Multiscale Symmetry Detection in Scalar Fields by Clustering Contours

Dilip Mathew Thomas and Vijay Natarajan, *Member, IEEE*

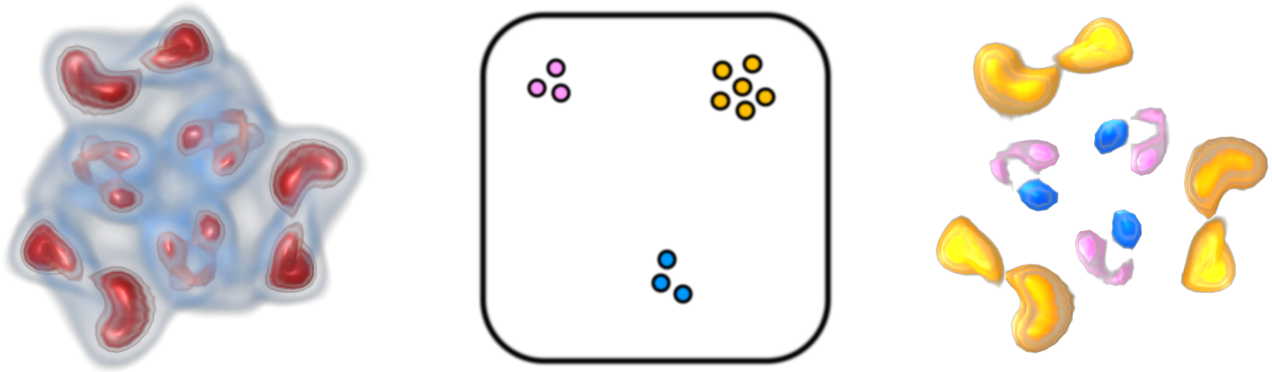


Fig. 1. Clustering based analysis detects symmetry at different scales in a 3D cryo-electron microscopy image of AMP-activated kinase (EMDB-1897). (left) The three-fold rotational symmetry is apparent from the volume rendering. (center) Contours are represented as points in a high-dimensional shape descriptor space (illustrated in 2D). Symmetric contours form a cluster in the descriptor space and can be easily identified. Three such clusters are shown in gold, blue, and pink. (right) Three symmetric regions of different sizes, highlighted in gold, blue, and pink, detected by the method.

Abstract—The complexity in visualizing volumetric data often limits the scope of direct exploration of scalar fields. Isocontour extraction is a popular method for exploring scalar fields because of its simplicity in presenting features in the data. In this paper, we present a novel representation of contours with the aim of studying the similarity relationship between the contours. The representation maps contours to points in a high-dimensional transformation-invariant descriptor space. We leverage the power of this representation to design a clustering based algorithm for detecting symmetric regions in a scalar field. Symmetry detection is a challenging problem because it demands both segmentation of the data and identification of transformation invariant segments. While the former task can be addressed using topological analysis of scalar fields, the latter requires geometry based solutions. Our approach combines the two by utilizing the contour tree for segmenting the data and the descriptor space for determining transformation invariance. We discuss two applications, query driven exploration and asymmetry visualization, that demonstrate the effectiveness of the approach.

Index Terms—Scalar field visualization, symmetry detection, contour tree, data exploration.

1 Introduction

Many scientific experiments and simulations generate scalar field data that contain symmetric or repeating patterns. In many disciplines, symmetry plays an important role in studying the underlying scientific phenomenon. For example, in crystallography, symmetry information is used to determine the structure of a crystal [7]. In product design, symmetry is important to ensure functional efficiency and optimal manufacturing cost [3]. Symmetry is a useful cue in biology for determining growth and development of organs [35]. Since the study of symmetric features is of great interest in scientific data analysis, the problem of detecting symmetry in scalar fields has received considerable attention among researchers in the recent past [10, 13, 19, 38, 39].

Automatic detection of symmetry in scalar fields is a challenging problem and the quest for a widely applicable, efficient, and robust method for symmetry detection is ongoing. Though symmetry identification in scalar fields is a relatively new area of research, the problem of detecting symmetry in shapes has been well studied in the geometry processing community. These studies have established that clustering based analysis result in superior performance and robust identification of symmetry. Some of these methods have been extended to scalar fields and they operate by determining symmetry transformations through aggregation of local symmetry of sample points of the domain. Symmetry in shapes is associated with a group structure on geometric objects that are invariant under transformations. In scalar fields, it is more meaningful to relax this constraint and identify all repeating occurrences since this is more useful for data exploration. Scalar field datasets are typically represented using scalar values assigned to a discrete set of sample points that represent the domain under consideration. However, the domain and the scalar values are assumed to be continuous by interpolating the values at the sample points. Therefore, the sample points in a scalar field capture the lowest level of information. In practice, scientists are more interested in higher level features, extracted through methods like segmentation and isosurface extraction, for studying the underlying physical

- *Dilip Mathew Thomas is with Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India. E-mail: dilip@csa.iisc.ernet.in.*
- *Vijay Natarajan is with Department of Computer Science and Automation, and Supercomputer Education Research Centre, Indian Institute of Science, Bangalore, India. E-mail: vijayn@csa.iisc.ernet.in.*

Manuscript received 31 Mar. 2014; accepted 1 Aug. 2014; date of publication xx xxx 2014; date of current version xx xxx 2014.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

phenomenon. Hence, symmetry identification methods that are based on local information available at the sample points encounter considerable difficulty in representing and extracting meaningful symmetric regions. Moreover, these methods are computationally expensive since the number of sample points in scalar field datasets is typically orders of magnitude higher than that in geometric shape datasets.

Though it is clear from methods proposed in the geometry processing community that a clustering based analysis offers significant advantages in recognizing symmetry, we believe that unlike shapes, low-level information available at the sample points of the domain is not suited for symmetry identification in scalar fields. In this work, we propose a novel symmetry detection method based on the idea of clustering contours. Isosurfaces are extensively used in studying scalar field datasets and contours, which are connected components of isosurfaces, capture information about a scalar field at a macroscale. Therefore, contours are more suitable for a clustering based analysis as opposed to sample points of the domain. Contours belonging to regions with symmetric scalar field distribution are also symmetric. Using an appropriate shape descriptor, our method maps contours to points in a descriptor space such that the distance between points in the descriptor space is a measure of similarity between the contours. As a result, points in the descriptor space representing symmetric contours lie in close proximity to each other and form clusters in the descriptor space. The region of the domain corresponding to each such contour can be extracted and these regions are reported as symmetric. Note that the choice of the shape descriptor is not fixed and depending on the noise characteristics and the definition of similarity relevant to the application of interest, an appropriate descriptor may be used. Fig. 1 illustrates our approach on a 3D cryo-EM image of AMP-activated kinase (EMDB-1897) with three-fold rotational symmetry. Our method identifies symmetric regions of different scales. The large-scale features shown in gold and the small-scale features shown in blue and pink highlight the multiscale aspect of our approach.

The main contributions of this paper are the following:

- A formulation of the problem of symmetry detection in scalar fields as a clustering problem in a shape descriptor space. This model provides a lot of flexibility in analyzing similarity of scalar fields as well as handling noise since it allows the shape representation and the descriptor space to be varied.
- A novel representation of contours as points in a contour descriptor space. Similarity between contours is naturally defined as the distance between points in this space. This is a generic representation of independent interest and we show its benefit in similarity analysis of scalar fields.
- A robust algorithm to detect symmetric regions at multiple scales. Though geometry based symmetry detection methods are typically computationally costly, we design an efficient algorithm that employs elegant optimizations by incorporating topological information about the contours using the contour tree.
- Applications to query driven exploration and asymmetry visualization.

Symmetry information in scalar fields has been used for transfer function design, exploration of isosurfaces, selection of cross-section planes and view directions, linked selection and editing, query driven exploration, and visualization of features through dual rendering [10, 19,38,39]. We believe that as better techniques for symmetry detection are developed, many more applications will emerge.

2 Related Work

Existing symmetry identification methods in scalar fields can be broadly classified into two categories, namely, geometry based methods and topology based methods. We briefly review these methods in this section.

