January 2010

A Geometric Approach for Deciphering Protein Structure from Cryo-EM Volumes

Sasakthi Abeysinghe
Washington University in St. Louis

WASHINGTON UNIVERSITY IN ST. LOUIS

School of Engineering and Applied Science

Department of Computer Science and Engineering

Dissertation Examination Committee:
Tao Ju, Chair
Matthew L. Baker
Nathan A. Baker
Jeremy Buhler
Cindy M. Grimm
Robert Pless

A GEOMETRIC APPROACH FOR DECIPHERING PROTEIN STRUCTURE FROM CRYO-EM VOLUMES

by

Sasakthi Senanayaka Abeysinghe

A dissertation presented to the School of Engineering
of Washington University in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2010
Saint Louis, Missouri

ABSTRACT OF THE DISSERTATION

A geometric approach for deciphering protein structure from cryo-EM volumes

by

Sasakthi Senanayaka Abeysinghe

Doctor of Philosophy in Computer Science

Washington University in St. Louis, 2010

Research Advisor: Dr. Tao Ju

Electron Cryo-Microscopy or cryo-EM is an area that has received much attention in the recent past. Compared to the traditional methods of X-Ray Crystallography and NMR Spectroscopy, cryo-EM can be used to image much larger complexes, in many different conformations, and under a wide range of biochemical conditions. This is because it does not require the complex to be crystallisable [74, 89]. However, cryo-EM reconstructions are limited to intermediate resolutions, with the state-of-the-art being 3.6Å [116], where secondary structure elements can be visually identified but not individual amino acid residues. This lack of atomic level resolution creates new computational challenges for protein structure identification.

In this dissertation, we present a suite of geometric algorithms to address several aspects of protein modeling using cryo-EM density maps. Specifically, we develop novel methods to capture the shape of density volumes as geometric skeletons. We then use these skeletons to find secondary structure elements (SSEs) of a given protein, to identify the correspondence between these SSEs and those predicted from the primary sequence, and to register high-resolution protein structures onto the density volume. In addition, we designed and

developed *Gorgon*[1], an interactive molecular modeling system, that integrates the above methods with other interactive routines to generate reliable and accurate protein backbone models.

Acknowledgments

I would like to thank my advisor, Tao Ju who took me under his wing as his first PhD student. During the last five years he has constantly challenged me to reach new heights, think critically and seek ways to solve even the toughest of challenges. He has taught me much more than how to be a researcher, and I feel privileged to have worked with him.

I would also like to thank the members of my dissertation committee: Matthew Baker, Nathan Baker, Jeremy Buhler, Cindy Grimm and Robert Pless for their time, advise and constructive critique during the formulation of this dissertation and the many other times I have stopped by their offices unannounced.

I would like to thank my collaborators Matthew Baker, Wah Chiu, Ross Coleman and Mike Marsh at the Baylor College of Medicine for introducing me to the field of structural biology. Without their collaboration this dissertation would have looked very different.

I am truly grateful to Stephen Schuh for implementing the β-sheet correspondence, and his invaluable assistance when evaluating our method. I would like to thank my ever-optimistic office-mate Nathan Jacobs, and my lab mates Gazihan Alankus, Tom Erez, Manfred Georg, Robert Glaubius, Paul Gross, Ruosi Li, Lu Liu, Ly Phan, and Ross Sowell for their ideas, critique, and patiently enduring through my many practice talks. A special thanks goes out to Sajeeva Pallemulle for all his support in getting me started in the United States, and Prabath Gunawardane for being there whenever I needed a helping hand.

I would like to thank Jean Grothe, Kelli Eckman, Mryna Harbison, Madeline Hawkins and Sharon Matlock, their magical hand has made many a logistical nightmare disappear.

I would also express my heartfelt gratitude to my parents and sister. With their unconditional love, continuous support and sacrifices, they shaped me to be who I am today. Most of all I would like to thank my loving wife Shehani. Her love, support and understanding gave me the strength to endure the late nights, stressful days and moments of doubt and indecision.

Sasakthi Senanayaka Abeysinghe

Washington University in Saint Louis
May 2010

Contents

List of Tables

List of Figures

List of Abbreviations

- **3D:** Three dimensional
- **ARG:** Attributed relational graph
- **Cryo-EM:** Electron cryo-microscopy
- **CT:** X-Ray Computed tomography
- **CVS:** Concurrent Versions System
- **EMDB:** Electron microscopy data bank (http://emdatabank.org)
- **NMR:** Nuclear magnetic resonance
- **I/O:** Input/Output
- **MRI:** Magnetic resonance imaging
- **OLS:** Oriented line segment
- **PDB:** Protein data bank (http://pdb.org)
- **RMSD:** Root mean square deviation
- **SSE:** Secondary structure element
- **UML:** Unified modeling language

Chapter 1

Introduction

Proteins are the fundamental building blocks of all life forms, and are made up of a linear sequence of amino acid residues[1] also known as its primary structure (Figure 1.1a). In these proteins, neighboring amino acid residues can form groups of continuous segments called *secondary structure elements (SSEs)* most often seen as long helical tubes known as α-helices or large flat plates known as β-sheets[2] (Figure 1.1b). Based on the global interactions between these SSEs as well as the local interactions between the amino acids, each protein "folds" up in space into a specific 3D shape (Figure 1.1d), that determines how it interacts with other molecules. As a result, determining the 3D structure of proteins has critical importance in biomedical research [90].

Traditionally, protein structure prediction involves the use of high resolution imaging techniques, such as X-Ray crystallography and NMR spectroscopy. However, X-Ray crystallography can only be used with small, crystallisable molecules, while NMR spectroscopy is limited to relatively small molecules of an atomic mass less than 50 kDa. Therefore, neither of these techniques can be used to understand most of the larger macromolecular complexes seen in nature in their original state. In order to overcome these problems, computational techniques such as *Ab-initio* modeling and homology modeling have been extensively studied and used. However, these methods are inherently limited by factors such as availability of sequence homologues, variable accuracy and high computational cost due to the large search space as described in Levinthal's Paradox [61, 132].

