features, some of which are covered densely and some sparsely.

Ultimately, it is a data- and application-dependent question whether the patterns of interest are covered by SIFT features. We will discuss this in the next section for the special case of pattern matching in vector fields.

5.2 Comparison to Vector-Based Matching Methods

In the following, we compare our method to the pattern matching methods for vector fields due to Ebling et al. [9] and Heiberg et al. [13]. We implemented both approaches in our system.

The major difference is that the above approaches work directly and only on vector-valued data, whereas our method deals with vector fields indirectly by considering multiple derived scalar fields. This has consequences when analyzing numerical flow simulations:

- Flow data sets are multi-fields. They contain the vector-valued flow velocity as well as scalar-valued traits such as viscosity, pressure, and density. Our method is able to incorporate these scalar fields, at the expense of treating the velocity field indirectly.

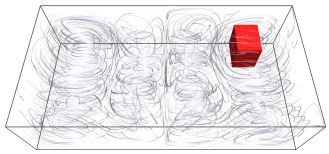
- The matching results of Ebling et al. [9] and Heiberg et al. [13] are not Galilean invariant, since they are directly obtained from the velocity vectors. Many interesting flow features such as the von Kármán vortex street (cf. Figure 16) can only be observed in the velocity field when choosing a particular frame of reference. This is often not trivial. On the other hand, our method incorporates Galilean invariant scalar fields and is therefore able to detect many flow features directly.

We devised a fair comparison to Ebling et al. [9] and Heiberg et al. [13] by concentrating on a single vector field and selecting a feature that can be observed in the original frame of reference. We chose the Rayleigh-Bénard flow as shown in Figure 8. It is a flow due to thermal convection of heated and cooled boundaries, obtained using the software NaSt3DGP [11][2]. The flow has 8 vortex structures. Half of them have a left-handed sense of rotation, and the other half a right-handed one. We selected a vortex with a left-handed sense of rotation as the search pattern (Figure 8a).

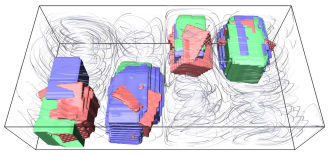
All three methods correctly identify the 4 left-handed vortices.[3] Figures 8b–e show the matched regions as well as stream line renderings

[2] NaSt3DGP was developed by the research group in the Division of Scientific Computing and Numerical Simulation at the University of Bonn.

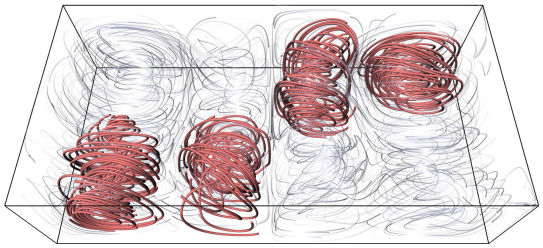
[3] As a side note, our method picks up on the rotation sense due to the *Helicity* trait field (the sign gives the chirality), whereas the other methods detect the rotation sense from the orientation of the velocity vectors.



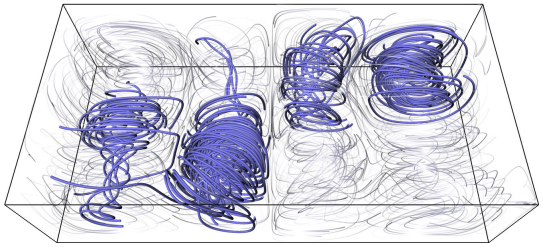
(a) Selection of a vortex with a left-handed sense of rotation.



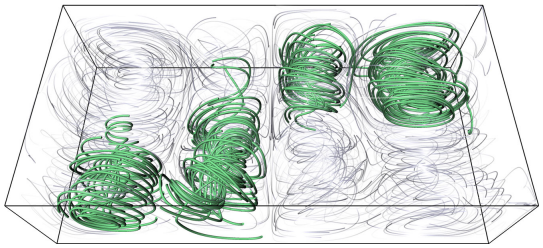
(b) Detections of all three methods overlaid.



(c) Stream lines seeded in the voxels detected by our method. Computation time for the pattern matching is 8 s, and requires the reusable SIFT features (49 s).



(d) Stream lines seeded in the voxels detected by Ebling et al. [9]. Computation time: 284 s.



(e) Stream lines seeded in the voxels detected by Heiberg et al. [13]. Computation time: 158 s.