2.1 Geometry based approaches

Several methods have been proposed in the literature for detecting symmetry in shapes as described in the survey paper by Mitra et al. [21]. Some of these methods [2, 12, 22] have been applied to scalar fields [10,13,19]. However, they struggle to address the challenges in extending geometric methods to scalar fields. Scalar field datasets are significantly larger in size and hence symmetry detection is computationally costly. Geometric methods typically consider geometric information derived from a small region around each sample point of the domain for symmetry recognition. A direct extension of this approach to scalar fields suffers from the difficulty of capturing important features and leads to poor performance in extracting higher level features and handling of noise in the data. Moreover, the scalar field is considered to be continuous over the domain by interpolating the values at the sample points. Inspecting only the sample points introduces additional challenges due to discretization errors since the symmetric counterpart for a given point may be an interpolated point.

Hong and Shen [10] propose a method to detect global reflective symmetry by identifying planes of reflection that minimize the difference between the scalar value at a point and its reflection. This method is computationally inefficient and cannot be easily extended to identify other types of symmetry. Kerber et al. [13] build a graph network of crease line features and detect symmetry by computing transformations that match subgraphs within the crease line network. Since only a small subset of features in scalar fields contain crease lines, this method is not very useful in practice. Masood et al. [19] detect symmetry by identifying symmetry transformations as clusters in the space of all transformations. The clusters are generated by aggregating local symmetry transformations of pairs of points in the domain. This method relies on local signatures of sample points for determining transformations and as a result several parameters need to be tweaked at various stages of the symmetry detection pipeline to limit the adverse effects of variations in the local signatures and discretization errors. Moreover, the transformation space often contains additional transformations that introduces artifacts. The above methods compute transformations between candidate pairs for identifying symmetry and are computationally costly. Moreover, they are driven by purely local geometric measures and do not incorporate any criterion to either recognize important features or discard pairs corresponding to noise. Our method, on the other hand, uses topological information derived from the contour tree to infer importance of a feature and this allows the design of a feature-aware algorithm for symmetry identification.

Bruckner et al. propose an information theoretic approach for isosurface similarity detection [4, 8]. They use mutual information between distance transforms to quantify the information common to two isosurfaces and build a similarity map between all pairs of isosurfaces. Clusters with high mutual information correspond to similar isosurfaces both within and across datasets. While this method is related since it is also based on isosurface similarity, the goal here is to select a subset of representative and possibly important isovalues. The distance transform is used to identify redundant isovalues corresponding to families of isosurfaces that form an onion-peel like layered arrangement. The goal of our method, on the other hand, is to locate regions that are similar and hence we compute similarity between contours and not isosurfaces. While the isosurface similarity map based method is limited to analyzing similarity between pairs of datasets, the descriptor space can be used to analyze multiple datasets simultaneously. Similarity between different scalar fields has also been studied by measuring the extent of overlap between contours [31, 32]. The distance transform descriptor and the overlap measure are affected by changes in orientation. Our method is not restricted to a particular choice of descriptor. Based on the requirements of the application under consideration, our method can be adapted to be sensitive or insensitive to orientation.

2.2 Topology based methods

Thomas and Natarajan propose topology based methods for symmetry identification and these methods are computationally efficient because they operate on graph representations of the scalar field like the contour tree [38] and the extremum graph [39]. The contour tree based method assumes that the subtrees of the contour tree corresponding to symmetric regions are structurally similar. They detect symmetry

by evaluating structural similarity between the subtrees using a similarity score that measures the overlap between the branches of the trees. This method can find symmetry at multiple scales but cannot handle noise that destroy the repeating structure of the subtrees. The extremum graph based method selects a set of extrema called seed set and estimates distances robustly through a graph traversal procedure. A carefully chosen distance threshold is used to disconnect the graph and classify the seeds into different groups called super-seeds. A region growing procedure is then used to identify the symmetric region corresponding to each super-seed. This procedure makes a strong assumption that the symmetric regions can be identified purely from the proximity relationship between the seeds. Hence, it relies heavily on a meaningful selection of seed set which involves significant effort and understanding about the symmetry of the domain. In addition, this method requires several thresholds to be set.

The above methods, being topological in nature, do not ensure that the regions reported by them are indeed geometrically symmetric while our method, being geometric in nature, ensures that the regions extracted are symmetric. Moreover, current methods compare candidate regions pairwise and rely on a similarity threshold to classify them into symmetric groups. Determining the similarity threshold is a challenge when using datasets with varying characteristics. Clustering based analysis avoids the need for pairwise comparisons. Instead, the symmetric regions are directly obtained as clusters in the descriptor space. Similarity between scalar fields have been studied in the context of shape matching applications by using graph matching methods on discrete approximations of the contour tree [9, 43]. The contour tree provides metadata information about the contours and allows integration of topological information in geometric processing. Thus, by utilizing the descriptor space to capture geometric information about contours together with the power of the contour tree as a topological abstraction of contours, our method offers significant advantages over existing symmetry identification methods.

3 Definitions

Consider a *scalar field*, $f : \mathbb{M} \to \mathbb{R}$, defined on a simply connected domain $\mathbb{M}$. The preimage of f for a given value $u \in \mathbb{R}$, $f^{-1}(u)$, is called a *level set* of f. A level set may have multiple components and each component is called a *contour*. Consider a sweep of the domain using level sets in the order of increasing function values. A contour is created at a *minimum*, may merge with another contour at a *join saddle* or split into different contours at a *split saddle*, and is destroyed at a *maximum*. The *contour tree* is a topological data structure that captures these changes in the connectivity of the level sets, see Fig. 2. Let R be an equivalence relation defined on points in $\mathbb{M}$: xRy for $x, y \in \mathbb{M}$ if x, y belong to the same contour. The contour tree is the quotient space induced by this relation.

Subdomains $\mathbb{M}_1, \mathbb{M}_2 \subseteq \mathbb{M}$ are said to be *symmetric* if there is a transformation T such that $\mathbb{M}_2 = T(\mathbb{M}_1)$ and $f(x) = f(T(x))$ for all $x \in \mathbb{M}_1$. If c_1 and c_2 are contours of the same level set that belong to $\mathbb{M}_1$ and $\mathbb{M}_2$ respectively, it is easy to see that if $\mathbb{M}_1$ and $\mathbb{M}_2$ are symmetric then c_1 and c_2 are also symmetric. We make use of this property and detect symmetric subdomains by identifying symmetry of the contours belonging to the subdomains. The above definition of symmetry requires computation of the symmetry transformation T, which is a costly operation. Therefore, we use an alternate definition of symmetry. Let $\mathbb{C}$ be the set of all contours. Consider a function $g : \mathbb{C} \to \mathbb{R}^n$ such that $g(c) = g(T(c))$ where T is a transformation. In other words, g is a function that maps each contour to a point in a high-dimensional space such that a contour and its copies are mapped to the same point. The point to which a contour is mapped is called a *descriptor* and the high-dimensional space is called the *descriptor space*, see Fig. 3. For illustration, the descriptor space is shown in 2D but the actual dimension of the space depends on the choice of the descriptor. The distance between contours in the descriptor space is a measure of their similarity. In practice, scalar fields do not exhibit perfect symmetry and therefore it is important to detect symmetry in an approximate sense. Ideally, deviation from perfect symmetry should be measured in the space of shapes but it is more convenient to measure deviations in the descriptor space. If contours c_1 and c_2 are not perfectly symmetric, then c_1 and c_2 will not be mapped to the same point in the descriptor space. The distance between the contours in the descriptor space, $\|g(c_1) - g(c_2)\|$, will be indicative of the deviation from perfect symmetry.

Definition (Symmetric Contours)**.** *Contours c_1 and c_2 are perfectly symmetric if $\|g(c_1) - g(c_2)\| = 0$, where $\|\cdot\|$ is a norm in the descriptor space. They are ε-symmetric if $\|g(c_1) - g(c_2)\| \leq \varepsilon$, for $\varepsilon > 0$.*

Shape descriptors have been extensively used in the geometry processing community for shape matching and there is a vast collection of research papers in this area [15, 26, 37, 40]. It is possible that a shape descriptor may incorrectly map contours c_1 and c_2 to the same point even when they have totally different shapes. A good shape descriptor should discriminate well between different shapes and minimize such incorrect mappings.

4 Symmetry Detection via Contour Clustering

Methods based on clustering [16,22,23,27,41,42] have shown superior performance in identifying symmetry in shapes. However, directly extending these methods to scalar fields is non-trivial. In this section, we describe a novel symmetry detection method based on the idea of clustering contours of the scalar field.