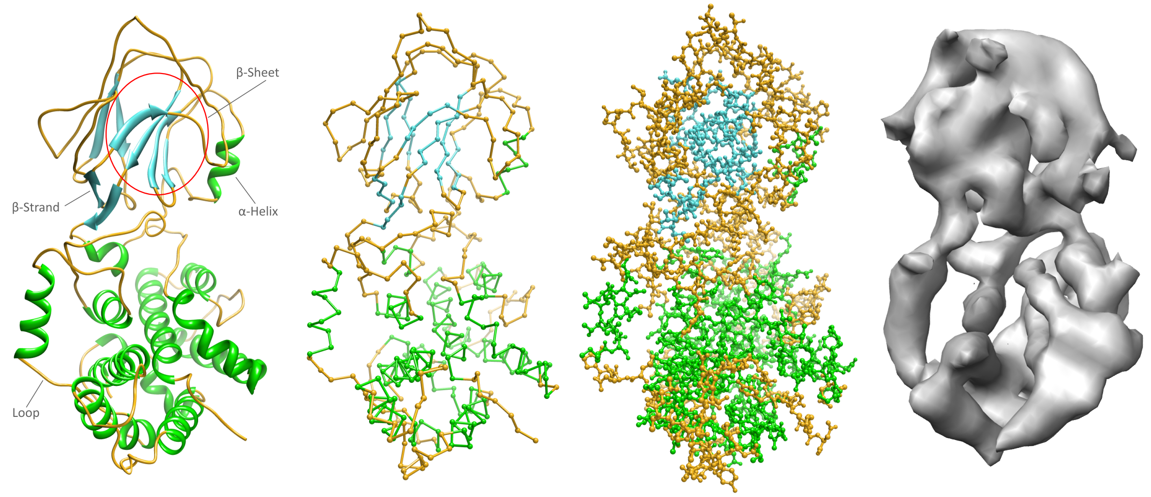
In an ongoing project between the Washington University in St. Louis and the Baylor College of Medicine, volumetric density maps of proteins obtained using an advanced imaging

[1] An amino acid is one of a class of organic compounds containing the amino (NH_2) and carboxyl (COOH) groups. An amino acid residue is such an amino acid that has lost a water molecule by joining with another amino acid.

[2] We will refer to α-helices, β-sheets and β-strands as helices, sheets and strands in this dissertation.

```
    1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
1   MSRQMWLDTSALLEAISEYVVRCNGDTFSGLTTGDFNALSNMFTQLSVSSAGYVSDPRVPLQTMSNMFVSFITSTDRCGYMLRKTWFNSDTKPTVSDDFI
101 TTYIRPRLQVPMSDTVRQLNNLSLQPSAKPKLYERQNAIMKGLDIPYSEPIEPCKLFRSVAGQTGNIPMMGILATPPAAQQQPFFVAERRRILFGIRSNA
201 AIPAGAYQFVVPAWASVLSVTGAYVYFTNSFFGTIIAGVTATATAADAATTFTVPTDANNLPVQTDSRLSFSLGGGNINLELGVAKTGFCVAIEGEFTIL
301 ANRSQAYYTLNSITQTPTSIDDFDVSDFLTTFLSQLRACGQYEIFSDAMDQLTNSLITNYMDPPAIPAGLAFTSPWFRFSERARTILALQNVDLNIRKLI
401 VRHLWVITSLIAVFGRYYRPN
```

(a) Sequence of amino acid residues



(b) Secondary structure elements (c) C_α backbone (d) Atomic model (e) Cryo-EM density volume

Figure 1.1: The sequence of amino acid residues of the Rice Dwarf Virus (RDV P8) protein (a), its secondary structure elements (α-helices in green, β-sheets and β-strands in blue) (b), backbone of amino-acid residues (c) and full atomic structure (d). Observe that the tubular and plate-like elements in the 6.8Å density volume obtained using Cryo-EM (e) roughly correspond to the α-helices and β-sheets seen in (b).

technique named electron cryo-microscopy (cryo-EM), are utilized to decipher the protein structure. Cryo-EM addresses most of the scalability concerns of the traditional techniques by being able to image large macromolecular complexes such as viruses at many different functional states [74, 89]. However, the resolutions of the volumes obtained most often range between 5Å and 10Å, with the state-of-the-art being 3.6Å [116]. Therefore, cryo-EM volumes cannot be used to directly determine the structure of proteins at atomic level resolution, or even its amino acid residue backbone. This limitation has motivated the development of many computational tools that use the intermediate-resolution cryo-EM volume to gather structural information about the protein [9, 59, 121, 106].

