

Semi-isometric registration of line features for flexible fitting of protein structures

S. Abeysinghe¹, M.L. Baker², W. Chiu², and T. Ju¹

¹Washington University in St. Louis, USA

²Baylor College of Medicine, USA

Abstract

In this paper, we study a registration problem that is motivated by a practical biology problem - fitting protein structures to low-resolution density maps. We consider registration between two sets of line features (e.g., helices in the proteins) that have undergone not a single, but multiple isometric transformations (e.g., hinge-motions). The problem is further complicated by the presence of symmetry in each set. We formulate the problem as a clique-finding problem in a product graph, and propose a heuristic solution that includes a fast clique-finding algorithm unique to the structure of this graph. When tested on a suite of real protein structures, the algorithm achieved high accuracy even for very large inputs containing hundreds of helices.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems I.4.7 [Image Processing and Computer Vision]: Feature Measurement—Invariants I.5.3 [Pattern Recognition]: Clustering—Similarity measures

1. Introduction

Registration is one of the fundamental problems in computer graphics and vision. Given two shapes that share similar features (e.g., a character in two different poses), registration seeks a correspondence between the two sets of features. Such correspondence could then be used for computing a deformation from one shape to another, or assessing the similarity of the two shapes. As a result, registration is important for a range of applications such as morphing and animation, model retrieval, and object understanding.

In this work, we consider a specific registration problem that is motivated by a real-world application in biology. The problem concerns the matching of two sets of line features (which are protein helices in this application) that have undergone multiple isometric deformations. We show that the problem is challenging to solve with existing registration techniques, and present a novel algorithm and demonstrate its effectiveness on a suite of real data.

1.1. Motivating application

Understanding protein structures in 3D is a primary goal of structural biology. Here, the 3D structure refers to the spa-

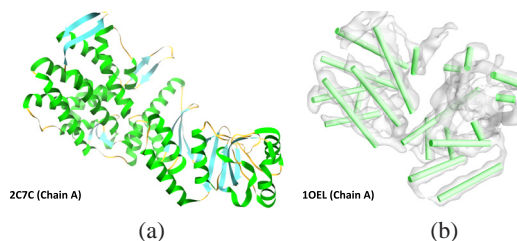


Figure 1: Protein fitting aims at deforming a protein structure (a) into a density volume (b) imaged at a low-resolution of a similar protein or the same protein at a different state. For validation purpose, the volume in (b) is simulated.

tial locations of the amino acids in the protein sequence. An example is shown in Figure 1 (a), where the molecule is rendered in a cartoon style that groups segments of amino acids into three types of structure components: α -helices (green spirals), β -strands (blue arrows) and loops (yellow wires). Traditionally, protein structures are obtained using high-resolution imaging methods like X-ray crystallography, which are suited for proteins that are small in size and isolated from their native environments. More recently, emerging techniques like cryo electron microscopy [CBJ*05] allow for imaging large protein complexes, such as viruses,

and proteins in their native conformations. On the downside, these newer techniques generate 3D volumes that only roughly depict the electron density around the protein and do not have sufficient resolution for locating individual amino acids. An example is shown in Figure 1 (b), which shows an iso-surface of the density volume in transparency.

To recover the 3D structure of some protein X from a low-resolution density map, a commonly used method involves deforming a known 3D structure of X at a different conformation, obtained for example by X-ray crystallography, to fit the density map. Alternatively, if the 3D structure is available for another protein Y that has a similar amino acid sequence as X , fitting the structure of Y (e.g., Figure 1 (a)) to the density map of X (e.g., Figure 1 (b)) would give biologists a good starting point to reconstruct the structure of X .

A major challenge to existing structure fitting methods is the often large, non-local shape difference between the structure and the target density map. This is usually caused by proteins undergoing “hinge-like” motions as they bend at one or multiple locations (e.g., a bend in the middle can be noted between the density and the structure in Figure 1). Existing fitting methods [TLW*08, TVM*08, WB01] typically start by a rigid-body alignment followed by local energy minimization. When the density map differs from the structure to be fitted by non-rigid, hinge-like motions, rigid-body alignment would not give a good starting point needed by the energy minimization step to converge or to converge efficiently.

Collaborating with a group of biologists, our longer-term goal is to develop efficient and robust structure fitting methods that better handle such hinge-like shape changes. A key question that needs to be addressed for our goal, which is the focus of this current work, is identifying where a part of the structure should move to in the density volume. For this purpose, we consider the problem of matching α -helices on the protein structure (e.g., green spirals in Figure 1 (a)) to identified helix locations in the density volume [BJC07] (e.g., green cylinders in Figure 1 (b)). Such matching would serve as anchors that can be utilized in a subsequent registration algorithm that deforms the protein structure to fit in the density volume, which we will explore in our future work.

There are a number of reasons for using α -helices as “anchors” for registration. First, they are fairly stable signatures of a protein, and tend to stay rigid in conformational changes. Second, the set of helices in a protein usually covers well the domain of the protein, hence their correspondence will be a good guide for subsequent structure fitting. Finally, they are of a small quantity (typically from tens to hundreds), making efficient computations possible.

1.2. Problem statement

The helix-matching problem mentioned above can be formulated as a non-rigid registration between two sets of line

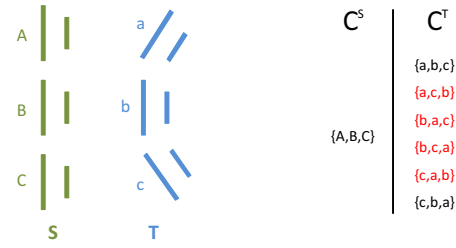


Figure 2: Two sets of features with symmetric subunits (left), and several ways these subunits can be matched (right). The (red) highlighted matchings are less plausible as they map neighboring subunits in one set to distant ones in the other.

features (the radius of α -helices are constant and hence can be ignored). Motivated by the hinge-like motions, we wish to identify *clusters* within each set such that corresponding clusters in the two sets can be mapped to each other using *isometric* (or *congruent*) transformations. Isometric transformations are more general than rigid-body ones, and additionally include reflections (by a plane or by a point), which can happen to proteins in nature [PY03]. We call this problem a *semi-isometric registration* problem.