4.1 Overview

Fig. 4 illustrates the main steps of our algorithm. Given a scalar field as input, we generate a set of contours. For each contour thus generated, a descriptor is computed. The descriptor for each contour can be considered to be a point in a high-dimensional descriptor space. The descriptor space is a transformation-invariant space, i.e., it reverses the effect of geometric transformation on contours. Thus, perfectly symmetric contours are mapped to the same point in the descriptor space. Imperfections in symmetry results in imperfections in the mapping. Since contours with similar shape have similar descriptors, the points in the descriptor space representing approximately symmetric contours will lie in close proximity to each other. Therefore, symmetric contours can be recognized by identifying clusters in the descriptor space. The volumetric regions represented by the contours within a cluster are then reported as symmetric regions.

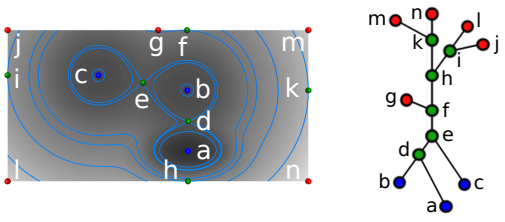


Fig. 2. The contour tree captures changes in the topology of the level sets of a scalar field. (left) A scalar field and its level sets. (right) The corresponding contour tree. The red nodes are local maxima, the blue nodes are local minima, and the green nodes are saddles.

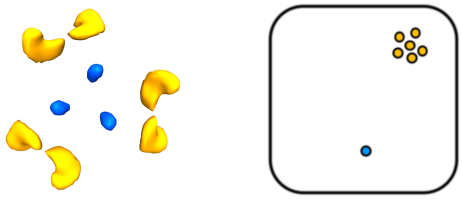


Fig. 3. Mapping contours to points in a descriptor space. Symmetric contours shown in blue are mapped to the same blue point in the descriptor space. Six approximately symmetric contours shown in gold are mapped to six points that lie in close proximity to each other in the descriptor space.

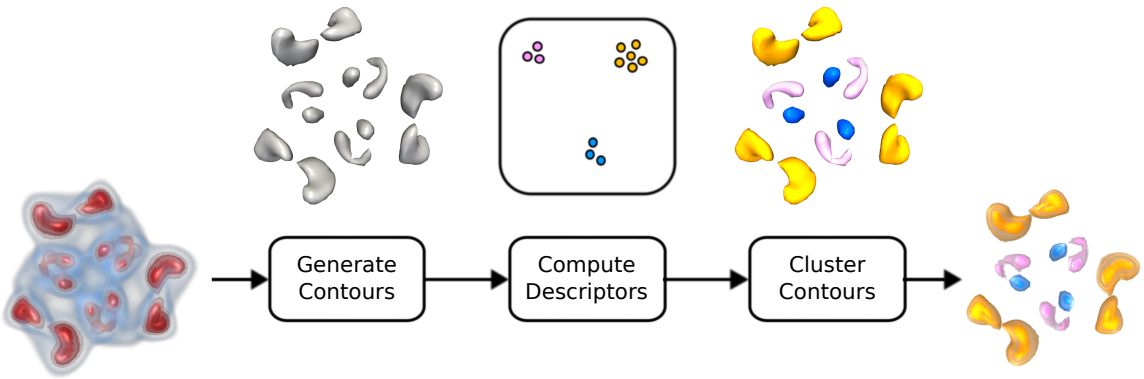


Fig. 4. Symmetry detection pipeline. Contours are extracted from the scalar field and a descriptor is generated for each contour. A similarity score is estimated between pairs of contours based on the distance between the points in the descriptor space. Next, the set of symmetric contours are identified via clustering. Finally, the region of the domain to which each symmetric contour belongs is extracted and reported.

4.2 Contour Generation

Our algorithm assumes that each region of interest in the domain is represented by a contour belonging to it. Hence, it is important to use a sampling strategy that generates a contour from each region of interest. The obvious method for generating contours is to sample isovalues uniformly from the range of the function values and extract contours corresponding to these isovalues. A coarse uniform sampling may not generate contours within a specific region and thus fail to recognize it as a symmetric region. On the other hand, a fine sampling may generate multiple contours within the same region and redundant computations. Ideally, each symmetric region should require only a single representative contour for its detection. The contour tree is a powerful tool that encapsulates information about the evolution of contours [5] and we leverage information obtained from the contour tree for optimal generation of contours. Each arc of the contour tree represents a family of contours that are nested one inside the other forming offset surfaces similar to layers of onion peel. Hence, to capture the geometry of the region of the domain corresponding to an arc of the contour tree, we select only one contour from each arc of the contour tree.

Selecting a representative contour from each arc of the contour tree ensures that no regions are missed in the subsequent symmetry analysis. An arc in the contour tree may either represent a feature associated with a single extremum or a region formed by the merger of multiple features and hence associated with multiple extrema. As a result, our method can detect symmetry at multiple scales. However, for noisy scalar fields, a large number of arcs of the contour tree may correspond to noise. Selecting a contour from each arc of the contour tree will result in significant amount of computational time spent in processing these noisy contours. To overcome this problem, we generate a contour from an arc of the contour tree only if the arc is deemed to represent a feature and not noise. The definition of noise is subjective and depends on the application. In this work, we consider an arc to be noise if the volume of the largest contour associated with the arc is below a user defined *noise threshold* δ. Assuming that the scalar field is uniformly sampled, the volume of a contour is approximated as the number of vertices of the domain enclosed by the contour. Since each arc of the contour tree can be associated with the set of vertices of the domain that comprise the subvolume corresponding to the arc, this estimation of the volume can be done efficiently [5]. In the absence of the metadata information provided by the contour tree, all contours would have had to be treated as equally important. In summary, contour tree driven sampling both ensures that each region has a unique representative contour and avoids sampling of contours from noisy regions.

4.3 Contour Representation in Descriptor Space

Once a representative contour is generated from each region of interest, the next step is to generate its descriptor. The similarity score between a pair of contours is estimated using the distance between their descriptors. Designing shape descriptors for matching and retrieving similar shapes is a well studied area in the geometry processing community. The notion of similarity is subjective and varies from application to application. A major advantage of our method is that it is not restricted, in principle, to a particular choice of the shape descriptor. Instead, it offers flexibility in choosing the descriptor appropriate for an application. Hence, our method may be viewed as a generic framework for identifying similar regions in a scalar field. For example, if an application is interested in identifying similarity only with respect to rotation, a rotation invariant descriptor may be used. The only prerequisite on the descriptor is that it should be discriminative, i.e., similar contours should be mapped to points that are nearby in the descriptor space while contours that differ from each other should be mapped to far away points. Therefore, it is important to use shape descriptors with high precision and recall ratios [15] for applications that cannot tolerate false positives during shape retrieval.

4.4 Contour Clustering

After the descriptor generation stage, any standard clustering method may be used to locate the clusters in the descriptor space that represent symmetric contours. However, requiring the explicit generation of descriptors as a constraint limits the flexibility of using our approach as a generic framework for similarity detection. We observe that mapping of contours to points in the descriptor space is not a prerequisite for clustering. A similarity correspondence graph can be constructed from a set of contours by representing each contour as a node in the graph and inserting an edge between two nodes if the respective contours are similar. It is easy to see that the contours that are similar form a clique under this representation [16] and these cliques can be identified to detect a set of similar contours. Thus, given a procedure that assigns a similarity score between pairs of contours, further processing is performed solely on the graph and is independent of the actual definition of similarity. This allows considerable freedom in choosing a similarity measure that is relevant to an application. In particular, for symmetry identification, the distance between points in the descriptor space is used to assign the score between pairs of contours.