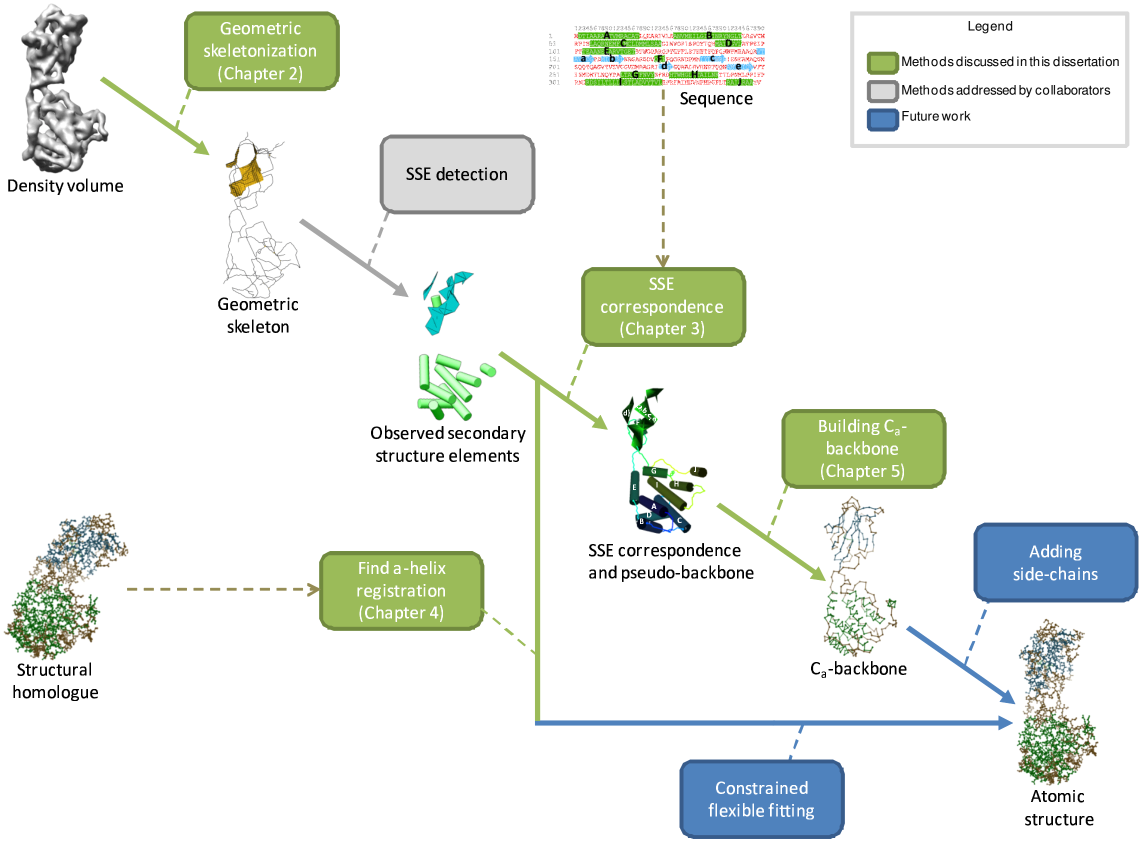

Geometric skeletonization (Chapter 2)
Density volume
Geometric skeleton
SSE detection
Sequence
Legend
🟩 Methods discussed in this dissertation
⬜ Methods addressed by collaborators
🟦 Future work
SSE correspondence (Chapter 3)
Observed secondary structure elements
SSE correspondence and pseudo-backbone
Building C$_a$-backbone (Chapter 5)
Adding side-chains
C$_a$-backbone
Structural homologue
Find a-helix registration (Chapter 4)
Constrained flexible fitting
Atomic structure

Figure 1.2: Different stages of the molecular modeling pipeline. The methods in green will be discussed in this dissertation, the method in gray has already been addressed by collaborators, and the methods in blue will be discussed as future work.

1.1 Method overview

As mentioned earlier, we are interested in the development of a computational approach that can be used to identify the atomic resolution structure of a molecule given its intermediate resolution cryo-EM volume. In other words, we are attempting to determine the 3D locations and orientations of every atom that makes up the protein in question. Apart from the density volume, we can also utilize the protein sequence information as well any known high resolution structures of the same (or similar) protein at different conformations.

Towards this goal, we have designed the molecular modeling pipeline described in Figure 1.2. Here we decompose the complex task of protein structure prediction into smaller, independent problems that we address using computational techniques that utilize the underlying geometric properties of the data.

A geometric approach: Our computational methods are grounded on the observation that the *geometry* of the density map at intermediate resolutions (Figure 1.1e) is closely related to the secondary structure elements and the shape of the protein backbone (Figure 1.1b). By understanding the geometry of the density maps, we are able to identify coarse-level protein components, and further utilize these elements to guide the modeling of fine-level structures.

Molecular modeling pipeline: Once again, our objective is to build a fine-level atomic resolution model given a coarse level cryo-EM density volume. For this purpose we base our molecular modeling pipeline on a coarse-to-fine paradigm, where we first identify coarse-level structural elements and then use that information to deduce finer and finer levels of detail. To realize this paradigm we must first identify geometric properties of the density volume. This can be achieved with geometric skeletonization techniques, where the shape and topology of the underlying density is captured in the form of a skeleton.

In the next stage of refinement, we use SSEHunter [9], a method proposed by our collaborators at the Baylor College of Medicine to identify the 3D positions, orientations and sizes of the secondary structure elements (SSEs). SSEHunter utilizes the geometric skeleton together with cross-correlation and geometric analysis to perform its function, and has been demonstrated with high accuracy for intermediate resolution cryo-EM density volumes.

At this stage of the pipeline we have two alternative paths to achieve finer levels of detail. The first path is based on the availability of a high-resolution structural homologue[3]. We can achieve the goal atomic level detail by fitting this structural homologue to the density in a flexible manner. For this purpose we first find the registration between the α-helices observed in the density, and those annotated in the structural homologue. This registration together with inter-atom bond properties can then form a set of constraints to an energy minimization routine to determine the best *flexible fit* of the homologue in the density.

In the absence of a structural homologue, we can make use of the sequence of amino acid residues to further refine our model. At this stage we find the correspondence between the SSEs observed in the density volume, and the SSEs predicted from the sequence. This

[3] A structural homologue is a protein which is similar (but not identical) in shape to another protein. Most often structural homologues have SSEs of the same shapes and sizes, but are scattered differently in 3D space due to many hinge-like motions on the loop segments.

correspondence together with the geometric skeleton allows us to find a mapping between the sequence and the density, leading to a pseudo-backbone trace of the protein.

The pseudo-backbone trace is based purely on the geometric properties of the density, and does not consider any inter-residue bond properties. Therefore, we can further refine this model by allowing a user to interactively build a C$_\alpha$ backbone trace that is guided by the pseudo-backbone, but is constrained by the inter-residue bond properties.

Finally, we envision refining this C_α backbone trace further by adding side-chains, and performing energy minimization at the scale of these side chains to build the final atomic structure.