Figure 8. Comparison of the results using our multi-field method and the vector-based methods from Ebling et al. [9] and Heiberg et al. [13]. Shown is the Rayleigh-Bénard convection flow. All three methods correctly identify the four vortices with left-handed sense of rotation, but require significantly different computation times.

highlighting these vortices.[4]

The computation times are rather different. Our method needs 8 seconds for the pattern matching itself, and 49 seconds for the precomputation of the SIFT features in seven trait fields. Note that these SIFT features can be reused for further pattern matching in this flow, i.e., to identify the right-handed vortices. The vector-based pattern matching methods need significantly more computation time. The method of Ebling et al. [9] requires 284 seconds, the method of Heiberg et al. [13] requires 158 seconds.

As a second experiment, we designed 12 vector field patterns in the spirit of Ebling et al. [9] and Heiberg et al. [13], including vortices, convergent/divergent flow, and all first-order critical points. We made sure to include all patterns mentioned in these papers. As we show in the supplemental material, we find SIFT features for all these patterns. Hence, our method can handle the same vector field patterns as these previous methods.

[4]Note that we seeded the stream lines in the matched regions, but their integration was unrestricted.

There is one exception. One pattern cannot be observed with our method: parallel flow as in $\mathbf{v}(\mathbf{x}) = (1,0,0)^T$. Such a flow does not contain any flow features and all conceivable derived scalar fields do not contain SIFT features.

5.3 Discussion of False Negatives and False Positives

A false negative is a pattern $\mathbf{P}'$ that has not been found despite it being a translated, rotated, and scaled copy of the reference pattern $\mathbf{P}$. It is easy to see that our algorithm *cannot* have false negatives: Each SIFT feature $\mathbf{k}$ in the reference pattern $\mathbf{P}$ has a corresponding SIFT feature $\mathbf{k}'$ in $\mathbf{P}'$, because SIFT features are exceptionally invariant to translation, rotation, and scaling as we show in Section 5.5. The SIFT features $\mathbf{k}$ and $\mathbf{k}'$ are practically identical, which makes it easy to find $\mathbf{P}'$.

It becomes more interesting when noise or other deformations cause a difference between $\mathbf{P}$ and $\mathbf{P}'$. SIFT features react gradually to such changes (see the noise experiment in Section 5.5), i.e., the difference between $\mathbf{k}$ and $\mathbf{k}'$ is comparable to the difference between $\mathbf{P}$ and $\mathbf{P}'$. Considering an increasing difference between $\mathbf{k}$ and $\mathbf{k}'$, we will stop computing the cost between $\mathbf{P}$ and $\mathbf{P}'$ after exceeding a certain threshold. As already discussed in Section 3.1, we can afford a larger threshold, since this will just have a slight impact on running time.

A false positive is a pattern $\mathbf{B}$ that has been found despite it being rather different from the reference pattern $\mathbf{P}$. It is easy to see that our algorithm *cannot* have false positives: Consider that $\mathbf{B}$ and $\mathbf{P}$ have no matching SIFT features. Then $\mathbf{B}$ will also not be considered a matching pattern of $\mathbf{P}$. Alternatively, consider that $\mathbf{B}$ and $\mathbf{P}$ have matching SIFT features. Then the region-based cost computation (15) will return the actual cost between these two patterns.

5.4 Discussion of Parameters

The following parameters pertain to the computation of the SIFT features. We list their values here and refer to the literature [22] for background information. See also Section 3.

- Number of octaves: 3
- Number of Gaussian blurred fields: 6
- Initial blur $\sigma_0 = 1.6$
- Factor for subsequent blurring $k = 2^{1/3}$
- DoG extrema neighborhood: $3 \times 3 \times 3 \times 3$
- SIFT descriptor neighborhood: $27 \times 27 \times 27$
- Number of SIFT histograms: $3 \times 3 \times 3$
- Number of bins per histogram: 12

Two parameters pertain to the core of our method:

SIFT descriptor matching cost threshold Two SIFT descriptors match, if their cost from Equation (4) is below this threshold. Increasing this threshold gives more matching SIFT features, which leads to more region-based cost computations using Equation (15), i.e., it has a slight impact on performance, but not on the quality of the matching. Decreasing this threshold gives less matching SIFT features, which imposes less flexibility for deformation of patterns other than translation, rotation and scaling. We suggest to err on the side of increasing this threshold.

Pattern matching cost threshold After computing the region-based costs using Equation (15), this threshold is defined by the user for the isosurface or volume rendering visualizations showing the matching patterns. Examples are shown throughout the paper, e.g., in Figures 8b and 11b. Increasing this threshold shows more matching patterns, decreasing it shows less.

5.5 Evaluation of the Invariance of the SIFT Features

We made the following experiment to evaluate how invariant the 3D SIFT features are under rotation, translation, scaling, and noise. Figure 9 shows the setup: a single scalar field with values in the range $[0,1]$ and the isosurface at 0.5 is a round, axis-aligned box. This field has 8 SIFT features corresponding to the corners of the cube. Let us denote this set with $\mathbf{A}$.