Given a set of contours marked as symmetric, the arc in the contour

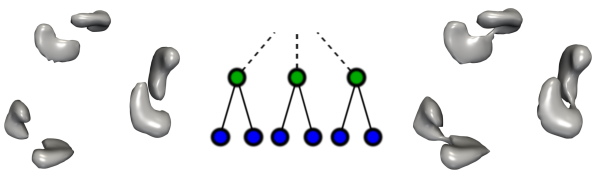


Fig. 5. When contours merge, their shape change significantly. (left) Six contours before they merge. (center) Part of the contour tree depicting merging of pairs of contours. (right) Three contours after the merge.

tree to which each contour belongs to can be determined. The region of the domain enclosed by the largest contour of the arc can then be extracted and reported as a symmetric region. Although this works well in practice, note that the contour itself may have evolved due to the merging of contours nested within it. An application with stricter requirements on symmetry detection may need to also incorporate the symmetry of these nested contours in the algorithm. This presents a challenge in directly using clusters in the descriptor space for detecting symmetric regions because the descriptor space is not continuous with respect to the evolution of the shape of a contour during a level set sweep, see Fig. 5. Recall that a cluster in the descriptor space represents a set of contours with the same shape. As illustrated in the left and right figures, the shape of individual nested contours before merge is different from the shape of the contour after the merge. Therefore the points corresponding to the contours before and after the merge may not be part of the same cluster.

To address this issue, we incorporate the similarity score between the children contours (contours before merging) into the calculation of the similarity score between a given pair of parent contours (contours after merging). Let p and q be two parent contours with children contours ${c_1}^p, \ldots, {c_n}^p$ and ${c_1}^q, \ldots, {c_m}^q$, respectively. Assume that the score between two children contours ${c_i}^p$ and ${c_j}^q$ is known. For contours with children, the procedure below can be applied bottom up to determine their score while for contours that do not have children, the score can be directly determined from the descriptor space. To calculate the similarity score between p and q, first the contribution from the children contours is determined. We construct a bipartite graph where nodes in the two partitions are ${c_1}^p, \ldots, {c_n}^p$ and ${c_1}^q, \ldots, {c_m}^q$, see Fig. 6. An edge between ${c_i}^p$ and ${c_j}^q$ is weighted with the score between ${c_i}^p$ and ${c_j}^q$. The maximum weight matching is computed to determine the similarity score between the children contours. The score between the contours p and q obtained directly from the descriptor space is added to the value of maximum weight matching to obtain the cumulative similarity score between p and q. An analogous procedure may be used for nested contours that split from a parent contour.

5 Implementation

We now elaborate on the implementation details of our symmetry identification algorithm. We describe the factors that determine the selection of isovalues, the particular shape descriptor that we use and its properties, and the clustering algorithm we employ.

5.1 Isovalue Selection

Each arc of the contour tree encodes the range of values of the scalar field restricted to the subdomain represented by the arc. The question that remains is which function value in this range should be used to generate the representative contour for the arc. To generate a large contour belonging to the subvolume represented by an arc, the isovalue selected should be close to the saddle value at which the contour merges into another contour. However, as discussed by Thomas and Natarajan [38], perturbation in function values due to noise may lead to instability of the saddles in the contour tree. As a result, the function value at which the contours from symmetric regions merge and split may not be consistent. Fig. 10(b) shows an example where, for the same isovalue, children contours of the green contours have already merged to form a single component while the blue contours are yet to merge. We require an isovalue that is both close to the saddle and generates isocontours that are consistent, with respect to merging. A simple strategy we adopt is to ensure that none of the contours merge by selecting an isovalue that is lower than the lowest join saddle at which the contours merge. We introduce a *stabilization parameter* α that determines the length of the interval of function values within which the saddle values are not consistent with respect to merging of contours. From the list of all saddles in the contour tree, we first remove all saddles belonging to arcs that are considered to be noise with respect to the noise threshold δ. Within the reduced list, we consider each join saddle s, where contours merge into one, and select an isovalue $w = f(s) - \alpha$ if there are no other saddles whose function value lie within the interval $[w, f(s)]$. Among the set of isovalues thus generated, if w is the highest isovalue that lies in between the function values at the end points of an arc, then w is chosen as the isovalue for the arc. An analogous procedure is used to select isovalues with respect to split saddles. A procedure for determining the value of α is described in Section 7.2.

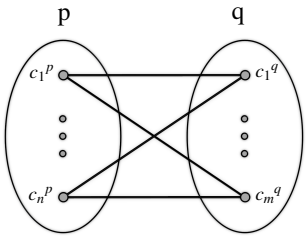


Fig. 6. Maximum weight matching is used to determine the contribution of children contours towards the similarity score between the parent contours p and q. Each edge ${c_i}^p{c_j}^q$ is weighted with the similarity score between the child contours ${c_i}^p$ and ${c_j}^q$.

5.2 Shape Descriptor

The choice of shape descriptor depends on the kind of similarity analysis required by an application. A function ϕ which satisfies the equation $\Delta\phi = -\lambda\phi$, where Δ is the Laplace-Beltrami operator, is called an eigen function of Δ and λ is called an eigen value [30]. We use the first ten non-zero eigen values of the Laplace-Beltrami spectra as the shape descriptor since noise in the shape has limited influence on the initial eigen values [15]. This descriptor has been used for shape matching and retrieval and is robust in the presence of noise, deformations in shape, and differences in the underlying triangulation [24, 28, 29]. Although it is possible that two different shapes may have the same spectra, it is very rare in practice [28]. The descriptor is discriminative with high precision and recall ratios [15]. We use the popular cotangent weighted scheme for computing the Laplace-Beltrami spectra [25]. Computation of the Laplace-Beltrami spectra on large meshes is costly and therefore we simplify the contour meshes so that the number of vertices is small. We simplify the mesh down to 1000 vertices in our experiments through edge collapses driven by the quadric error metric [11]. If the isosurfaces contain skinny triangles that result in numerical errors in the computation of the spectra, mesh quality aware isosurface generation or remeshing [33,34] may have to be performed.

5.3 Clustering

We observe that clusters in the descriptor space are well separated and therefore employ a simple scheme based on nearest neighbor search for clustering. For a given contour c, other contours that are ε-symmetric with respect to c are determined by locating points in the descriptor space that lie within a sphere of radius ε centered at p_c, where p_c is the point in the descriptor space to which c is mapped. We limit the number of points in the search space by considering only those points that represent the contours that belong to the level set of c. The ideal metric for computing distances in the descriptor space may be determined using methods like metric learning [14]. However, we use Euclidean distances since it has yielded good results for shape retrieval with the Laplace-Beltrami spectra [15, 28]. The value of the approximation parameter ε is specified by the user.

6 Applications

It is easy to see that our method can be used to demonstrate the existing applications of identifying symmetry in scalar fields reported earlier in the literature. In this section, we describe two new applications for improving exploration and visualization of scalar fields.

6.1 Query Contour Driven Exploration

As the size and complexity of data generated through imaging and simulations grows, newer paradigms for effective exploration and interaction with the data are required. Query driven exploration is one

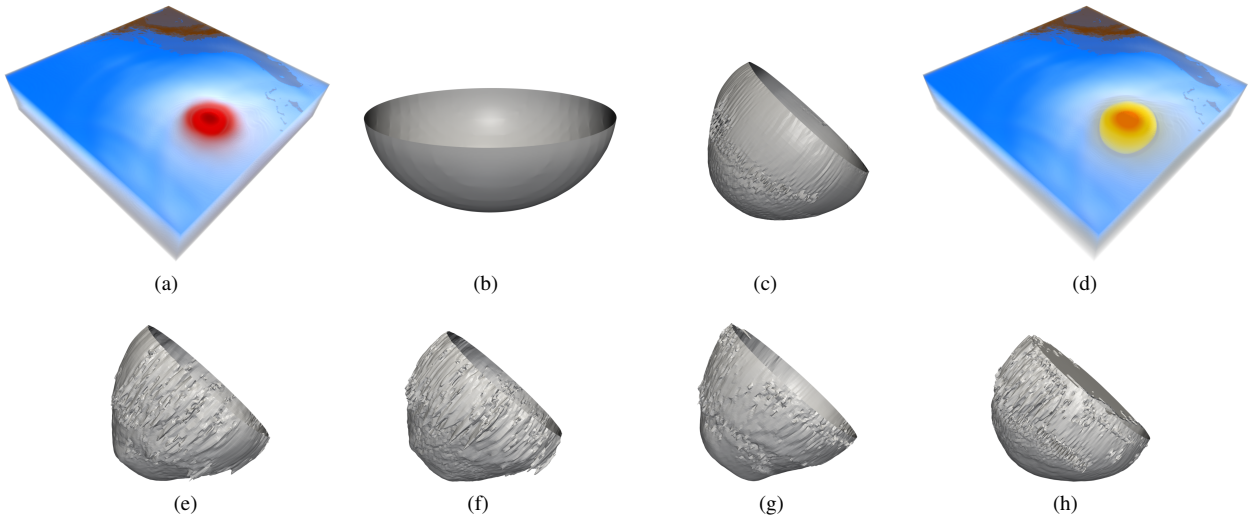


Fig. 7. Query driven exploration using contours. (a) Volume rendering of the first time step of the hurricane Isabel dataset. Low pressure region shown in red. Querying with (b) a synthetically generated half-ellipsoid finds (c) an isocontour with similar shape from the low pressure region. (d) The corresponding volumetric region shown in yellow. (e)-(h) The isocontour from the low pressure region of the first time step is employed as the query contour and the result on four different time steps, 12, 24, 36, and 48 is shown. Note that even though the contours contain noise, they are located correctly due to the robustness of our method. The corresponding volumetric regions are shown in the supplementary material.