1.2 Contributions

In this dissertation we address the activities highlighted in green in Figure 1.2. More specifically, we make the following novel contributions:

- **Segmentation-free geometric skeletonization:** As mentioned earlier, geometric skeletons give valuable insight towards understanding cryo-EM density volumes. However, obtaining a geometric skeleton that accurately captures the shape and topology of a density volume is a challenging task as there is no clear separation between the protein and background. The lack of resolution and the presence of noise in cryo-EM volumes further complicates this task. In Chapter 2, we discuss a segmentation-free geometric skeletonization technique that we developed [4] that can be used to quickly and accurately identify shape and topological features of complex and noisy biomedical images.

- **Correspondence between observed and predicted SSEs:** In the absence of a structural homologue, creating a *de novo* model of a molecule imaged using cryo-EM is a challenging and time consuming task. Knowing the correspondence between the secondary structure elements visible in the density volume and predicted from the sequence allows us to form an initial hypothesis on the structure of the protein backbone. In Chapter 3, we extend our previous work [3] and propose a novel and efficient method to identify a set of these most likely correspondences while being robust under SSE detection errors. This correspondence can be used to deduce a pseudo-backbone that can then be used to help structural biologists build *de novo* models.

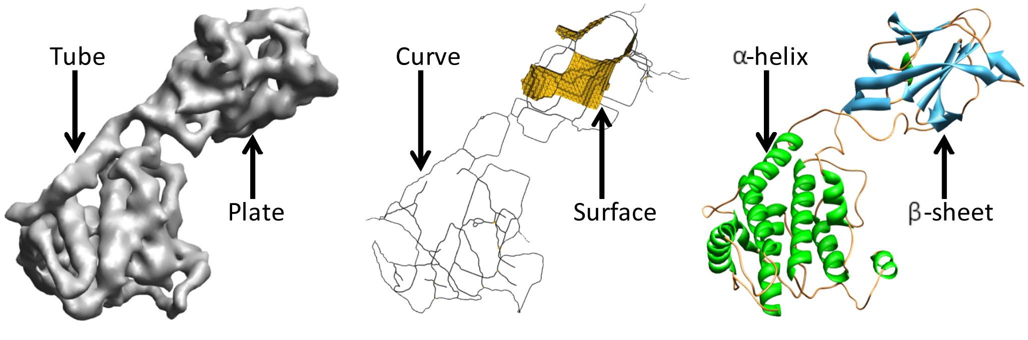
- **α-helix registration for flexible fitting:** Given a high-resolution structural homologue, one quick way of determining an approximate model is to fit the homologue into the density. However, proteins undergo many hinge-like deformations at different conformations making this fitting problem a challenging task. In Chapter 4, we propose a novel method that can be used to register the α-helices annotated in a high-resolution model with those observed from the volume. In the future, these registrations can be used as constraints for a better initialized flexible fitting routine. Our method is robust under α-helix detection errors, and is also capable of finding clusters of helices that remain isometric relative to each other.
- **Gorgon: a molecular modeling system:** Given a cryo-EM density volume, *de novo* modeling today is often performed using a wide variety of tools usable only as command line arguments or scripts. In Chapter 5, we describe Gorgon, a freely distributed molecular modeling system that we created to enable users to visually and interactively build molecular models. Gorgon provides a unique set of features geared towards *de novo* modeling from Cryo-EM density volumes, and can be easily extended to incorporate user-specific functionality.

Chapter 2

Segmentation-free geometric skeletonization

2.1 Introduction

As mentioned in the last chapter, we are faced with the task of understanding the structure of a protein given an intermediate resolution density volume obtained using electron cryo-microscopy. In these volumes, we lack the resolution to locate each individual atom or even amino acid residue. However, a visual inspection of the density volume can be used to roughly identify the locations of the secondary structure elements. For example, Figure 2.1a presents a simulated cryo-EM density volume of the 1BVP protein of the Bluetongue virus (BTV). Here, we can visually identify tubular and plate-like elements in the volume that correspond to the α-helices and β-sheets of the protein in 2.1c.

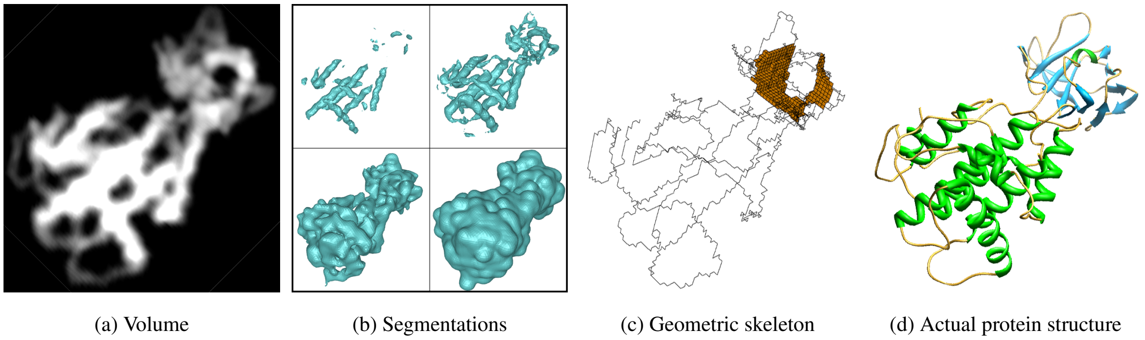


(a) Volume

(b) Geometric skeleton

(c) Actual protein structure

Figure 2.1: The geometric skeleton of a density volume gives important information about its secondary structure elements and connectivity. Observe that the plate-like and tubular parts of the density volume (a) captured as surfaces and curves in the skeleton (b) correspond well with the helices and sheets of the actual protein structure (c).


(a) Volume

(b) Segmentations

(c) Geometric skeleton

(d) Actual protein structure

Figure 2.2: A density volume of a protein molecule (a), the segmentation at various thresholds (b), the skeleton generated by our segmentation-free method (c), and the ground-truth structure of the protein (d).

For solid objects, a typical approach for identifying their rod-like and plate-like shape components is to consider the object's *skeleton* [14]. As seen in Figure 2.1b, rod-like and plate-like features are captured as curves and surfaces in the geometric skeleton, and correspond well to the secondary structure elements of the protein. Based on this observation, methods have been suggested [9, 3] that use geometric skeletons as an essential shape descriptor towards protein structure prediction from cryo-EM density volumes.