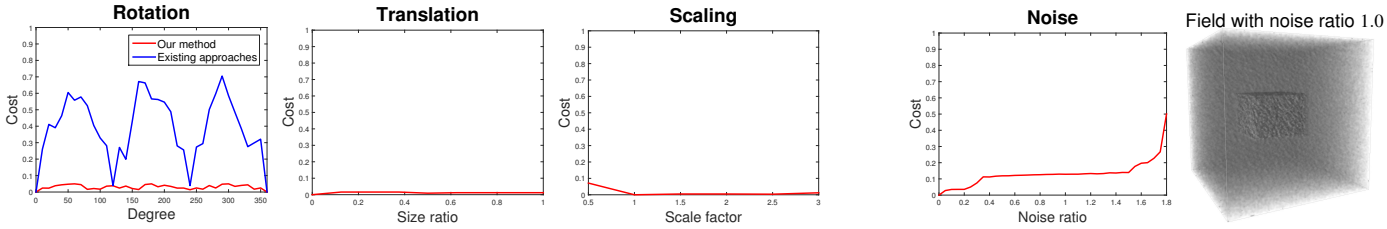


Figure 10. Evaluation of the invariance of the SIFT features against rotation, translation, scaling, and adding noise. The setup is shown in Figure 9. For the rotation evaluation, we included the results of existing approaches [7, 29]. In these plots, lower values are better.

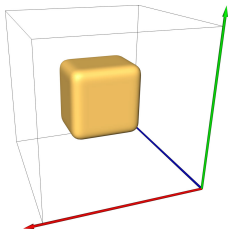


Figure 9. Setup for the evaluation of the invariance of the SIFT features. The scalar field has values in the range $[0,1]$ and the isosurface at 0.5 is a round, axis-aligned box. For the evaluation, we rotate, translate and scale the domain as well as adding noise to the data. Results are shown in Figure 10.

After transforming or adding noise to the scalar field, we compute the set of new SIFT features $\mathbf{B}$. We compare the sets $\mathbf{A}$ and $\mathbf{B}$ using Hausdorff distance $H(\mathbf{A},\mathbf{B})$:

$$D(\mathbf{x},\mathbf{Y}) = \min\{\text{cost}(\mathbf{x},\mathbf{y}) \mid \mathbf{y} \in \mathbf{Y}\} \tag{16}$$

$$H(\mathbf{A},\mathbf{B}) = \max\{\max\{D(\mathbf{a},\mathbf{B}) \mid \mathbf{a} \in \mathbf{A}\}, \max\{D(\mathbf{b},\mathbf{A}) \mid \mathbf{b} \in \mathbf{B}\}\} \tag{17}$$

Figure 10 shows the results. For interpretation, remember that the SIFT feature descriptors have unit length, i.e., the largest possible cost is 2. We will detail the experiments in the following.

Rotation We rotate the domain in steps of 10 degrees around the axis $(1,1,1)^T$. We also made this experiment with the existing approaches [7, 29]. As discussed earlier, they are not fully invariant against rotation as the blue curve shows. Our results are shown by the red curve and show a high rotation invariance.

Translation We translate the domain along the x-axis until the shifted distance reaches the size of the box. The result shows the expected high invariance against translation.

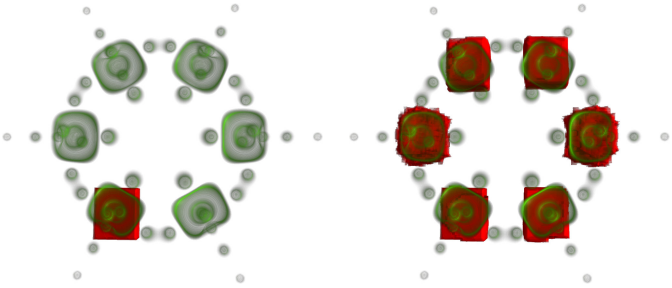
Scaling We scale the domain uniformly with the factors $[0.5, 1.0, 1.5, 2.0, 2.5, 3.0]$. The cost between 1.0 and 3.0 is constant zero, which shows full scale invariance. The cost is slightly higher for the factor 0.5, because when the box becomes small enough, the largest scale in the scale space sees the corners as one feature.

Noise We add white noise with increasing amplitude to the data. The shown *noise ratio* refers to the amplitude. A noise ratio of 1 means that the value range of the noise and the data are equal. The cost remains quite low until a noise ratio of 1.5. After that, the data is corrupted and the cost increases rapidly. As it can be seen in Figure 10, a noise ratio of 1 creates already a highly distorted field, yet the cost is still within an acceptable range. This shows how stable the SIFT features are against noise.