such paradigm that facilitates navigation through complex data and is gaining popularity over the last few years [1, 17]. Symmetric regions in a dataset facilitate query driven exploration of the data [19, 39]. A region of interest that is segmented from the domain is employed as the query region and symmetry information is utilized to identify other regions similar to the query. To facilitate exploration using regions of the domain as query, the domain needs to be segmented and this is often a challenge. Instead, we visualize the evolution of contours during a level set sweep and select a contour of interest. We demonstrate that our symmetry detection method can be used to explore datasets using a contour as the query object.

A simulation of the hurricane Isabel, which struck the west Atlantic in 2003, was performed on a $500\times500\times100$ grid over 48 time steps. Fig. 7(a) shows a volume rendering of the pressure field in the first time step. The low pressure region is shown in red. A domain expert with the knowledge that isocontours in the low pressure region roughly has the shape of a hemisphere can use a synthetically generated shape, shown in Fig. 7(b), as the query object. We generate a scale invariant version of the Laplace-Beltrami spectra by dividing all the eigen values with the first non-zero eigen value. The scale invariant descriptor for the query object is then computed. Next, we apply the first and second steps of our symmetry identification method to generate different contours of the volume and compute their scale invariant descriptors. Finally, we locate the nearest neighbor of the descriptor representing the query object in the descriptor space. We identify the contour shown in Fig. 7(c) through this search. Note that the size and the exact shape of the query object is different from that of the contour. The descriptor is robust to small variations and therefore the query successfully identifies a contour from the low pressure region. The volumetric region of the domain from which the contour is generated is extracted and displayed in the context of the rest of the volume in Fig. 7(d).

We describe the above experiment to demonstrate the power of the descriptor space as a geometry based representation of the scalar field. But, we concede that using a synthetic shape as query object may not always be possible. However, the user could select one of the contours from a region of interest as the query object. It is straightforward to see that this approach may be used to query for symmetric regions within the same dataset, similar to earlier approaches [19, 39]. What is more interesting is that the same approach can be extended to search through multiple datasets. We use the contour shown in Fig. 7(c) from the first time step as the query object to search through the other time steps. The contours detected as a result of executing the query on four different time steps, 12, 24, 36, and 48, are shown in Fig. 7(e)-7(h). Observe that the contours show variations in their shape and have considerable amount of noise. Also, as opposed to the query contour, a significant portion of the mouth of the contour is closed in Fig. 7(h) as the hurricane makes landfall. The query is successful even with these challenges and emphasizes the robustness of our method. Similar to Fig. 7(d), the volumetric region of the domain can be extracted in each case as shown in the supplemental material. We also show results on another weather simulation dataset in the supplemental material.

6.2 Asymmetry Visualization

Asymmetry in scalar fields often reveals crucial characteristics of the underlying physical phenomenon. For example, asymmetry between the left and the right breasts in mammogram images is an indicator of breast cancer [36]. Given a set of contours $c_1,\ldots,c_n$, marked as symmetric by our algorithm, we determine asymmetry of contour c_k, $1 \le k \le n$, with respect to the set of remaining contours $S = \{c_1,\ldots,c_{k-1},c_{k+1},\ldots,c_n\}$. For this, we first determine the symmetry transformation that maps each $c_i \in S$ onto c_k and spatially align c_i with c_k by applying the transformation. The transformation may be determined using the ICP (Iterative Closest Point) algorithm. These spatially aligned meshes in S are then glued together to generate a single mesh, say c_{avg}, with respect to which asymmetry of c_k is determined. For this purpose, we use the `Metro` tool [6], which evaluates the distance from the vertices of the mesh c_k to the mesh c_{avg} by computing for each vertex in c_k the closest point in c_{avg}. A large distance means that there was no point corresponding to it in the other contours, an indication of asymmetry. Note that the `Metro` tool cannot compute distances between contours directly from their original spatial coordinates. It requires both the meshes to be spatially aligned. The distance computed at each vertex is stored as a scalar field of c_k and the procedure is repeated for each contour. The asymmetry in the contours can then be located by visualizing the distance fields. Fig. 8 shows a cryo-EM dataset in which the top and bottom regions are symmetric. The procedure above is applied to two contours marked as symmetric by our method. Observe that the tip of the long club-like portion of the top contour is thin while that of the bottom contour is fat. This asymmetry can be distinguished from the dark red regions that depict vertices with large distance values.

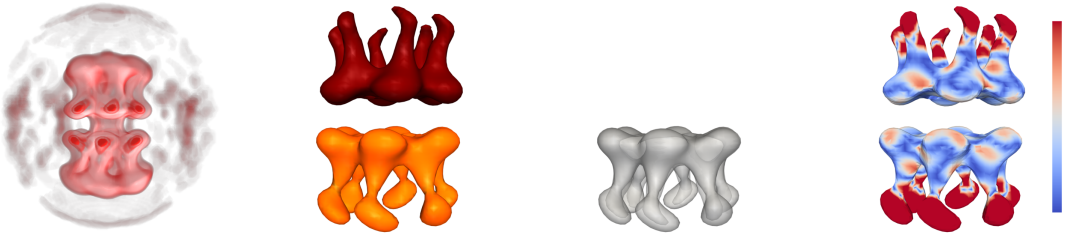


Fig. 8. Asymmetry visualization. (a) Volume rendering of a cryo-EM dataset (EMDB-1134) depicts two symmetric regions. (b) Two symmetric contours extracted by our algorithm shown in maroon and orange. The tip of the long club-like portion of the contours at the top and bottom is asymmetric. (c) The top contour is aligned with the bottom contour and a distance field is computed. (d) Visualization of the distance field. The dark red regions are asymmetric.