An ideal skeleton for this purpose would consist of medial curves and surfaces lying centered at the rod-like and plate-like parts of the object [87, 15, 54]. Unfortunately, the 3D data produced by cryo-EM (and most other medical imaging techniques) is usually in the form of a grayscale volume, which lacks a clear boundary between the object and the background. Although an object segmentation can be obtained by some particular threshold gray value, the segmented objects may have widely varying shapes depending on the choice of the threshold (as illustrated in Figure 2.2b). The skeletons of these segmentations would assume very different shapes, and the skeleton at a particular threshold may not reflect all shape components intrinsic to the grayscale volume.

In this chapter, we discuss a novel skeletonization algorithm we proposed in [4] that does not require an explicit segmentation of the volume into object and background, and is capable of producing skeletal curves and surfaces that lie centered at rod-shaped and plate-shaped sections of the volume. Our algorithm is not restricted to the protein domain, and can be generally applied to most biomedical images. We demonstrate this using a suite of data sets ranging from MRI scans of vascular structures to CT scans of bones.

2.1.1 Problem statement

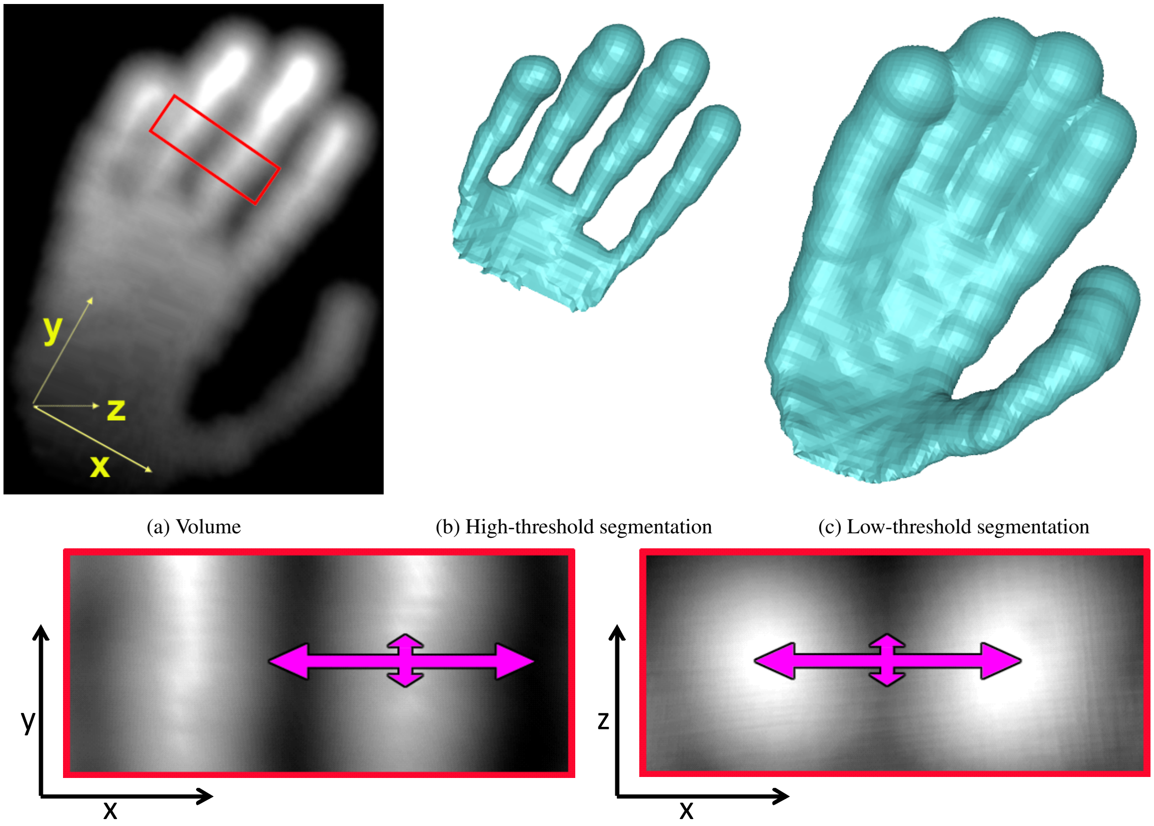
We are interested in computing the skeleton of a grayscale volume for identifying its intrinsic shape components. Instead of depending on the segmentation at some threshold, the skeleton should consist of curves and surfaces that are centered at the rod-like and plate-like parts of the full, un-segmented grayscale volume.

In contrast to the vast literature on skeletonization of solid models, computing skeletons on un-segmented 3D volumes has received much less attention (see review in Section 2.2). In particular, we know of no existing method capable of extracting both skeletal curves and surfaces from a grayscale volume for the purpose of shape understanding without specific domain knowledge or an object segmentation.

One aspect of this skeletonization problem that requires further clarification is what constitutes a rod-like part or a plate-like part in a grayscale volume. In these volumes, the gray values behave like a density distribution, where voxels with higher values are likely to be located closer to the center of the imaged subject. In addition, different parts of the subject may exhibit different brightness levels. Many bio-medical imaging techniques (i.e. MRI, CT, EM) produce volumes that have features that satisfy these observations. However, different imaging modalities (ex: T1-weighted and T2-weighted MRI scans) capture different features of the subject being imaged, and thus what is characterized as a rod-like part or a plate-like part will vary based on the imaging modality. We explain our observations using a synthetic Hand volume in Figure 2.3a, that contains 5 rod-like parts (the fingers) and 1 plate-like part (the palm):

Observation 1: A shape component, such as a rod or a plate, in a density-like volume is usually captured by the segmented object at *some* threshold values. For example, the top four fingers in the Hand volume appear as rods in the segmentation at one threshold (b), while the palm forms a plate at a different threshold (c). However, this observation alone is not sufficient to disambiguate the different types of shape components located in the same part of the volume. For example, the top four fingers also form a plate in the segmentation (c).