More specifically, consider two sets of line features S, T that may have different cardinalities. The difference could be attributed to errors in feature detection, loss or gain in features, or mapping a smaller shape to a much larger shape (or vice versa). Our goal is to identify a set of feature clusters $C^S = \{C_1^S, \dots, C_k^S\}$ in S and another set $C^T = \{C_1^T, \dots, C_k^T\}$ in T with the following properties:

- Isometric:** For any $i \in [1, k]$, corresponding clusters C_i^S, C_i^T have the same cardinality, and there is an isometric transformation M_i such that $M_i(C_i^S)$ approximately matches C_i^T up to some user-specified tolerance.
- Disjoint:** For any $i \neq j \in [1, k]$, $C_i^S \cap C_j^S = \emptyset$ and $C_i^T \cap C_j^T = \emptyset$.
- Maximal:** Each cluster in C^S or C^T cannot be expanded, while satisfying (1,2), by either merging with other clusters or adding additional features. Also, no additional clusters can be added to C^S or C^T while satisfying (1,2).

Note that cluster sets C^S, C^T satisfying the above properties may not be unique. For example, if S or T contains *symmetric* parts, a group of features in S may be isometrically mapped to multiple groups in T . This is illustrated in the example in Figure 2, where both S and T consist of three symmetric subunits (A, B, C and a, b, c respectively), each can be considered as a cluster. There are 6 different ways in which these clusters can be mapped that all satisfy the above properties (shown on the right). However, some mappings (colored red) are obviously less plausible than others, as they map neighboring clusters in S to far-away clusters in T (or vice versa).

The symmetry scenario can be common in our motivating application, since a protein complex may contain a number of symmetrically arranged sub-complexes. With this scenario in mind, we wish to find the matching cluster sets C^S, C^T that best maintain the *spatial coherence* of the clusters in S and T . That is, if clusters C_i^S, C_j^S are located nearby, so should be C_i^T, C_j^T , and vice versa, for any pair $\{i, j\}$.

The semi-isometric nature of our registration problem is challenging for existing techniques (see review in Section 2). While a few methods are capable for identifying multiple isometric transformations for registration, the presence of symmetry creates additional challenges that leave room for improvement. In addition, matching of line features has been rarely discussed in the literature.

1.3. Method

We propose a graph-based method for the presented registration problem. To identify the matching clusters, our key observation is that a pair of isometric clusters, one in S and the other in T , can be represented as a *clique* in an appropriately constructed product graph. In this graph, each vertex represents a pair of line features with matching length, one in S and the other in T , and each edge represents a pair of features in S that can be isometrically mapped to another pair in T within some user-specified tolerance. This graph formulation will be detailed in Section 3.

Based on the observation, we derive a greedy heuristic to search for the desirable matching clusters as a set of maximal cliques in this graph, which represent disjoint clusters in S and T and which maximize the spatial coherence among these clusters. The search involves a best-first tree expansion, and uses a fast approximated clique-searching algorithm enabled by the specific structure of the graph. The details of the algorithm will be explained in Section 4.

1.4. Contributions

The contributions of this work to computer graphics are two fold. First, we give a graph-based formulation for mapping line features. The formulation represents each isometry as a clique in a product graph, and we show that such cliques can be found efficiently using a triangle-based search. Second, we propose a heuristic solution to semi-isometric mapping in the presence of symmetric components. We anticipate that both the graph formulation and the heuristic can be applied to other registration problems, beyond our motivating application, that involve lines features and/or multiple isometric transformations.

This work also makes the first step towards a more robust and efficient way for fitting protein structures into low-resolution density maps, the completion of which will greatly benefit structural biologists in their pursuit of understanding protein structures.

2. Previous work

Computing the registration between rigid or isomorphic sets of features has been extensively researched in the past. The majority of this work first identifies two sets of annotated landmark features by using local shape descriptors such as *SIFT* [Low99], *local spherical harmonics* [FS06], *salient surface features* [GCO06], *curvature maps* [GZGG05], *spin images* [JH99], *geometric hashing* [WR97], etc. Thereafter, *Iterative closest point (ICP)* [BM92] or statistical analysis methods such as *RANSAC* [FB81, RFP08] are used to find the registration as the lowest-error alignment of the feature sets. These methods are very efficient [AMCO08, HQS*09], but are only suited when there is a significant amount of rigid overlap between the two shapes.

At the other end of the spectrum, methods have been developed for computing registration between fully flexible sets of features. Spectral methods such as that of Jian and Zhang [JZ07] embed both shapes to a high-dimensional space where correspondences can be more easily computed. Zheng and Doermann [ZD06] formulate matching as a combinatoric optimization problem guided by the principle of retaining local spatial relationships. While being general and producing robust registrations, these methods can be computationally expensive as they explore a much larger space of possible deformations than rigid motions. More specific methods have been developed for matching particular types of features, for example, deformed triangular meshes [ZSCO*08], feature points on surfaces [LF09], and skeletons [ATCO*10].

There has been few methods that address the problem of semi-isometric registration, or computing a mapping made up of multiple isometric transformations. Most notably, Wang and Guibas [WG06] consider a similar biologically motivated problem, matching between two molecular surfaces extracted from two low-dimensional density maps that correspond to a same protein in two different conformations. Both our algorithm and theirs share a common overall approach, which successfully identifies the largest unmatched parts of each shape that can be rigidly mapped. However, the algorithm in [WG06] is limited to detecting two rigid body transformations, and does not consider the presence of symmetry. Also, a different type of feature is considered in their algorithm (based on a surface elevation function).

3. Graph formulation

As mentioned earlier, our method is based on a graph formulation of isometry between line features. We will first explain the construction of the graph, and then establish the relation between two isometrically aligned feature clusters and a clique in this graph.

3.1. Graph construction

We consider a *product graph* that represents the associative relation between the two sets of line features S and T . Vertices in this graph represent a pair of features, one from S and one from T , that have similar lengths. An edge between two vertices indicates a likely isometric mapping between two features in S and two features in T . To facilitate subsequent analysis, we consider each line to be represented by two *oriented line segments* (OLSs) that have the same length as the line feature but are pointing in opposite directions. As we will show in the next section, with this construction, a clique in the graph represents two clusters in S and T associated with an approximate isometry. More specifically:

Vertex construction: For each pair of OLSs $\vec{e} \in S, \vec{f} \in T$, we can create a vertex in the graph if $\|L(\vec{e}) - L(\vec{f})\| \leq \epsilon_l$, where L indicates length, and ϵ_l is a user-chosen threshold. This construction yields four vertices for each pair of features. However only two of these vertices define a unique (direction specific) registration between this feature pair, and therefore only these two vertices are added to the graph.