5.6 Invariance against Intensity Scaling or Shifting

The invariances discussed above relate to transforming the domain. What about transformations of the *data values* such as a scaling or shifting? SIFT features are naturally invariant against it, since the SIFT descriptor contains only gradient information and is normalized.

In detail: A multiplication of the scalar field with a constant factor (value scaling) changes only the magnitude of the gradient and not its direction. The normalization of the SIFT descriptor makes it invariant against this. An addition of a constant value (value shifting) does not change the gradient of the scalar field at all. Hence, the SIFT descriptor is invariant against it. Finally, the locations of SIFT features are computed as extrema of the DoG fields, which are unaffected by these transformations.



(a) Selection. (b) Matching patterns.

Figure 11. Selection of an area around a carbon atom in the electrostatic potential of the Benzene molecule. This pattern can be found six times in this scalar field, namely around all six carbon atoms.

6 Results

All results have been computed in a single thread on a 3.1 GHz Intel Xeon E31225 with 16 GB main memory. Computation times as well as the number of SIFT features are shown in Table 1.

Benzene We start with a single scalar field to showcase the matching qualities of our algorithm in a setting that is easy to understand. The electrostatic potential of the Benzene molecule in Figure 11 exhibits a 6-fold symmetry. We selected the area around one of the six carbon atoms that make up the inner ring of this molecule. The result of the pattern matching highlights all six carbon atoms. This real-world example shows once more that our 3D SIFT descriptors are rotation invariant.

Borromean Magnetic Flux Vector Field Figures 12-14 show the Borromean magnetic flux – a vector field from an experiment studying magnetic energy decay [5]. In its initial state, it features interlocked magnetic rings. The shown state exhibits already a large amount of decay. We include this vector field to show how our multi-field pattern matching can help in understanding vector fields despite working only with derived scalar fields.

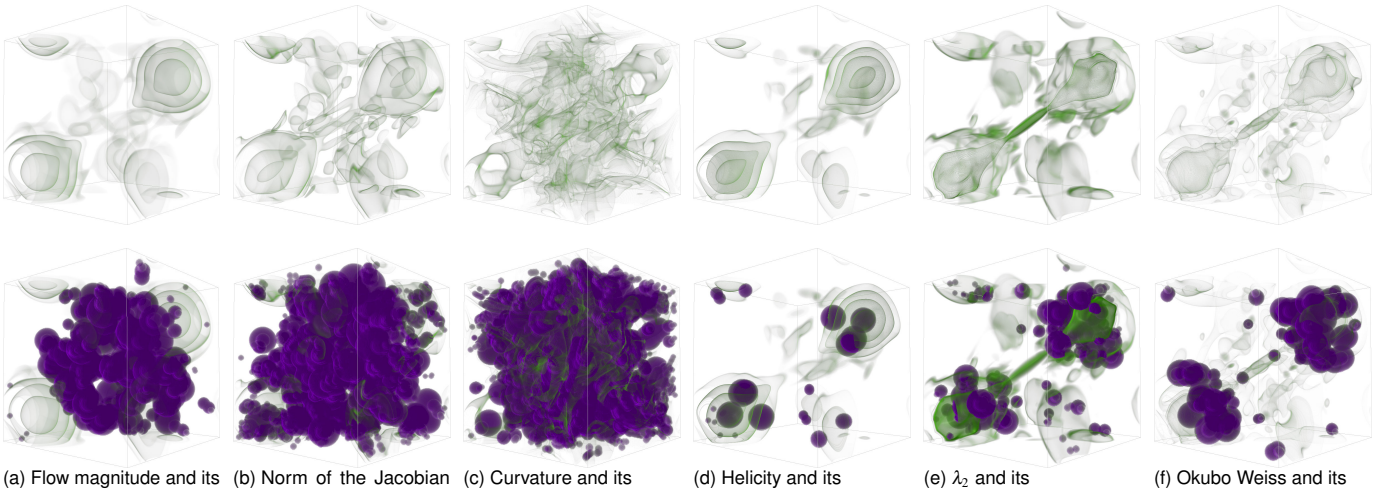
Figure 12 shows the six trait fields we computed from the original vector field: magnitude, norm of the Jacobian, stream line curvature, helicity, λ_2, and Okubo-Weiss. Besides a volume rendering of each trait field, Figure 12 also shows their SIFT features as spheres. The size of the spheres denotes the scale of the SIFT feature, i.e., the size of the supporting neighborhood. Some of the trait fields are densely covered with SIFT features, while others exhibit them only in distinguished regions. This is not much of an issue, since (i) all regions have coverage by at least one of the trait fields, and (ii) SIFT features are only used to generate candidate patterns and the subsequent cost computation involves again all traits (see Section 4.3).