Observation 2: The center of a shape component usually lies at the "ridge" of the underlying density function. In particular, the *variation* of gray values at a ridge point is usually



(a) Volume (b) High-threshold segmentation (c) Low-threshold segmentation

(d) Volume X-Y cross-section (e) Volume X-Z cross-section

Figure 2.3: A grayscale volume (a), segmented surfaces at two thresholds (b,c), and close-up view of two fingers at different angles (d,e) where the pink arrows illustrate the magnitude of grayscale variation in different directions (see Section 2.1.1).

smaller along the ridge than in other directions. For example, the grayscale variation along the center line of each finger is much smaller than along other directions (as illustrated in the close-up view in (d)). In contrast, a plate-like shape formed by the top four fingers would not have this property, as the grayscale variation along the center surface of this plate can be much greater than in some other direction, especially between two fingers (as illustrated in the cross-section view in (e)).

2.1.2 Method

Our algorithm extracts the skeleton of a density-like grayscale volume guided by the above two principle observations. There are two main stages in this algorithm. In the first stage, the algorithm visits all possible threshold values and identifies the shape components on the segmented object at each threshold by their center curves and surfaces. According to observation 1, the collection of these shape components on various segmentations is a super-set of those intrinsic to the grayscale volume, the latter of which, by observation 2, are characterized by the ridge-like centers in the density distribution. Hence, the second stage extracts the final skeleton as the sub-set of curves and surfaces generated in the first stage that exhibit small grayscale variation.

An example result of our method is shown in Figure 2.2c. Note that the curves (black lines) and surfaces (orange faces) in the skeleton correspond well to α-helices and loops (which appear as rods in the volume) as well as β-sheets (which appear as plates in the volume) in the actual protein structure in (d).

2.1.3 Contribution

The primary contribution of this chapter is a novel algorithm for computing the skeleton of a density-like grayscale volume. The algorithm does not require segmentation at any particular threshold, and the resulting skeleton consists of curves and surfaces centered at rod-like and plate-like parts of the grayscale volume. The independence from a threshold makes the skeletonization process less sensitive to human bias and allows for the understanding of the intrinsic shapes in such a volume. The method is demonstrated on both synthetic and medical data.

2.2 Previous work

Here we briefly review skeletonization methods for 3D solid and grayscale models, with an eye towards identifying shape components.

Solid models: Computing skeletons of 3D solid models has been extensively researched in the past. A number of representative methods include morphological thinning [11, 79, 100], distance transforms [17], potential field methods [6], and Voronoi diagrams [95, 32]. For the purpose of identifying shape components, Saha *et al.* [87] and Bonnassie *et al.*[15] differentiates curves and surfaces in the skeletons generated by morphological thinning by classifying skeleton voxels based on their local neighborhood. While the result of such classification can be highly sensitive to the quality of the skeleton, the method of Ju *et al.* [54] directly extracts skeletal curves and surfaces during the thinning process without need for post-classification. These methods have been used to classify rod-like and plate-like structures in bone matrices [87, 15] and proteins [54].

Grayscale volumes: In contrast, few works have addressed skeletonization of unsegmented grayscale volumes. Although the use of morphological thinning has been well-studied in the vision community for skeletonizing 2D grayscale images (see the excellent survey by Mersal and Darwish [72]), extensions to 3D volumes have been rare. Segmentation techniques (see surveys [88, 40, 115]) have been used to build solid models of grayscale volumes; however, skeletons computed from these models are medial to the segmentation and do not align well with the high density regions of the grayscale volume. In contrast, the method of Svensson *et al.* [101] generates skeletal surfaces starting from a known object segmentation, but utilizes the interior grayscale information. Similar to ours, the method of Doklada *et al.* [33] computes an initial skeleton by thinning on the full grayscale volume, but it then requires a grayscale threshold to remove insignificant skeleton parts and is designed only for skeletal curves.

In a different approach, Lopez *et al.* [64] identifies centers of a grayscale distribution using a multi-local crease-ness measure, continuing a body of research on ridge and valley detection in 2D images (see the survey and evaluation in [63]). However, Lopez's method results in a collection of center points that lack any curve or surface structure necessary for identifying shape components.

Several researchers have proposed to explicitly extract both curves and surfaces in a grayscale volume based on the second-order tensor field of the volume [128, 55, 124]. However, these methods are either designed for visualizing flow anisotropy [128, 55] rather than locating shape components, or require domain-specific knowledge to find those curves or surfaces at the center of the shape components [124]. In contrast, our method relies on the same tensor field for extracting curve and surface geometry but is capable of placing such geometry at the center of grayscale shape components without the use of application-domain specific knowledge.

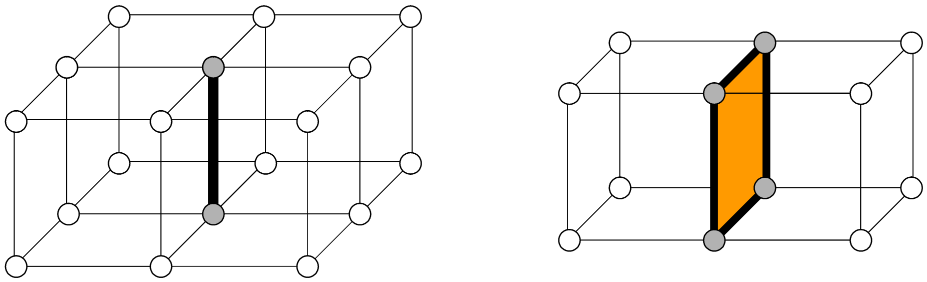


Figure 2.4: Examples of grids with voxels (circled dots) where skeleton voxels (gray) form edges (black lines) and a face (orange quad).

2.3 Overview

2.3.1 Data representation

Our algorithm takes in a volume represented as a 3D rectilinear grid, where each grid point, called a *voxel*, is associated with a grayscale value. The output skeleton of our algorithm consists of a subset of voxels on this grid. Figure 2.4 shows examples of grids and voxels where skeleton voxels are colored gray.