Edge construction: To assess how well a pair of OLSs in S can be mapped isometrically to another pair in T , we first introduce a descriptor for a pair of OLSs that is invariant to isometric transforms. This descriptor, $R(\vec{e}_1, \vec{e}_2) = \{d, \alpha, \beta, \theta\}$ consists of four scalars measuring various distances and angles within a pair of OLSs $\{\vec{e}_1, \vec{e}_2\}$, as illustrated in Figure 3. In particular: d is the Euclidean distance between the midpoints p_1, p_2 respectively of \vec{e}_1, \vec{e}_2 . α and β are the angles respectively spanned by vector pairs $\{\vec{e}_1, (p_2 - p_1)\}$ and $\{\vec{e}_2, (p_1 - p_2)\}$, and θ is the angle spanned by vector pair $\{\vec{e}_1, \vec{e}_2\}$. It is easy to see that R is unaffected under any isometric transformation of $\{\vec{e}_1, \vec{e}_2\}$. The rationale behind this choice of metrics is that it is intuitive for users to specify error tolerances in registration, based on their understanding of how much the features could have moved or rotated relative to each other in the different shapes.

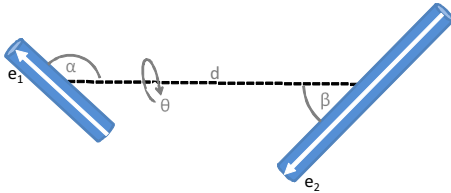


Figure 3: The $R(\vec{e}_1, \vec{e}_2)$ descriptor for the oriented line segments \vec{e}_1 and \vec{e}_2 consists of the distance between their centroids d , and the angles α , β and θ .

Given two graph vertices representing OLSs $\{\vec{e}_1, \vec{f}_1\}$ and $\{\vec{e}_2, \vec{f}_2\}$, where $\vec{e}_1, \vec{e}_2 \in S$ and $\vec{f}_1, \vec{f}_2 \in T$, we connect the two vertices by an edge if \vec{e}_1, \vec{e}_2 (and \vec{f}_1, \vec{f}_2) are not representing the same feature in S (and T), and if $\|R(\vec{e}_1, \vec{e}_2) - R(\vec{f}_1, \vec{f}_2)\| \leq \{\epsilon_d, \epsilon_a, \epsilon_b, \epsilon_\theta\}$, where $\epsilon_d, \epsilon_a, \epsilon_b, \epsilon_\theta$ are user-specified distance and angular tolerances.

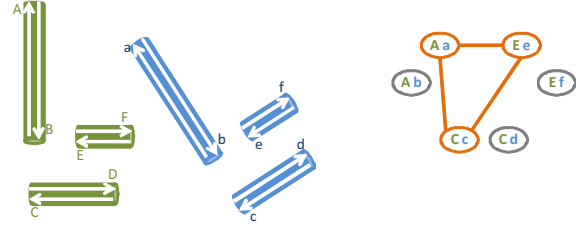


Figure 4: The zero-tolerance product graph G_0 (right) constructed for two sets of line features S, T (left), where each feature is represented by two oriented line segments (OLSs) indicated by arrows and labeled by letters. A clique (highlighted in orange) in the graph corresponds to an isometric mapping from a cluster of OLSs in S to another cluster in T .

The graph: We denote the graph constructed this way as G_ϵ , where $\epsilon = \{\epsilon_l, \epsilon_d, \epsilon_a, \epsilon_\theta\}$ denotes the various error tolerances. In particular, we use G_0 as a shorthand to denote the graph constructed with zero tolerances $\epsilon = \{0, 0, 0, 0\}$. In this graph, the vertices and edges represent features and feature pairs between S and T that are mapped by exact isometries. An example of the graph G_0 is shown in Figure 4 (right). In our experiments we use $\epsilon = \{5, 5, 0.3, 0.3\}$.

3.2. Isometric clusters as cliques

Based on the above construction, we can show that two feature clusters in S and T that are mapped by an exact isometry is equivalently represented by a clique in the zero-tolerance graph G_0 , and vice versa. This is the key observation that motivates our clique-based search algorithm (discussed in the next section), which identifies nearly isometric clusters as cliques in the non-zero-tolerance product graph G_ϵ . Next we will prove the equivalence claim in each direction.

Proposition 1 Let $\{\vec{e}_1, \dots, \vec{e}_n\} \subseteq S$ and $\{\vec{f}_1, \dots, \vec{f}_n\} \subseteq T$ be two clusters of OLSs such that no two \vec{e}_i, \vec{e}_j (or \vec{f}_i, \vec{f}_j) represent a same feature for any $i \neq j \in [1, n]$. If the two clusters can be exactly aligned using an isometric transformation, then there is a clique $\{v_1, \dots, v_n\} \subseteq G_0$ where each vertex v_i ($i \in [1, n]$) represents the pair $\{\vec{e}_i, \vec{f}_i\}$ (or $\{-\vec{e}_i, -\vec{f}_i\}$).

Proof: By isometry, $L(\vec{e}_i) = L(\vec{f}_i)$ for any $i \in [1, n]$, hence vertices v_i exist in the graph G_0 . Also by isometry, $R(\vec{e}_i, \vec{e}_j) = R(\vec{f}_i, \vec{f}_j)$ for any $i \neq j \in [1, n]$, thus each pair of vertices v_i, v_j is connected by an edge. Hence the vertices $\{v_1, \dots, v_n\}$ form a clique in G_0 . \square

Proposition 2 Consider a clique $\{v_1, \dots, v_n\} \subseteq G_0$, and denote the two OLSs represented by each vertex v_i ($i \in [1, n]$) as $\{\vec{e}_i, \vec{f}_i\}$. Then the clusters $\{\vec{e}_1, \dots, \vec{e}_n\} \subseteq S$ and $\{\vec{f}_1, \dots, \vec{f}_n\} \subseteq T$ can be exactly aligned using an isometric transformation, and no two \vec{e}_i, \vec{e}_j (or \vec{f}_i, \vec{f}_j) represent a same feature for any $i \neq j \in [1, n]$.

Proof: Being a clique, it follows that $L(\vec{e}_i) = L(\vec{f}_i)$ for all $i \in [1, n]$ and $R(\vec{e}_i, \vec{e}_j) = R(\vec{f}_i, \vec{f}_j)$ for all pairs $i \neq j \in [1, n]$.