We made two pattern selections in this data set. In Figure 13 we selected one of the outer rings. The pattern matching yields the other outer ring on the opposite side of the volume. The stream line rendering highlights these structures. Note how our multi-field approach is able to address structures that are inherent to the vector field.

For Figure 14 we selected a region in the middle of the domain. The matching result shows a ring-like structure. The stream line rendering

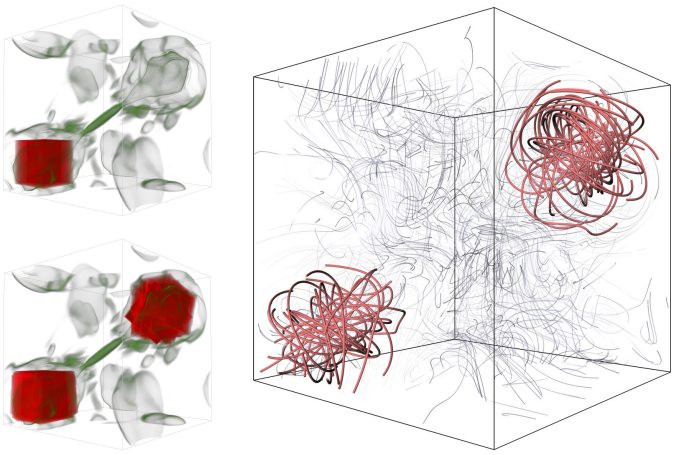
Data set	Dims	# Traits	# SIFT features		Timings in Seconds			
			total	selected	SIFT localization	SIFT Descriptor	Feature Matching	Cost Computation
Benzene	$257 \times 257 \times 257$	1	38	8	82	0.002	0.001	0.154
Borromean	$256 \times 256 \times 256$	6	5773	92	6×120	0.1 - 0.8	0.008	8.5
				45			0.006	2.5
Climate	$480 \times 241 \times 27$	3	737	20	3×5	0.07 - 0.13	0.005	0.1
Isabel	$500 \times 500 \times 100$	11	23057	2047	11×120	1	1.6	3.5
Rayleigh-Bénard	$127 \times 127 \times 127$	7	1054	56	7×7	0.1 - 0.2	0.01	7.7
Square Cylinder	$115 \times 64 \times 48$	7	1207	670	7×0.5	7×0.03	0.03	2

Table 1. Running times and number of SIFT features for the data sets used in this paper.



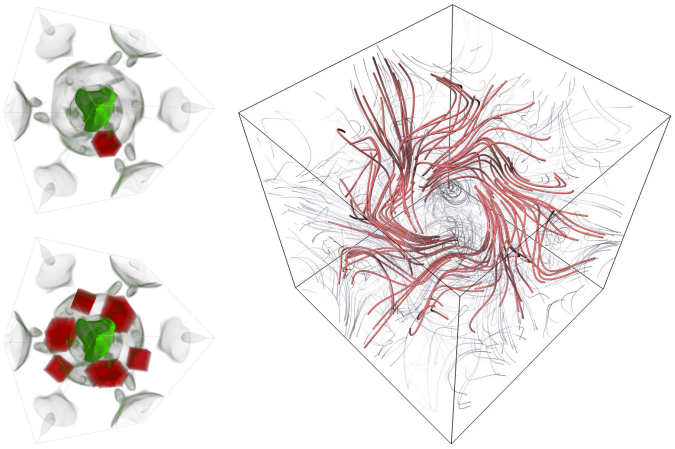
(a) Flow magnitude and its SIFT features. (b) Norm of the Jacobian and its SIFT features. (c) Curvature and its SIFT features. (d) Helicity and its SIFT features. (e) λ_2 and its SIFT features. (f) Okubo Weiss and its SIFT features.

Figure 12. Trait fields of the Borromean magnetic flux vector field. Their SIFT features are shown as spheres.



(a) Selection (top) and matching patterns. (b) Stream lines shown for the entire vector field and highlighted within the matching patterns.

Figure 13. Borromean data set with matching outer rings.

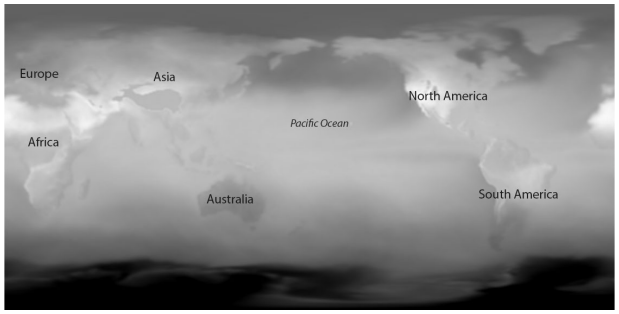


(a) Selection (top) and matching patterns. (b) Stream lines shown for the entire vector field and highlighted within the matching patterns.