For the purpose of shape understanding, we define two types of geometry, curves and surfaces, on a set of voxels. A curve is a collection of *voxel edges*, each consisting of two voxels lying on the ends of a grid edge. A surface is a collection of *voxel faces*, each consisting of four voxels lying on the corners of a grid face (i.e. a grid face surrounded by four voxel edges). For example, the skeleton voxels on the left of Figure 2.4 form a voxel edge, and those on the right form four voxel edges and a voxel face. In this figure (and other figures in the chapter), voxel edges are drawn as black lines and voxel faces as orange quads.

2.3.2 The algorithm

Given a density-like grayscale volume, our algorithm, guided by the observations in Section 2.1.1, extracts skeletons consisting of curves and surfaces corresponding to the rod-like and plate-like parts of the volume. Since each shape component in the volume is captured by the segmented object at some threshold value, we first identify the set of all shape components at a range of threshold values. This is done by accumulating the skeletal curves and surfaces

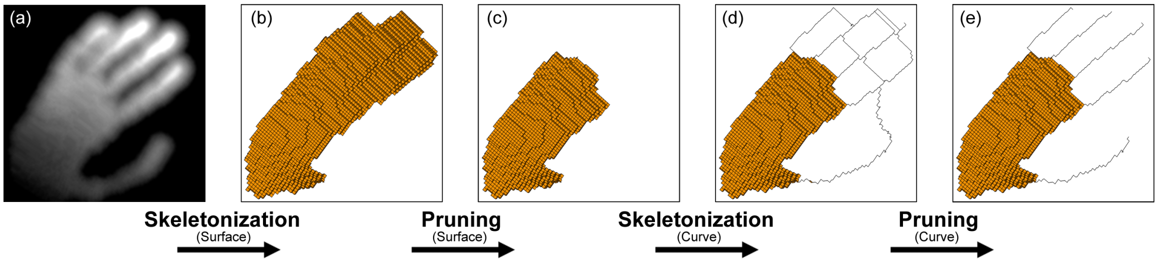


Figure 2.5: The four steps in our algorithm.

computed from the segmented objects at each threshold value. The sub-set of this initial skeleton that represents the grayscale shape components are then identified as those curves and surfaces exhibiting small grayscale variations. This is done by comparing the directions of these geometric elements with the shape of grayscale variation in a local neighborhood of each voxel.

In order to extract two types of skeletal geometry, namely curves and surfaces, we devise a four-step flow that first extracts the skeletal surfaces followed by the skeletal curves. The generation of each type of skeletal geometry follows the same stages of initial skeleton generation and skeleton pruning, as illustrated in Figure 2.5:

- **Step 1: Initial skeletonization.** Accumulate the skeletal *surfaces* of each segmentation of the grayscale volume at a range of thresholds.
- **Step 2: Pruning.** Identifies those *surfaces* in the result of step 1 that exhibit small grayscale variations.
- **Step 3: Initial skeletonization.** Accumulate the skeletal *curves* of each segmentation of the *original* grayscale volume at the same range of thresholds.
- **Step 4: Pruning.** Identifies those *curves* in the result of step 3 that exhibit small grayscale variations.

To ensure accurate identification of surface and curve features, surface skeletonization and pruning has to be done *before* curve skeletonization and pruning. This restriction arises from the fact that a curve can be geometrically defined as a subset of a surface, and performing curve skeletonization (and pruning) on a feature which is actually a surface (but not yet classified as a surface) will result in that feature being incorrectly classified as a curve.

In the following section, we will describe the skeletonization and pruning steps in detail.

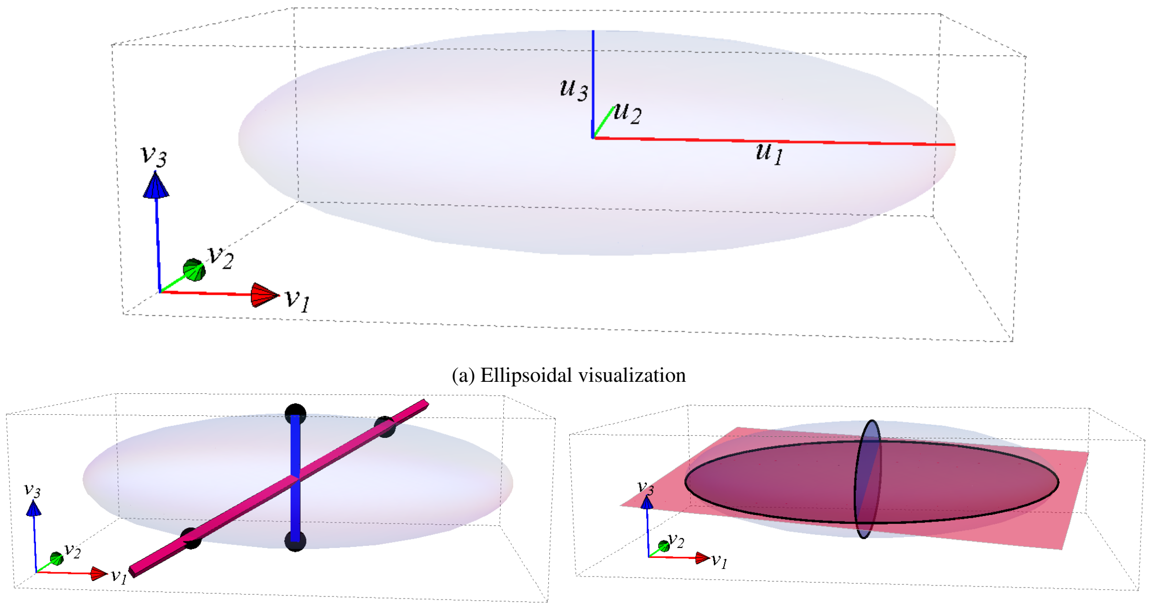
2.4 Method details

2.4.1 Initial skeletonization

Given a segmented object at a particular threshold, that is made up of voxels on a grid, a classical approach of obtaining its skeleton is morphological thinning [11]. To be able to identify shape components such as rods and plates, we consider the iterative thinning approach of Ju *et al.* [54], which selectively generates curves or surfaces (as defined in Section 2.3.1) centered at these parts. Briefly, this method shrinks the object to its medial structure by iteratively removing its border voxels. Skeletal curves or surfaces are identified by preserving voxels at the ends of either curves or surfaces during thinning.