By definition of L, R , these relations imply that the six pair-wise distances between the four end-points of any two OLSs \vec{e}_i, \vec{e}_j ($i \neq j \in [1, n]$) are the same as those of in \vec{f}_i, \vec{f}_j . It can be shown using an inductive approach that an isometric transformation exists between two point sets where all pair-wise distances are preserved. Hence the end points in the cluster $\{\vec{e}_1, \dots, \vec{e}_n\}$ can be mapped to those in $\{\vec{f}_1, \dots, \vec{f}_n\}$ by some isometry M . Also, since an isometry is linear, it maps straight lines to straight lines. Hence the same isometry M also maps the OLSs between those end points. \square

As an example, the highlighted triangle (a 3-clique) in the graph on the right of Figure 4 represents an isometric transform from the OLSs A,C,F in S to a,c,f in T . Note that there may exist many cliques in a product graph when matching a large number of line features. Next, we will describe our algorithm for identifying cliques that represent disjoint matching clusters, while additionally considering the problem of symmetry (as shown in Figure 2).

4. The algorithm

4.1. Overview

Recall from our problem statement (Section 1.2) that our goal is to identify a set of isometric pairs of feature clusters in S and T that are disjoint, maximal, and preserving the spatial coherence among neighboring clusters in S or T . Now that we have formulated a isometric cluster pair as a clique in the product graph G_ϵ , the task becomes searching for a set of *maximal* cliques that represent disjoint, spatially coherent cluster pairs. This is a hard combinatoric optimization problem, as the number of maximal cliques in a graph can be very large (exponential to the size of the graph in general), not to mention the number of different combinations of these maximal cliques. Here we resort to a greedy heuristic that finds a locally optimal solution, which we observed to give reasonable matching results in our tests.

Our algorithm is a greedy, best-first tree search. A node in the search tree consists of a set of maximal cliques $Q = \{q_1, \dots, q_k\} \subseteq G_\epsilon$ that represent two disjoint, isometrically matched sets of feature clusters $C^S = \{C_1^S, \dots, C_k^S\}, C^T = \{C_1^T, \dots, C_k^T\}$ respectively in S, T . A *cost function* $H(Q)$ is used to assess how well the neighborhood relation among clusters in C^S is preserved among their counterparts in C^T . The lower the $H(Q)$, the more coherent is the matching between C^S and C^T .

At the root of the tree, we create one child node for the largest clique q in G_ϵ and more child nodes for each of its *symmetric* cliques. A clique q' is symmetric to q if they map a similar cluster of features in S (or T) to different clusters in T (or S). At each iteration of the algorithm, we pick the tree node representing cliques Q with the lowest cost $H(Q)$, and consider the residue graph of G_ϵ that consists of only vertices (and their edges) representing features that have not appeared in Q . We then expand multiple children nodes, each

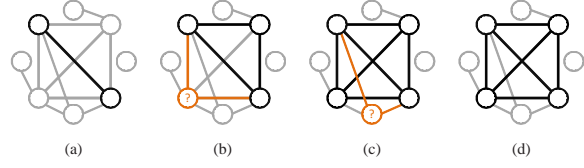


Figure 5: Our triangle-based search finds the largest clique in our product graph by first finding the highest valence edge (darkened in (a)), and for each vertex that forms a triangle with this edge (“?” in (b)) we add it to the clique if it shares edges with the vertices already in the clique, and discard it otherwise (“?” in (c)). The final clique is shown in (d).

adding to Q either the largest clique q in the residue graph or a symmetric clique of q . The search terminates when, at the point of expansion, no more cliques (containing at least two vertices) are found, and the cliques represented by the to-be-expanded node are output as the matching.

In practice, we have observed that the most time-consuming part of the search is expanding size-1 cliques (e.g., individual vertices in the residue graph) near the bottom level of the search tree. This is because symmetric 1-cliques are typically much more abundant than larger cliques (which represent symmetry involving larger clusters), thus significantly increasing the branching factor. Hence, in practice, we terminate the search when, at the point of expansion, the residue graph does not contain cliques of size larger than 1. The remaining 1-cliques are identified using a simpler, even more greedy voting procedure instead of the best-first tree search, guided by the same spatial-coherence principle.

In the next sections, we will detail the components of this algorithm. In particular, we will present a fast approximate algorithm for finding the largest clique based on the specific structure of our graph. We will then discuss finding symmetric cliques, our cost function, and the voting procedure for identifying smaller cliques.

4.2. Finding largest cliques

The key operation in our algorithm is identifying the largest clique in G_ϵ or its subgraph (e.g., residue graph). The problem of finding the largest clique in a general graph is known to be NP-complete and hard to approximate. The best algorithm runs in $O(2^{0.249n}) = O(1.1888^n)$ where n is the size of the graph [Rob01]. However, due to the special structure of cliques in G_ϵ (which represent isometry), we can use a much simpler, polynomial-time algorithm to find an approximate solution. This algorithm runs in $O(m^2)$ where m is the number of edges in G_ϵ , and is guaranteed to find the largest clique in the special case of a zero-tolerance graph G_0 .

Our algorithm performs a *triangle-based search*. For each edge in the graph, we compute its “valence” as the number of triangles in the graph that contains the edge. If the graph is void of edges, an arbitrary vertex is output as the largest

clique. Otherwise, as shown in Figure 5 we take the edge with the greatest valence as our initial clique, and visit the third vertex in each triangle containing the edge, each time adding the vertex to the clique if the vertex is also connected to all existing vertices in the clique. The complexity of the algorithm is dominated by the valence computation, which we do by taking the union of the adjacency list at each vertex of the edge. A naïve implementation of this union for each edge uses time linear to the total number of out-going edges at the two end-vertices of the edge, hence the total time is bounded by $O(m^2)$.

We next show that the algorithm indeed returns the largest clique when running on the zero-tolerance product graph G_0 . We will start with a few lemmas that will lead to this claim.

Lemma 1 Let v_1, v_2 be two vertices in G_0 connected by an edge and representing OLSs $\{\vec{e}_1, \vec{f}_1\}, \{\vec{e}_2, \vec{f}_2\}$. If \vec{e}_1, \vec{e}_2 are non-coplanar, then there is a unique isometric transform from $\{\vec{e}_1, \vec{e}_2\}$ to $\{\vec{f}_1, \vec{f}_2\}$.

Proof: By Proposition 2, an isometry exists between $\{\vec{e}_1, \vec{e}_2\}$ and $\{\vec{f}_1, \vec{f}_2\}$. Such isometric transform is unique, because there is a unique isometric transform (if such transform exists) between a pair of four non-coplanar points in 3D. \square

Lemma 2 Let v_1, v_2 be two vertices in G_0 connected by an edge and representing OLSs $\{\vec{e}_1, \vec{f}_1\}, \{\vec{e}_2, \vec{f}_2\}$, and $\{u_1, \dots, u_l\}$ be vertices connected to both v_1, v_2 and representing OLSs $\{\vec{g}_i, \vec{h}_i\}$ ($i \in [1, l]$). If \vec{e}_1, \vec{e}_2 are non-coplanar, then $\{v_1, v_2, u_1, \dots, u_l\}$ is a clique.