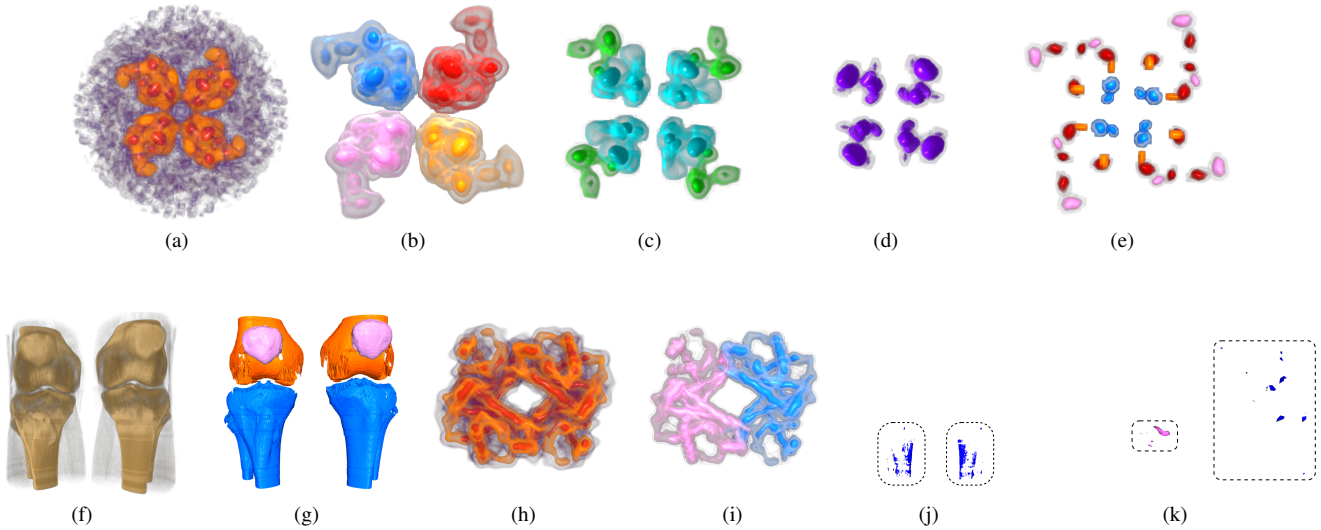


Fig. 9. Comparison with topological methods for symmetry detection. (a) Volume rendering of a cryo-EM dataset (EMDB-1654), with noisy regions depicted in violet. (b)-(e) Symmetric regions at multiple scales detected by our method. The extremum graph based method [39] reports symmetry only at the largest scale. (f) Volume rendering of a CT scan of a pair of knees. (g) Bilateral symmetry of the bones detected by our method, shown in orange, blue, and pink, even in the presence of missing regions and noise in the contours of the dataset. (h) Volume rendering of the electron density of a hemoglobin molecule (PDB-ID 1HGA) that exhibit 2-fold rotational symmetry. (i) The two symmetric regions are detected by our method even though they lie in close proximity to each other. (j) Flat regions corresponding to the global maximum of the knee dataset and (k) asymmetric distribution of the extrema of the molecule dataset, shown within dotted boxes, pose challenges in determining the seed set for the extremum graph based method.

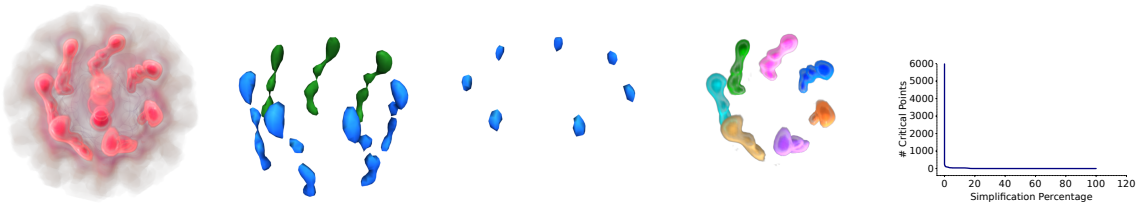
7 Discussion

In this section, we report the results of applying our method on different datasets and elaborate on comparison with existing topology based methods, selection of parameters, and computational performance.

7.1 Comparison With Topological Methods

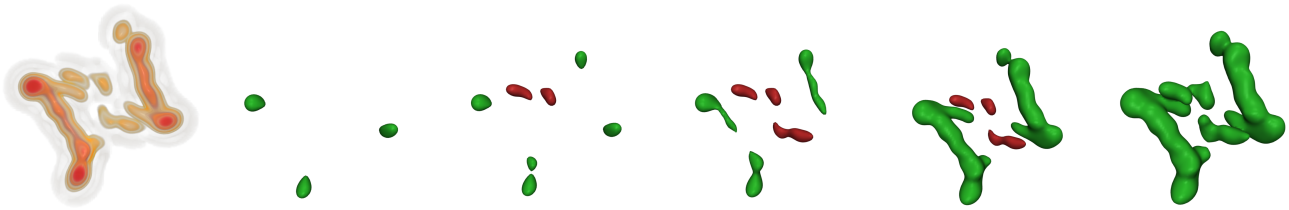
The segmentation of the domain utilized by our algorithm is induced by topological features identified through the contour tree. We compare our method with existing symmetry detection methods that also segment the domain on the basis of topological features. While the contour tree based method relies on structural similarity between the subtrees for symmetry identification [38], the extremum graph based method depends on distances between the extrema evaluated on an augmented version of the extremum graph [39].

Thomas and Natarajan [39] point out that noise in the data that destroy the repeating structure of the subtrees of the contour tree poses challenges in determining symmetry using the contour tree. They argue that their method using the extremum graph is better at handling noise and report symmetry detected at the largest scale for several cryo-EM datasets. The contour tree based method, on the other hand, can detect symmetry at different scales. Our method combines the advantage of these two methods. The first row in Fig. 9 shows the result of our method on one of the noisy datasets used by Thomas and Natarajan [39], EMDB-1654. In addition to detecting symmetry at the largest scale, our method is able to identify smaller symmetric regions at different scales. The supplemental material shows the projection of the descriptors onto 2D using multidimensional scaling and illustrates that the clusters representing symmetric regions are well separated and can be easily identified. In contrast, the extremum graph based method requires user guidance for carefully selecting a set of well separated and symmetrically distributed extrema, called the seed set, for symmetry detection. Typical cryo-EM datasets is devoid of noise in the high and low density regions and therefore the symmetry of the extrema belonging to these regions can be easily identified for choosing a seed set. Our method does not make such assumptions and we show the results of our method on two datasets where selection of seed set is non-trivial. Fig. 9(f) shows a volume rendering of a CT scan of a pair of knees. Observe the dark regions in the bone where the scalar values are noisy. Our method identifies the symmetry between the contours belonging to three portions of the bones in the left and the right knee, shown in orange, blue, and pink in Fig. 9(g). An inspection of these contours show that the dark regions of the bone in the volume ren-



(a) (b) (c) (d) (e)

Fig. 10. Influence of stabilization parameter, α, on symmetry detection. (a) Volume rendering of a cryo-EM dataset (EMDB-1292) shows long tube-like structures that exhibit 7-fold rotational symmetry. (b) Setting α to 0.2% of the length of the range of scalar values generates unstable contours within the tube-like regions. The inconsistency in the merging of the blue and green contours indicates this instability. (c) Setting α to 1% discards the unstable contours. Instead, the small blue contours are generated at a more stable isovalue. (d) The 7-fold symmetry of the tube-like regions are identified in both cases since these larger features are not sensitive to the exact value of α. (e) The plot of the number of critical points with increasing simplification shows a significant drop at a low value of α. We set α to a value immediately after this drop.



(a) (b) (c) (d) (e) (f)

Fig. 11. Influence of approximation parameter, ε, on symmetry detection. (a) Volume rendering of a cryo-EM dataset (EMDB-5214) with 2-fold symmetry where the left and the right regions show deviations from perfect symmetry. (b) The green oval contours that are almost perfectly symmetric are identified when $\varepsilon = 0.1\%$. (c) Increasing ε to 1% results in the addition of two more smaller contours of oval shape and a new set of symmetric contours shown in maroon. (d) At $\varepsilon = 4\%$ a new maroon contour, which is only partially symmetric with respect to the existing maroon contours, is located. Similarly, the green contours also deviate from perfect symmetry. (d) Larger green regions are detected at $\varepsilon = 8\%$ and (e) ε =10%. The occluded contours are not shown.

dering denote missing regions and noise in the contours. Despite these missing regions and noise, our method successfully identifies the symmetry of the bones. The isosurface for this dataset at the global maximum in Fig. 9(j) shows that the global maximum is degenerate and forms a flat region. Fig. 9(h) shows an electron density field derived from the atomic coordinates of a hemoglobin molecule using EMAN software [18]. Our method identifies two symmetric regions, shown in blue and pink in Fig. 9(i), despite these two regions being in close proximity to each other. The isosurface for an isovalue close to the global maximum, see Fig. 9(k), shows that the extrema belonging to the high density regions are not symmetrically distributed within the two regions. In both these cases, selection of a set of extrema as seed set is not an easy task because of the flat regions and the asymmetric distribution of the extrema.

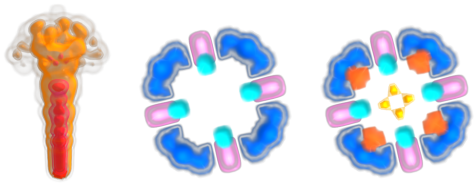
7.2 Parameter Selection

The approximation parameter ε and the noise parameter δ are external to our algorithm. Hence, the stability parameter α is the only parameter that is specific to our method.