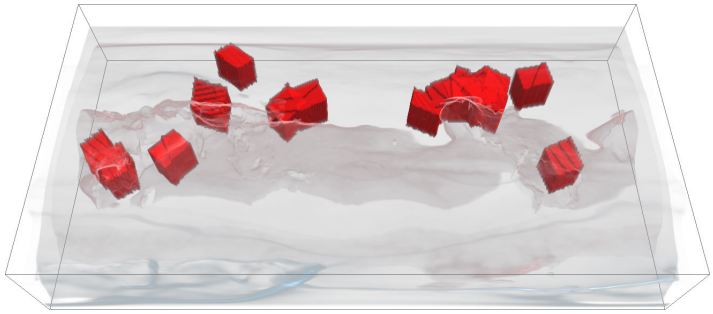
Figure 14. Borromean data set with the inner ring revealed by our method.

reveals that these are the remains of the interlocked rings from the beginning of the magnetic energy decay experiment.

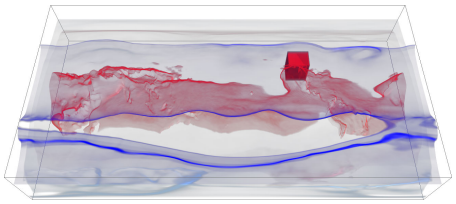
Climate Multi-Field Data Set Figure 15 shows a multi-field climate data set. This is a time step of a large re-analysis of the world's climate in the years 1979–2013. The data set is courtesy of Dim Coumou and Thomas Nocke from the Potsdam Institute for Climate Impact Research (PIK). This 3D multi-field data set spans the entire planet and several kilometers of the atmosphere. Figures 15c–e show volume renderings of the considered trait fields iso-pressure height, temperature, and wind speed. We selected a region at the North-West coast of the USA. Interestingly, the iso-pressure height did not produce any matching patterns, since this particular location does not contain SIFT features. This is not much of a surprise, since this data set shows only structures near the ground, but the isosurfaces are almost planar in higher regions. Anyway, we got plenty of matches in the other two trait fields. In Figure 15b we show the final pattern matching result for all traits combined.

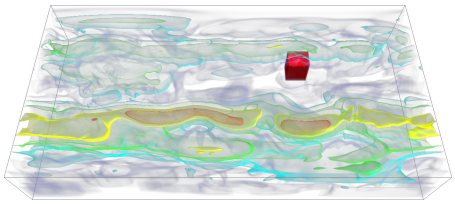


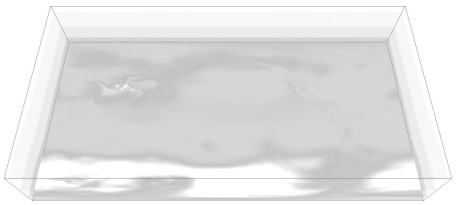
(a) Overview

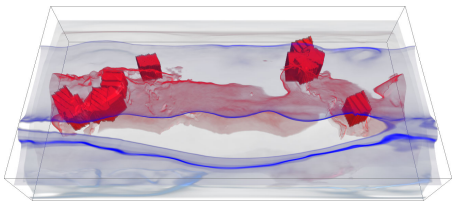


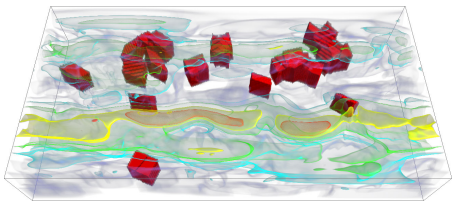
(b) Multi-field pattern matching result with all three traits.











(c) Iso-Pressure Height: selection (top) and matching patterns (none).

(d) Temperature: selection (top) and matching patterns.

(e) Wind speed: selection (top) and matching patterns.

Figure 15. Climate multi-field data set with three traits.

Hurricane Isabel Multi-Field Data Set In Figure 1 we applied our method to the Hurricane Isabel data set from the IEEE Visualization 2004 contest. This is a complex 3D time-dependent data set produced by the Weather Research and Forecast (WRF) model courtesy of NCAR and the U.S. National Science Foundation (NSF). It contains 10 scalar fields and 1 vector field. For our purposes, we considered all 10 scalar fields as well as the magnitude of the flow (wind speed).

We made a more advanced experiment with this data set: we select the eye of the hurricane in the time step $T = 20$ and make this our reference pattern. However, we apply the pattern matching to the following time steps $T \in [21, \ldots, 41]$. All 11 trait fields are considered for this. As Figure 1 as well as the accompanying video show, this leads to a stable tracking of the eye of the hurricane.