Proof: Since each triple $\{v_1, v_2, u_i\}$ ($i \in [1, l]$) forms a clique, by Proposition 2, there is an isometric transform, denoted as M_i , between $\{\vec{e}_1, \vec{e}_2, \vec{g}_i\}$ and $\{\vec{f}_1, \vec{f}_2, \vec{h}_i\}$. Following Lemma 1, all such M_i for $i \in [1, l]$ must be identical as they all involve mapping from $\{\vec{e}_1, \vec{e}_2\}$ to $\{\vec{f}_1, \vec{f}_2\}$. By Proposition 1, $\{v_1, v_2, u_1, \dots, u_l\}$ forms a clique. \square

Based on Lemma 2, the valence of an edge in G_0 indicates the size of the largest clique containing the edge. As a result, the edge with the greatest valence is contained in the largest clique, which will be found by our triangle-based search algorithm.

4.3. Finding symmetric cliques

In the presence of symmetry in S or T , a cluster in one model can be equally well matched to different clusters in another model. The largest clique we found in the product graph G_E only finds one matching that involves the largest number of features, and this matching may not be the optimal one in terms of maintaining spatial coherence (as seen in Figure 2). As a result, in addition to finding the largest clique, we will explore symmetric cliques that map the same cluster in S (or T) to different clusters in T (or S).

Given the largest clique q found in some residue graph G that represents an isometric mapping from a cluster $E =$

$\{\vec{e}_1, \dots, \vec{e}_l\} \subseteq S$ to another cluster $F = \{\vec{f}_1, \dots, \vec{f}_l\} \subseteq T$, we use an iterative procedure to find other cliques in G that represent mapping from E to different clusters in T (and similarly for cliques representing mapping from different clusters in S to F). Starting with an empty set of symmetric cliques $W = \emptyset$, we proceed as follows:

- Step 1: Consider the subgraph G' of G consisting of only those vertices (and their edges) representing two OLSs $\{\vec{e}, \vec{f}\}$ such that $\vec{e} \in E$ and $\vec{f} \notin F$. Find the largest clique q' in G' that represents a mapping between clusters $E' \subseteq E$ and $F' \subseteq T \setminus F$.
- Step 2: If the ratio of the size of q' over that of q is smaller than a user-chosen threshold, stop and output W . Otherwise, add q' to W , update $F = F \cup F'$ and repeat from Step 1.

4.4. Cost function

The cost function in our best-first tree search is used for measuring the spatial non-coherence among the isometric mappings represented by a set of cliques. We first introduce a pair-wise cost function between two cliques $p, q \subseteq G_E$:

$$h(p, q) = \min_{v \in p, u \in q} d(v, u)$$

Here, v, u are two vertices in the corresponding clique. Denote the OLSs represented by them as $\{\vec{e}_v, \vec{f}_v\}$ and $\{\vec{e}_u, \vec{f}_u\}$, $d(v, u)$ evaluates $|d_1 - d_2|$ where d_1 and d_2 are respectively the distance between the midpoints of \vec{e}_v, \vec{e}_u and \vec{f}_v, \vec{f}_u . Intuitively, h measures how far two clusters in S (or T) have been torn apart when transformed to T (or S) using the transformations represented by p, q .

To make our tree search better approximate the optimal solution, we would like the cost function associated with the tree nodes to be non-decreasing as the nodes are expanded. To achieve this effect, we consider the cliques at each tree node as an ordered set $Q = \{q_1, \dots, q_k\}$ where newly added cliques are appended to the end, and the cost function as

$$H(Q) = \sum_{i=2}^k w(q_i) \min_{j < i} h(q_i, q_j) \quad (1)$$

where the weight function $w(q) = 1/(1 - e^{-Size(q)/\lambda})$ downgrades the penalty for larger cliques. This is because larger cliques represent isometric mapping between greater number of features, and are less susceptible to noise. Hence we can place a higher confidence on them. For our experiments we found a value of $\lambda = 10$ to give good results.

4.5. Finding single feature registrations using a cost matrix

As mentioned earlier, we stop the best-first search when only 1-cliques remain in the residue graph, due to the high branching factor (and hence computational cost) of node expansion with 1-cliques. To find the remaining 1-cliques,

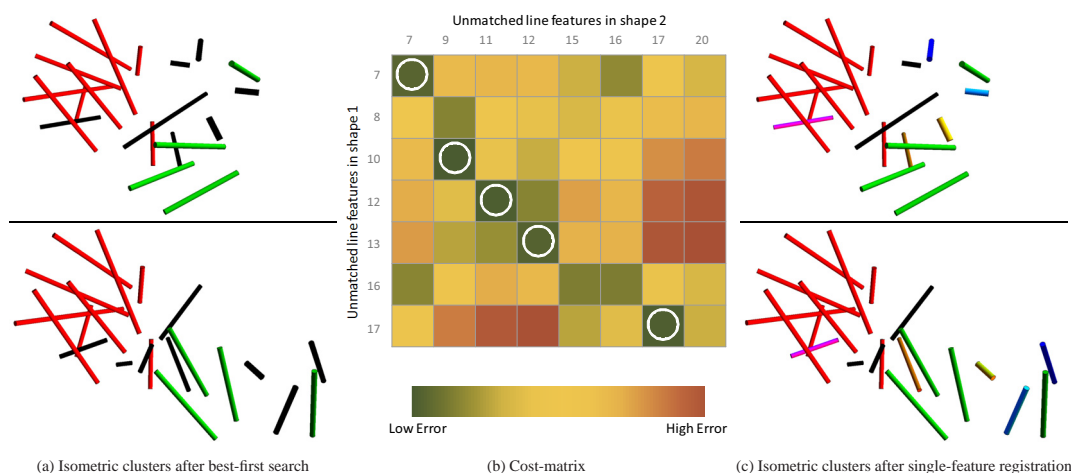


Figure 6: Isometric clusters identified after the best-first search (a) (different clusters are colored differently, and black lines are un-assigned), the cost matrix for the single-feature registration (selected feature pairs have been highlighted with white circles) (b), and the final set of isometric clusters after the single-feature matching step.

which represent matching between individual features in S and T that do not belong to any larger isometric clusters, we resort to a greedy voting procedure guided by the same spatial coherence principle that we used to find larger clusters.