Stabilisation parameter α. The value of the stabilization parameter affects the detection of features that are very transient with respect to the evolution of contours during a level set sweep. Choosing too small a value for the stabilization parameter may result in only partial detection of such unstable symmetric regions while choosing too high a value may result in ignoring these unstable regions in the symmetry analysis. We illustrate this with a cryo-EM dataset (EMDB-1292) with 7-fold rotational symmetry, see the volume rendering in Fig. 10(a). In an attempt to capture the symmetry of the small unstable regions within the long tube-like regions that form the 7-fold symmetry, we set α to 0.2% of the total range of scalar values. However, only four of the seven symmetric regions generate small-scale contours belonging to the tube-like region, shown in blue in Fig. 10(b). Due to the instability of the saddles, the contours belonging to the remaining three regions, for the same isovalue, have already merged, as shown by the green contours.

To determine a suitable value for α, we adopt the same heuristic used by Thomas and Natarajan to stabilize the branch decomposition representation of the contour tree [38]. We plot the drop in the number of critical points with increasing simplification of the contour tree, see Fig. 10(e). Initially, there is a sudden drop due to the unstable critical points, after which the curve tapers off. Based on this plot, we select a value immediately after the initial drop. Setting α to 1% discards the unstable isovalue generated earlier. Fig. 10(c) depicts the 7-fold symmetry of the small blue contours from the top portion of the tube-like regions identified using the new isovalue generated. The symmetry of the remaining small regions are not detected since the higher value of α discards the unstable contours in this region. Note that the exact value of α is crucial only while determining symmetry of the contours belonging to very small arcs of the contour tree. These contours are very transient and quickly merge or split into another contour during a level set sweep. For both values of α, the symmetry of the tube-like regions is detected correctly at a different isovalue that is more stable, see Fig. 10(d). This is because these contours belong to longer arcs and are insensitive to the exact value of α. The inability to detect symmetric regions due to unstable contours is a limitation of our implementation. This limitation arises from the constraint that for the same isovalue, contours should be consistent with respect to merging. A solution is to relax this constraint and allow a small interval of isovalues from which stable contours may be generated independently by employing a geometric criterion like maximally stable contour [20]. However, when contours at different isovalues are used to detect symmetry, the variation in the isovalues should also be considered in determining the quality of the symmetry. For this purpose, we are currently exploring ways to include the differences in the isovalues into the distance measure between contours in the descriptor space.

Approximation parameter ε. Changing the value of ε directly affects



(a) (b) (c)

Fig. 12. Influence of noise parameter, δ, on symmetry detection. (a) Volume rendering of the Fuel dataset, which exhibits 4-fold rotational symmetry at the top of the long tube-like region. (b) Three symmetric regions reported when δ is set to 200 vertices. (c) Lowering δ to 20 vertices results in the detection of two additional symmetric regions that are smaller.

the output of our method since ε controls the quality of the symmetry detected. We believe that it is best to let the user choose the value of ε for a dataset, on the basis of the output it generates, instead of determining a specific value through a heuristic procedure. Note that the first and the second stages of our algorithm, namely, generation of contours and computation of descriptors, are independent of the value of ε. Nearest neighbor search can be performed efficiently. Therefore, if the first and second stages of our method are precomputed, it is possible to efficiently regenerate the output as ε is varied. This makes it possible to provide the user with a slider interface to visualize the effect of varying ε and choose the ideal value for a dataset.

Fig. 11 illustrates the effect of varying ε on a cryo-EM dataset (EMDB-5214). Even though the global symmetry is evident, carefully observe the variation between the left and the right regions. These variations affect the quality of symmetry and is captured by different values of ε in our experiment. We determine the maximum distance between a pair of points in the descriptor space and specify different values of ε as a percentage of the maximum distance. For $\varepsilon = 0.1\%$, the oval contours that are highly symmetric, shown in Fig. 11(b), are detected. As the value of ε is increased, more approximately symmetric contours are detected as shown in Fig. 11(c)-11(f). In each figure, contours belonging to the same cluster are shown with the same color. Note that the same dataset was used by Thomas and Natarajan to emphasize the advantage of the extremum graph based method [39] over the contour tree based method [38]. They illustrate, that the extremum graph based method is able to detect the global symmetry by virtue of being geometry-aware, while the contour tree based method fails to do so due to variations in the features of the left and the right regions. Our method is able to quantify the effect of these variations on the output quality. Moreover, it can recognize partial symmetry at different scales in addition to identifying the global symmetry.

Noise parameter δ. We determine the number of arcs of the contour tree with volume above a particular value and plot the drop in the number of arcs with increasing value of volume. For the Fuel dataset in Fig. 12(a), this plot is shown in the supplemental material. The plot shows a sudden drop in the number of arcs initially, similar to the graph used for determining the stabilization parameter α. This drop is due to a large number of small sized features in the data which are considered to be noise. We choose a value immediately after this sudden drop. When δ is set to 200 vertices for the Fuel dataset, three sets of symmetric features, each exhibiting 4-fold rotational symmetry are identified, see Fig. 12(b). Lowering δ to 20 vertices identifies two more smaller symmetric regions as illustrated in Fig. 12(c). Thus, δ acts as a parameter that controls the size of features analyzed for multiscale symmetry detection. Setting δ to an even lower value generates contours of very small size whose descriptors cannot be reliably compared due to numerical errors. In comparison, the contour tree based method is able to easily capture the symmetry of smaller symmetric regions for this dataset. They use a different model for determining noise based on persistence, which is the difference in function value between the extremum and the saddle of a topological feature. Although our implementation may be modified to incorporate persistence based noise determination, it is not possible to use a single model that works across datasets with different noise characteristics. Therefore, we suggest that the model for determining noise be adapted based on the application under consideration.

Table 1. Running time, measured in seconds, for the different steps in the symmetry detection algorithm. All experiments were performed on a 2 GHz Intel Xeon processor with 8GB RAM. The time taken for computing contour tree is denoted by t_{ct}, generating contours by t_{cg}, and computing contour descriptors by t_{cd}. The number of contours processed is denoted by #ctrs and the number of critical points by #crit.

Dataset	#vertices	$t_{ct}(s)$	$t_{cg}(s)$	$t_{cd}(s)$	#ctrs	#crit
1HGA	$72\times72\times72$	1.3	7.7	28.7	78	4698
EMDB-1654	$112\times112\times112$	4.6	1.0	8.7	92	55762
Vortex	$253\times253\times253$	82.0	19.9	163.7	119	1184142
EMDB-1179	$255\times255\times255$	67.4	10.6	84.9	339	338688
Knee	$379\times229\times305$	120.4	94.6	429.1	442	1222491

7.3 Performance

Table 1 lists the running time of our algorithm on different datasets. The time required to compute the contour tree depends on the size of the dataset and the number of critical points present in it. The contour generation and the descriptor computation stages are primarily affected by the size and number of contours processed. The contour generation stage includes the time taken for the simplification of large meshes. In our implementation, symmetry with respect to contours that evolve from maxima and minima are identified in two separate passes and the running time listed above is for a single pass. The clustering stage is very quick and requires less than a second. The current implementation is not very efficient and offers a lot of scope for improvement. However, we believe that even with our suboptimal implementation, the running times that we report are reasonable for a geometry based algorithm that identifies symmetry at multiple scales. The symmetries detected in the Vortex and EMDB-1179 datasets are shown in the supplemental material.

8 Conclusions

In this paper, we present a novel symmetry detection method based on the idea of clustering contours. We show that mapping contours to points in a descriptor space is a powerful representation for performing similarity analysis on scalar fields. One of the main limitations of our method is that symmetry identification is restricted to regions obtained from the segmentation of the domain induced by the contour tree. We plan to extend our method to handle other kinds of segmentation by designing an appropriate surface representation for the segments. A related issue is that our method does not consider symmetry within a contour. A preprocessing step that segments contours into smaller partial surfaces may be designed to solve this problem. The current implementation does not incorporate geometric stability criteria for selecting representative contours. It is beneficial to employ such a criteria that selects a stable contour from each arc of the contour tree to ensure consistency in the shape of the contours extracted. We believe that the descriptor space representation of scalar fields will spur research in determining structurally similar features for applications like querying and tracking in time-varying and ensemble data.

Acknowledgments

This work was partially supported by the Department of Science and Technology, India, under Grant SR/S3/EECE/0086/2012 and by the Robert Bosch Centre for Cyber Physical Systems, Indian Institute of Science. The second author was supported by a fellowship for experienced researchers from the Alexander von Humboldt Foundation. Hurricane Isabel data was produced by the Weather Research and Forecast (WRF) model, courtesy of NCAR and the U.S. National Science Foundation (NSF). Volume rendered images used in the paper were generated using Voreen (www.voreen.org).