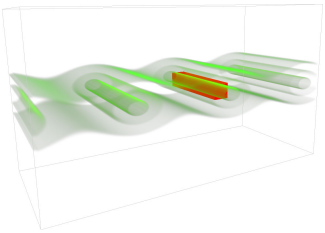
Square Cylinder Flow In Figure 16 we take this approach even one step further. Instead of applying a pattern from one time step to another, we apply a pattern from a different data set to the square cylinder flow [4, 34].

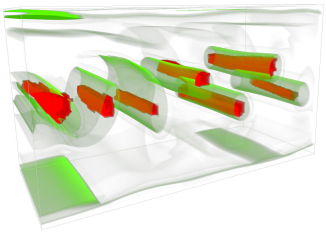
The interesting part in this flow is the von Kármán vortex street. It is characterized by alternating vortices created by periodic vortex shedding directly behind the cylinder. In our experiment, we attempt to capture these vortices using pattern matching. However, unlike our other experiments, the pattern is not a selection from the same data set, but an analytic vortex description often referred to as the *Stuart Vortex*:

$$\mathbf{v} = \left(\sinh(2y) \, , \, \frac{1}{4}\sin(2x) \, , \, z\,(\cosh(2y) - \frac{1}{4}\cos(2x)) \right)^T .$$

We sampled this field and computed the same trait fields that we also have for the square cylinder flow, namely: magnitude, stream line curvature, helicity, divergence, vorticity magnitude, λ_2, and Okubo-Weiss.

Figure 16a shows the selection in the Stuart Vortex. This selected vortex pattern has been applied to the unrelated square cylinder flow.





(a) Analytic vortex description (Stuart Vortex) used to define a reference pattern.

(b) Matching patterns in the numerical simulation of a flow behind a square cylinder.

Figure 16. A pattern has been analytically "designed" and then applied to a real-world flow in order to find all vortex structures in the von Kármán vortex street. Both images show a volume rendering of the *vorticity magnitude* trait field of the respective flow.

Figure 16b shows the matching result, which nicely covers the vortices in the von Kármán vortex street. This example shows that our algorithm can also be applied in scenarios, where a reference pattern is "designed" beforehand as a way to describe features of interest.

7 Conclusion

We introduced a novel pattern matching approach for multi-field data sets. It bundles the information from different fields into the description of a pattern. The method is very efficient, since we work with a sparse set of features to drastically reduce the search space for the pattern matching. We discussed how to achieve full rotation invariance for the SIFT features.

For future work, tensor fields should be taken into consideration. Is there a set of characteristic traits for tensor fields that can be used for our multi-field approach?

REFERENCES

[1] S. Battiato, G. Gallo, G. Puglisi, and S. Scellato. Sift features tracking for video stabilization. In *Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference on*, pages 825–830. IEEE, 2007. 2

[2] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *International journal of computer vision*, 74(1):59–73, 2007. 2

[3] R. Bujack, I. Hotz, G. Scheuermann, and E. Hitzer. Moment invariants for 2d flow fields using normalization. In *Pacific Visualization Symposium (PacificVis), 2014 IEEE*, pages 41–48. IEEE, 2014. 1, 2

[4] S. Camarri, M.-V. Salvetti, M. Buffoni, and A. Iollo. Simulation of the three-dimensional flow around a square cylinder between parallel walls at moderate Reynolds numbers. In *XVII Congresso di Meccanica Teorica ed Applicata*, 2005. 9

[5] S. Candelaresi and A. Brandenburg. Decay of helical and nonhelical magnetic knots. *Phys. Rev. E*, 84(1):16406–16416, 2011. 7

[6] H. Carr and D. Duke. Joint contour nets. *Visualization and Computer Graphics, IEEE Transactions on*, 20(8):1100–1113, 2014. 2

[7] W. Cheung and G. Hamarneh. n-sift: n-dimensional scale invariant feature transform for matching medical images. In *In Proceedings of the Fourth IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2007 (ISBI 2007*, pages 720–723, 2007. 1, 2, 3, 7

[8] H. Doleisch, M. Gasser, and H. Hauser. Interactive feature specification for focus+context visualization of complex simulation data. In *Proc. VisSym 03*, pages 239–248, 2003. 2

[9] J. Ebling and G. Scheuermann. Clifford convolution and pattern matching on vector fields. In *Proc. IEEE Visualization*, pages 193–200, 2003. 1, 2, 5, 6

[10] J. Flusser and T. Suk. Rotation moment invariants for recognition of symmetric objects. *IEEE Transactions on Image Processing*, 15(12):3784 – 3790, 2006. 1