We construct a two dimensional cost matrix whose rows and columns are respectively the un-matched features in S and T after the best-first tree search. Each entry in the matrix records the spatial non-coherence of matching a feature x in S and a feature y in T with respect to the already matched clusters. More specifically, denote the cliques found by tree search as Q , and the two graph vertices representing the matching between OLSs for x and OLSs for y as v_1, v_2 , the entry of the matrix is set to be

$$\min_{i \in \{1,2\}} H(Q \cup \{v_i\})$$

where H is the cost function in Equation 1. If none of these vertices exist in the graph, the cost is set to be ∞ . Figure 6(b) shows such a cost matrix. Given this matrix, we match up the features that give the lowest cost in the matrix, eliminate the corresponding row and column from the matrix, and repeat this process until all features in S or T are assigned, or a maximum cost threshold has been exceeded.

5. Results

We show here several results of our algorithm on matching α -helices. For the purpose of validation, each of our test will be matching between known structures of two proteins that have a similar amino acid sequence, so that we can obtain the ground truth correspondence of the α -helices based on comparing the two sequences. Note that the same algorithm can be equally applied to match a structure to a low-resolution density map with identified helices, which is our intended application. All protein structures used are obtained

from the Protein Data Bank [BHN03]. We selected a suite of structures whose sizes range from small (< 10 helices) to large (> 300 helices), which is typical for those imaged by low-resolution methods such as cryo electron microscopy. In practice, the number of helices in such density volumes rarely goes beyond 500.

We found that a common set of parameters work well in all our tests, i.e., $\epsilon = \{5, 5, 0.3, 0.3\}$ for graph construction and $\lambda = 10$ in the cost function of Equation 1. In practice, the user can alter these parameters based on their knowledge of the data. For example, if the two proteins differ by less rigid transformations in its components, the tolerances ϵ in graph construction can be suitably relaxed. A more ideal mechanism would be to let the algorithm automatically pick the parameters based on the data, which would be interesting to explore in the future. A related approach that could potentially offer benefits in this direction is by Li et al. [LLM08].

Figure 7 shows the registration result on the example in Figure 1, where matching clusters are differentiated by colors in (a,b). In this example, 18 of the 20 correspondences in the ground truth are correctly found by our algorithm. The unmatched helices (colored black) are due to a break-up helix (the long one in (a) broken into two in (b)), and small helices that only appear in one of the structures. Figure 7 (c,d) show the result after transforming the structure of (a) using respectively the isometry represented by the two (red and green) cluster pairs. Note that there is no single isometry that captures well the difference between the two structures, supporting the need for solving the registration in a semi-isometric manner.

A more complex example is shown in Figure 8, where both structures (a,b) contain over 50 helices and a number of identical sub-units (five aligned in a circle and one hanging

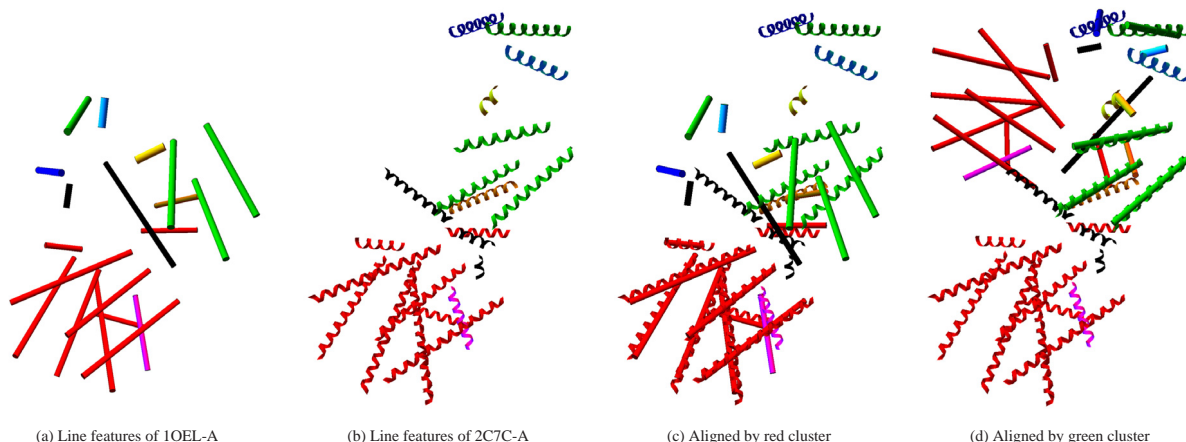


Figure 7: Line features in the example of Figure 1, shown as cylinders (a) and coils (b), are colored by the isometric clusters identified using our method (black features and unassigned). (c,d) show the alignment of the two structures respectively using the isometry of red and green cluster pairs.

out). In this example, our method correctly identified 55 out of the 57 correspondences in the ground truth, even in the presence of symmetry. Note that even though there are some helices missing in one structure and some broken-up helices (as in closed-up thumbnails), the matching of the remaining helices is not affected, demonstrating the robustness of our algorithm.

With the ability to deal with symmetry, our method can be used to register a smaller protein unit to a much larger structure. This feature is particularly important in our application setting: oftentimes the structures are only known for a small protein unit, while the knowledge of an entire complex can only be obtained using low-resolution imaging methods. Figure 9 shows one such example: a short protein chain (1OEL-A) is registered onto a large complex containing multiple similar chains (2C7C). Note that without considering spatial-coherence (e.g., setting $H(Q) = 0$ in Equation 1), different parts of the chain will be registered to isolated regions on the target complex (shown in Figure 9 (a)), while the use of spatial-coherence enforces the integrity of the chain (shown in (b,c)). Running registration for a number of times allows one to segment the complex into individual chains (shown in (d)), where the color now differentiates between chains). More such segmentation results are shown in Figure 10.

Table 1 reports the statistics of correspondences for a suite of test cases and the performance of our method (running on a 3 GHz Pentium D CPU utilizing only one core and 4GB memory). Note that our algorithm performs well for the majority of the data, finding over 90% of the helix correspondences in the ground truth. In all tests, over 95% of the found correspondences are correct. The algorithm is quite efficient, finishing under a few minutes for all tested structures and under seconds for small to medium sized structures. The most time-consuming part is graph construction, which currently

has a complexity of $O(n^4)$ where n is the number of line features in one shape.