REFERENCES

[1] J. Beyer, A. Al-Awami, N. Kasthuri, J. W. Lichtman, H. Pfister, and M. Hadwiger. Connectomeexplorer: Query-guided visual analysis of large volumetric neuroscience data. *IEEE Trans. Vis. Comput. Graph.*, 19(12):2868–2877, 2013.

[2] M. Bokeloh, A. Berner, M. Wand, H. Seidel, and A. Schilling. Symmetry detection using feature lines. *Computer Graphics Forum*, 28(2):697–706, 2009.

[3] G. Boothroyd, P. Dewhurst, and W. Knight. *Product Design for Manufacture and Assembly, Third Edition*. CRC Press, 2010.

[4] S. Bruckner and T. Möller. Isosurface similarity maps. *Comput. Graph. Forum*, 29(3):773–782, 2010.

[5] H. Carr, J. Snoeyink, and M. van de Panne. Flexible isosurfaces: Simplifying and displaying scalar topology using the contour tree. *Comput. Geom.*, 43(1):42–58, 2010.

[6] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring error on simplified surfaces. *Comput. Graph. Forum*, 17(2):167–174, 1998.

[7] M. de Graef and M. E. McHenry. *Structure of Materials*. Cambridge University Press, 2007.

[8] M. Haidacher, S. Bruckner, and M. E. Gröller. Volume analysis using multimodal surface similarity. *IEEE Trans. Vis. Comput. Graph.*, 17(12):1969–1978, Oct. 2011.

[9] M. Hilaga, Y. Shinagawa, T. Komura, and T. L. Kunii. Topology matching for fully automatic similarity estimation of 3D shapes. In *SIGGRAPH*, pages 203–212, 2001.

[10] Y. Hong and H.-W. Shen. Parallel reflective symmetry transformation for volume data. *Computers & Graphics*, 32(1):41–54, 2008.

[11] H. Hoppe. New quadric metric for simplifying meshes with appearance attributes. In *IEEE Visualization*, pages 59–66, 1999.

[12] M. M. Kazhdan, B. Chazelle, D. P. Dobkin, T. A. Funkhouser, and S. Rusinkiewicz. A reflective symmetry descriptor for 3D models. *Algorithmica*, 38(1):201–225, 2003.

[13] J. Kerber, M. Wand, J. Krüger, and H. Seidel. Partial symmetry detection in volume data. In *Vision, Modeling, and Visualization*, pages 41–48, 2011.

[14] B. Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2013.

[15] Z. Lian, A. Godil, B. Bustos, M. Daoudi, J. Hermans, S. Kawamura, Y. Kurita, G. Lavoué, H. Van Nguyen, R. Ohbuchi, et al. A comparison of methods for non-rigid 3d shape retrieval. *Pattern Recognition*, 46(1):449–461, 2013.

[16] Y. Lipman, X. Chen, I. Daubechies, and T. Funkhouser. Symmetry factored embedding and distance. *ACM Transactions on Graphics (TOG)*, 29(4):103, 2010.

[17] Z. Liu, B. Jiang, and J. Heer. *imMens*: Real-time visual querying of big data. *Comput. Graph. Forum*, 32(3):421–430, 2013.

[18] S. J. Ludtke, P. R. Baldwin, and W. Chiu. Eman: semiautomated software for high-resolution single-particle reconstructions. *Journal of structural biology*, 128(1):82–97, 1999.

[19] T. B. Masood, D. M. Thomas, and V. Natarajan. Scalar field visualization via extraction of symmetric structures. *The Visual Computer*, 29(6-8):761–771, 2013.

[20] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image Vision Comput.*, 22(10):761–767, 2004.

[21] N. Mitra, M. Pauly, M. Wand, and D. Ceylan. Symmetry in 3D geometry: Extraction and applications. In *EUROGRAPHICS State-of-the-art Report*, pages 29–51, 2012.

[22] N. J. Mitra, L. J. Guibas, and M. Pauly. Partial and approximate symmetry detection for 3D geometry. *ACM Trans. Graph.*, 25:560–568, 2006.

[23] N. J. Mitra, L. J. Guibas, and M. Pauly. Symmetrization. *ACM Trans. Graph.*, 26(3):63, 2007.

[24] M. Niethammer, M. Reuter, F.-E. Wolter, S. Bouix, N. Peinecke, M.-S. Koo, and M. E. Shenton. Global medical shape analysis using the laplace-beltrami spectrum. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2007*, pages 850–857, 2007.

[25] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

[26] Z. Qin, J. Jia, and J. Qin. Content based 3d model retrieval: A survey. In *Content-Based Multimedia Indexing, 2008. CBMI 2008. International Workshop on*, pages 249–256. IEEE, 2008.

[27] D. Raviv, A. Bronstein, M. Bronstein, and R. Kimmel. Full and partial symmetries of non-rigid shapes. *International journal of computer vision*, 89(1):18–39, 2010.

[28] M. Reuter, F.-E. Wolter, and N. Peinecke. Laplace–beltrami spectra as shape-dna of surfaces and solids. *Computer-Aided Design*, 38(4):342–366, 2006.

[29] M. Reuter, F.-E. Wolter, M. Shenton, and M. Niethammer. Laplace–beltrami eigenvalues and topological features of eigenfunctions for statistical shape analysis. *Computer-Aided Design*, 41(10):739–755, 2009.

[30] S. Rosenberg. *The Laplacian on a Riemannian Manifold*. Cambridge University Press, 1997.

[31] D. Schneider, C. Heine, H. Carr, and G. Scheuermann. Interactive comparison of multifield scalar data based on largest contours. *Computer Aided Geometric Design*, 30(6):521–528, 2013.

[32] D. Schneider, A. Wiebel, H. Carr, M. Hlawitschka, and G. Scheuermann. Interactive comparison of scalar fields based on largest contours with applications to flow visualization. *IEEE Trans. Vis. Comput. Graph.*, 14(6):1475–1482, 2008.

[33] J. Schreiner, C. Scheiclegger, and C. Silva. High-quality extraction of isosurfaces from regular and irregular grids. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):1205–1212, Sept 2006.

[34] J. Schreiner, C. E. Scheidegger, S. Fleishman, and C. T. Silva. Direct (re)meshing for efficient surface processing. *Computer Graphics Forum*, 25(3):527–536, 2006.

[35] R. Stevenson and J. Hall. *Human Malformations And Related Anomalies*. Oxford University Press, 2006.

[36] D. Tahmoush and H. Samet. An improved asymmetry measure to detect breast cancer. In *Proc. SPIE 6514, Medical Imaging 2007: Computer-Aided Diagnosis*, 2007.

[37] J. W. Tangelder and R. C. Veltkamp. A survey of content based 3d shape retrieval methods. *Multimedia tools and applications*, 39(3):441–471, 2008.

[38] D. M. Thomas and V. Natarajan. Symmetry in scalar field topology. *IEEE Trans. Vis. Comput. Graph.*, 17(12):2035–2044, 2011.

[39] D. M. Thomas and V. Natarajan. Detecting symmetry in scalar fields using augmented extremum graphs. *IEEE Trans. Vis. Comput. Graph.*, 19(12):2663–2672, 2013.

[40] O. Van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or. A survey on shape correspondence. *Computer Graphics Forum*, 30(6):1681–1707, 2011.

[41] K. Xu, H. Zhang, W. Jiang, R. Dyer, Z.-Q. Cheng, L. Liu, and B. Chen. Multi-scale partial intrinsic symmetry detection. *ACM Trans. Graph.*, 31(6):181, 2012.

[42] K. Xu, H. Zhang, A. Tagliasacchi, L. Liu, G. Li, M. Meng, and Y. Xiong. Partial intrinsic reflectional symmetry of 3D shapes. *ACM Trans. Graph.*, 28(5), 2009.

[43] X. Zhang, C. L. Bajaj, B. Kwon, T. J. Dolinsky, J. E. Nielsen, and N. A. Baker. Application of new multi-resolution methods for the comparison of biomolecular electrostatic properties in the absence of global structural similarity. *SIAM J. Multiscale Modeling and Simulation*, 5:1196–1213, 2006.