[11] M. Griebel, T. Dornseifer, and T. Neunhoeffer. Numerical simulation in fluid dynamics, a practical introduction. In *SIAM*, 1998. 5

[12] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part IV*, ECCV'12, pages 459–472, Berlin, Heidelberg, 2012. Springer-Verlag. 1

[13] E. Heiberg, T. Ebbers, L. Wigstrom, and M. Karlsson. Three dimensional flow characterization using vector pattern matching. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):313–319, 2003. 1, 2, 5, 6

[14] L. Huettenberger, C. Heine, H. Carr, G. Scheuermann, and C. Garth. Towards multifield scalar topology based on pareto optimality. *Computer Graphics Forum*, 32(3pt3):341–350, 2013. 2

[15] J. Hunt. Vorticity and vortex dynamics in complex turbulent flows. *Proc CANCAM, Trans. Can. Soc. Mec. Engrs*, 11:21, 1987. 4

[16] J. Jeong and F. Hussain. On the identification of a vortex. *J. Fluid Mechanics*, 285:69–94, 1995. 4

[17] R. Jianu, C. Demiralp, and D. H. Laidlaw. Exploring 3d dti fiber tracts with linked 2d representations. *Transactions on Visualization and Computer Graphics*, 15(6):1449–1456, 2009. 2

[18] J. J. Kuffner. Effective sampling and distance metrics for 3d rigid body path planning. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 4, pages 3993–3998. IEEE, 2004. 5

[19] Y. Li, C. Wang, and C. Shene. Streamline similarity analysis using bag-of-features. In *Proc. SPIE*, volume 9017, pages 90170N–90170N–12, 2013. 2

[20] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society. 2

[21] D. G. Lowe. Local feature view clustering for 3d object recognition. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–682. IEEE, 2001. 2

[22] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004. 1, 2, 3, 6

[23] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *In VISAPP International Conference on Computer Vision Theory and Applications*, pages 331–340, 2009. 5

[24] L. Paulevé, H. Jégou, and L. Amsaleg. Locality sensitive hashing: A comparison of hash function types and querying mechanisms. *Pattern Recognition Letters*, 31(11):1348–1358, 2010. 5

[25] C. Rössl and H. Theisel. Streamline embedding for 3d vector field exploration. *IEEE Transactions on Visualization and Computer Graphics*, 18(3):407–420, 2012. 2

[26] H. Saikia, H.-P. Seidel, and T. Weinkauf. Extended branch decomposition graphs: Structural comparison of scalar data. *Computer Graphics Forum (Proc. EuroVis)*, 33(3):41–50, June 2014. 2

[27] N. Sauber, H. Theisel, and H.-P. Seidel. Multifield-graphs: An approach to visualizing correlations in multifield scalar data. *IEEE Transactions on Visualization and Computer Graphics*, 12:917–924, 2006. 2, 4

[28] M. Schlemmer, M. Heringer, F. Morr, I. Hotz, M. Hering-Bertram, C. Garth, W. Kollmann, B. Hamann, and H. Hagen. Moment invariants for the analysis of 2d flow fields. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1743–1750, 2007. 2

[29] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th International Conference on Multimedia*, MULTIMEDIA '07, pages 357–360, New York, NY, USA, 2007. ACM. 1, 2, 3, 7

[30] J. Tao, C. Wang, and C. Shene. Flowstring: Partial streamline matching using shape invariant similarity measure for exploratory flow visualization. In *Proc. IEEE Pacific Visualization*, March 2014. 2

[31] D. Thomas and V. Natarajan. Multiscale symmetry detection in scalar fields by clustering contours. *Visualization and Computer Graphics, IEEE Transactions on*, 20(12):2427–2436, Dec 2014. 2

[32] D. M. Thomas and V. Natarajan. Symmetry in scalar field topology. *IEEE TVCG*, 17(12):2035–2044, 2011. 2

[33] Z. Wang, J. M. Esturo, H.-P. Seidel, and T. Weinkauf. Pattern search in flows based on similarity of stream line segments. In *Proc. Vision, Modeling and Visualization*, 2014. 1, 2

[34] T. Weinkauf, H.-C. Hege, and H. Theisel. Advected tangent curves: A general scheme for characteristic curves of flow fields. *Computer Graphics Forum (Proc. Eurographics)*, 31(2):825–834, April 2012. Eurographics 2012, Cagliari, Italy, May 13 - 18. 9

[35] T. Weinkauf and H. Theisel. Curvature measures of 3D vector fields and their applications. *Journal of WSCG*, 10(2):507–514, 2002. 4

[36] Z. Yi, C. Zhiguo, and X. Yang. Multi-spectral remote image registration based on sift. *Electronics Letters*, 44(2):107–108, 2008. 2