We are currently developing an interactive interface for inspecting and refining the matching result. Initial feedback from biologists who test-ran the prototype of the interface indicated their preference of the automated matching algorithm over a completely manual matching process, where the user has to juxtapose two sets of line features in 3D and select corresponding pairs one by one. In addition, since the majority of the correspondences found by our algorithm are correct and most errors take place in smaller clusters, we observed that it is not difficult to visually spot regions of the structure that may contain erroneous correspondences produced by the algorithm, hence directing manual corrections only to isolated locations. Together with the matching algorithm, we plan to distribute the interface freely as part of a molecular modeling software, *Gorgon*, at <http://gorgon.wustl.edu/>.

6. Conclusion and discussion

In this paper, we considered the registration between two sets of line features that undergo multiple isometric deformations. The study was motivated by a biology application for fitting protein structures. We proposed a graph-based formulation of the registration problem, and an efficient heuristic for solving the problem, with a particular emphasis on handling symmetric components. The algorithm was tested on a suite of real protein structures and was observed to achieve accurate results even for large inputs.

Limitations: There are a number of limitations of the current algorithm that we seek to improve in the future. First, while the algorithm achieves best results when there are large components in each line set that can be isometrically mapped, it performs less well for matching small isomet-

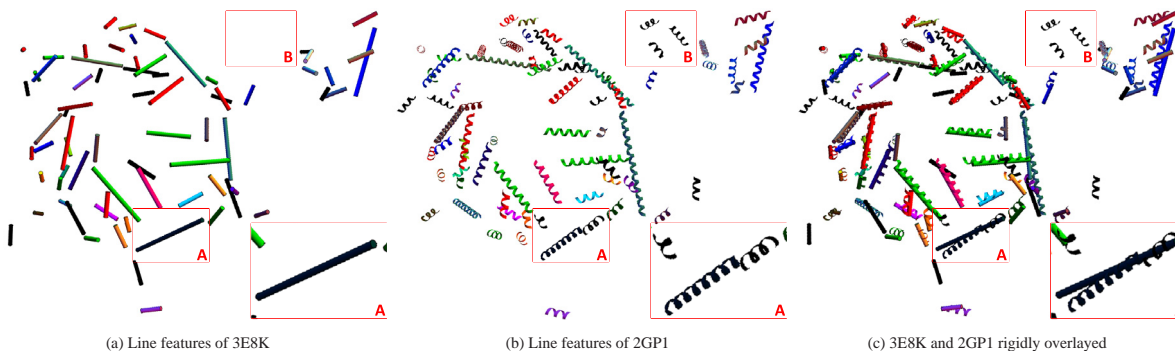


Figure 8: Line features detected in the 3E8K (a) and 2GP1 (b) proteins colored by the identified isometric clusters. The two shapes are overlaid (c) to show their differences. Observe that our method has successfully tolerated feature identification errors where one line feature has broken into multiple lines (region A), and where features in one shape are not present in the other (region B).

	Experiment	Ground truth corr. count	Correctly found		Not found		Incorrectly found corr.	Time (seconds)			
			corr.	%	corr.	%		Graph const.	Best-first	Cost matrix	Total
1	3E8K-A 2GP1-A	6	6	100.00%				0.01	0.00	0.00	0.01
2	3E8K-A 3DDX-A	7	7	100.00%				0.01	0.00	0.00	0.01
3	1OEL-A 1SS8-A	19	19	100.00%				0.04	0.00	0.00	0.04
4	1OEL-A 2C7C-A	20	18	90.00%	2	10.00%		0.09	0.01	0.01	0.11
5	3E8K-A 2GP1	46	45	97.83%	1	2.17%	2	0.93	0.15	0.01	1.09
6	3E8K-A 3DDX	49	49	100.00%				1.28	0.17	0.03	1.48
7	3E8K 2GP1	57	55	96.49%	2	3.51%	2	32.07	5.43	0.10	37.6
8	1OEL-A 1SS8	133	133	100.00%				9.90	1.48	0.06	11.44
9	1OEL-A 2C7C	331	242	73.11%	89	26.89%	7	151.04	51.34	0.71	203.09

Table 1: The results of our method compared against the ground truth, and the times taken at each stage.

ric components such as individual lines, which are resolved using a very greedy, voting-based approach. We note that small, individual helix deformations are not uncommon in proteins, in which case our method would perform poorly (e.g., this is the case for the 9th test in Table 1). However, these individual motions are still constrained by the protein structure (which is a wind-up string). To address this issue, it would be interesting to explore a more general, articulated registration formulation for line features.

Second, our algorithm may take a very long time for extremely large inputs (e.g., with over 400 lines), as the graph construction phase has a quadric complexity to the size of the input. We will explore ways to reduce the complexity, such as limiting edges in our product graph to be constructed only for line pairs in a given size of neighborhood.

Third, for the protein application, our method currently cannot match a long helix in one structure to its two broken-up pieces in a second structure. While the user can interactively specify this matching during the validation process, it would be interesting to consider algorithmic means to allow such one-to-many matching.

Acknowledgement This work is supported in part by NSF grants IIS-0705474, IIS-0705538 and the NCM grant through NIH RR002250.

References

- [AMCO08] AIGER D., MITRA N. J., COHEN-OR D.: 4-points congruent sets for robust pairwise surface registration. *ACM Transactions on Graphics (TOG)* 27, 3 (August 2008), 1–10.
- [ATCO*10] AU O. K.-C., TAI C.-L., COHEN-OR D., ZHENG Y., FU H.: Electors voting for fast automatic shape correspondence. In *Computer Graphics Forum (Proc. of Eurographics 2010)* (2010), IEEE.
- [BHN03] BERMAN H., HENRICK K., NAKAMURA H.: Announcing the worldwide protein data bank. *Nature Structural Biology* 10, 12 (2003), 980–980.
- [BJC07] BAKER M. L., JU T., CHIU W.: Identification of secondary structure elements in intermediate-resolution density maps. *Structure* 15, 1 (January 2007), 7–19.
- [BM92] BESL P. J., MCKAY N. D.: A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14, 2 (February 1992), 239–256.
- [CBJ*05] CHIU W., BAKER M. L., JIANG W., DOUGHERTY M., SCHMID M. F.: Electron cryomicroscopy of biological machines at subnanometer resolution. *Structure* 13, 3 (March 2005), 363–372.
- [FB81] FISCHLER M. A., BOLLES R. C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24, 6 (June 1981), 381–395.
- [FS06] FUNKHOUSER T., SHILANE P.: Partial matching of 3D shapes with priority-driven search. In *Symposium on Geometry Processing* (June 2006), IEEE, pp. 131–142.
- [GCO06] GAL R., COHEN-OR D.: Salient geometric features for partial shape matching and similarity. *ACM Trans. Graph.* 25, 1 (2006), 130–150.

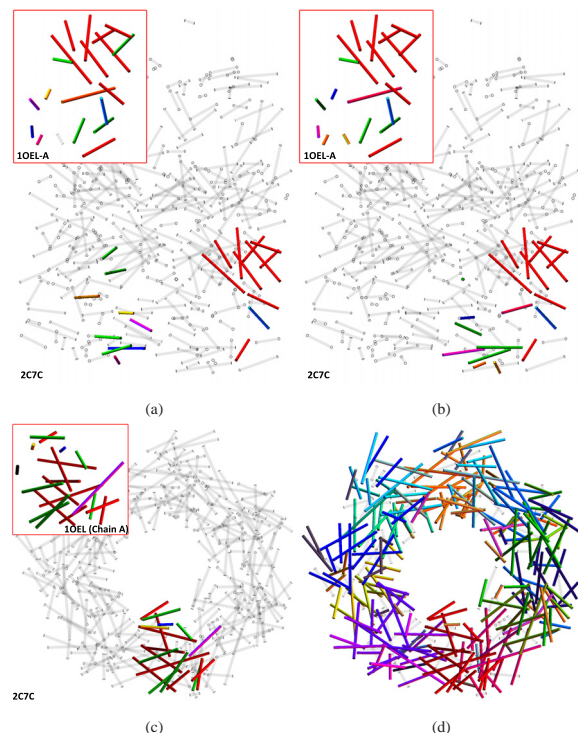


Figure 9: Registering a single chain (1OEL-A) to a large complex containing multiple similar chains (2C7C) without (a) and with (b,c) the spatial-coherence term, and the segmentation of the complex into individual chains (d, colored by chains) after repeated execution of the algorithm.

- [GZGG05] GATZKE T., ZELINKA S., GRIMM C., GARLAND M.: Curvature maps for local shape comparison. In *Shape Modeling International* (June 2005), IEEE, pp. 244–256. A local shape comparison technique for meshes.
- [HQ5*09] HOU F., QI Y., SHEN X., YANG S., ZHAO Q.: Automatic registration of multiple range images based on cycle space. *The Visual Computer: International Journal of Computer Graphics* 25 (2009), 657–665.
- [JH99] JOHNSON A. E., HEBERT M.: Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21, 5 (May 1999), 433–449.
- [JZ07] JAIN V., ZHANG H.: A spectral approach to shape-based retrieval of articulated 3d models. *Computer-Aided Design* 39, 5 (May 2007), 398–407.
- [LF09] LIPMAN Y., FUNKHOUSER T.: Möbius voting for surface correspondence. *ACM Transactions on Graphics* 28, 3 (August 2009), 1–12.
- [LLM08] LI M., LANGBEIN F. C., MARTIN R. R.: Detecting approximate symmetries of discrete point subsets. *Comput. Aided Des.* 40, 1 (2008), 76–93.
- [Low99] LOWE D. G.: Object recognition from local scale-invariant features. In *Proc. Seventh IEEE International Conference on Computer Vision* (1999), vol. 2, pp. 1150–1157.
- [PY03] PARK S.-J., YAMAMURA M.: Two-layer protein structure comparison. In *15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'03)* (2003).

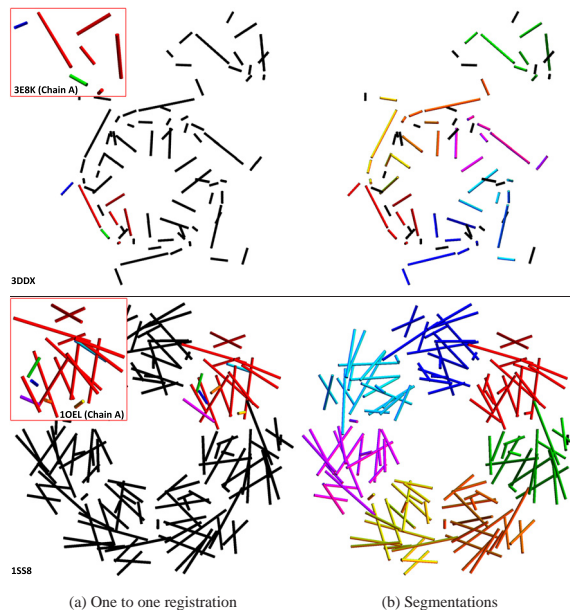


Figure 10: Applying our method repeatedly to segment individual chains, showing one chain (left, where colors indicate matching clusters) and the segmentation (right, where colors indicate chains).

- [RFP08] RAGURAM R., FRAHM J.-M., POLLEFEYS M.: A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. In *Computer Vision – ECCV 2008* (2008), ECCV, pp. 500–513.
- [Rob01] ROBSON J.: *Finding a maximum independent set in time $O(2^n/4)$* . Tech. rep., Laboratoire Bordelais de Recherche en Informatique, 2001.
- [TLW*08] TOPF M., LASKER K., WEBB B., WOLFSON H., CHIU W., SALI A.: Protein structure fitting and refinement guided by cryo-em density. *Structure* 16 (2008), 295–307.
- [TVM*08] TRABUCO L. G., VILLA E., MITRA K., FRANK J., SCHULTEN K.: Flexible fitting of atomic structures into electron microscopy maps using molecular dynamics. *Structure* 16, 5 (2008), 673–683.
- [WB01] WRIGGERS W., BIRMANNS S.: Using situs for flexible and rigid-body fitting of multiresolution single-molecule data. *Journal of Structural Biology* 133, 2-3 (2001), 193–202.
- [WG06] WANG Y., GUIBAS L. J.: Toward unsupervised segmentation of semi-rigid low-resolution molecular surfaces. *Geometric Modeling and Processing - GMP 2006 4077*, 2006 (July 2006), 129–142.
- [WR97] WOLFSON H., RIGOUTSOS I.: Geometric hashing: An overview. *IEEE Computational Science and Engineering* 4, 4 (1997), 10–21.
- [ZD06] ZHENG Y., DOERMANN D.: Robust point matching for nonrigid shapes by preserving local neighborhood structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28 (2006), 643–649.
- [ZSCO*08] ZHANG H., SHEFFER A., COHEN-OR D., ZHOU Q., VAN KAICK O., TAGLIASACCHI A.: Deformation-driven shape correspondence. *Computer Graphics Forum* 27, 5 (2008), 1431–1439.