To accumulate the skeletal surfaces or curves computed at multiple thresholds, we modify the method of Ju *et al.* [54] to utilize skeletons generated at different thresholds. Specifically, we segment the volume with decreasing threshold values. At each threshold, we compute the skeletal surfaces (in step 1) or curves (in step 3) by thinning the segmented object while additionally preserving, at each thinning iteration, the voxels belonging to the skeletons generated at previous thresholds. This incremental approach, combined with iterative thinning, ensures that skeletons computed at lower thresholds are aligned with skeletons at higher thresholds, and hence to regions with high gray values, which are likely to be centers of the shape components in the grayscale volume. The results of this incremental thinning for the hand example in Figure 2.5a are shown in Figures 2.5b and 2.5d.

In our implementation, the range of threshold values is taken as the full range of gray values in the volume, unless the user specifies a maximum and/or minimum gray value of interest. As enumerating each gray value present in the volume within the range can be time-consuming, we may also use values at discrete intervals. In all our examples, we discretize an input threshold range into 256 levels and visit each level in descending order.



(a) Ellipsoidal visualization

(b) Line scoring

(c) Surface scoring

Figure 2.6: Grayscale variation visualized as an ellipsoid (a), scoring of a line (red) as ratio between length of the shortest axis (blue) over that of the line segment in the ellipsoid (b), and scoring of a plane (red) as ratio between area of the smallest axial ellipse (blue) over that of the cross-sectional ellipse (c).

2.4.2 Pruning

The initial skeleton generated by the previous step (e.g. Figures 2.5b, 2.5d)) contains a super-set of skeletal surfaces or curves that represent the actual shape components of the grayscale volume. Based on our earlier observation, the desired sub-set of surfaces or curves are those along which the grayscale variation is smaller than along other directions. We will identify this sub-set in two phases. First, we will compute a score at each skeleton voxel based on the direction of the skeletal surface or curve at that voxel with respect to the shape of the local grayscale variation. Next, we will extract well-formed surfaces or curves consisting of high-score voxels.

Scoring

Structure tensor: While it is possible to measure the grayscale variation at each voxel by its local gradient, such measurement easily becomes unreliable in the presence of noise, which is typical in medical volumes. Instead, the structure tensor offers an average measurement of such variation within a neighborhood of each voxel, which is much more robust under noisy conditions. Specifically, we first compose a tensor T' at a voxel p as a 3×3 matrix:

$$T'_p = \begin{bmatrix} I_x \\ I_y \\ I_z \end{bmatrix} \times \begin{bmatrix} I_x \\ I_y \\ I_z \end{bmatrix}^T = \begin{bmatrix} I_x^2 & I_xI_y & I_xI_z \\ I_xI_y & I_y^2 & I_yI_z \\ I_xI_z & I_yI_z & I_z^2 \end{bmatrix} \tag{2.1}$$

where $I_x = \frac{\partial I}{\partial x}$, $I_y = \frac{\partial I}{\partial y}$ and $I_z = \frac{\partial I}{\partial z}$ are the partial derivatives in the x, y and z directions of the grayscale volume I at p. Performing spatial averaging of these tensors in the neighborhood of p using a Gaussian convolution mask g_σ with standard deviation σ gives the *structure tensor* T_p:

$$T_p = g_\sigma * T'_p. \tag{2.2}$$

The key piece of information offered by the structure tensor T_p is in its Eigen-structure, which reveals the principal directions and magnitudes of grayscale variation in the neighborhood of p. As shown in Figure 2.6a, the *shape* of this variation can be visualized as an ellipsoid whose axes are along the eigenvectors of T_p with the magnitude of the corresponding eigenvalues. Intuitively, the gray values around the voxel p vary more dramatically along directions closer to the major axis of the ellipsoid (i.e. the eigenvector with the largest eigenvalue), and less along directions closer to the minor axis (i.e. the eigenvector with the smallest eigenvalue).

Scoring surfaces and curves: The ellipsoidal representation of the grayscale variation offers an intuitive way of measuring the variation in any given direction. For example, the grayscale variation along a given line can be measured as the length of the line segment within the ellipsoid (Figure 2.6b). Since we are more interested in whether such variation is *smaller* than variations in other directions, we can score a voxel on a skeletal curve by the *ratio* of the minimum length of such line segments (i.e., the shortest axis of the ellipsoid) over the actual length along the tangent line of the curve. Likewise, we can score a voxel on a skeletal surface by the ratio of the minimum area of a cross-section in the ellipsoid (i.e. formed by the two shortest axes) over the actual area of the cross-section along the tangent plane of the surface (Figure 2.6c).

Specifically, denote the eigenvectors and eigenvalues of the structure tensor T_p by $\{v_1, v_2, v_3\}$ and $\{u_1, u_2, u_3\}$. As T_p is a positive semi-definite matrix $u_1 \geq u_2 \geq u_3 \geq 0$. Given a line passing through the origin in the unit direction of $c = \{c_x, c_y, c_z\}$, the length of the line segment within the ellipsoid is:

$$L(c) = \frac{u_1 u_2 u_3}{\sqrt{u_2^2 u_3^2 c_x^2 + u_1^2 u_3^2 c_y^2 + u_1^2 u_2^2 c_z^2}}. \tag{2.3}$$

Given a plane passing through the origin defined by two orthogonal unit vectors $n_1 = \{n_{1x}, n_{1y}, n_{1z}\}$ and $n_2 = \{n_{2x}, n_{2y}, n_{2z}\}$ on the plane, the area of the cross-section of the ellipsoid (which is an ellipse) is computed as:

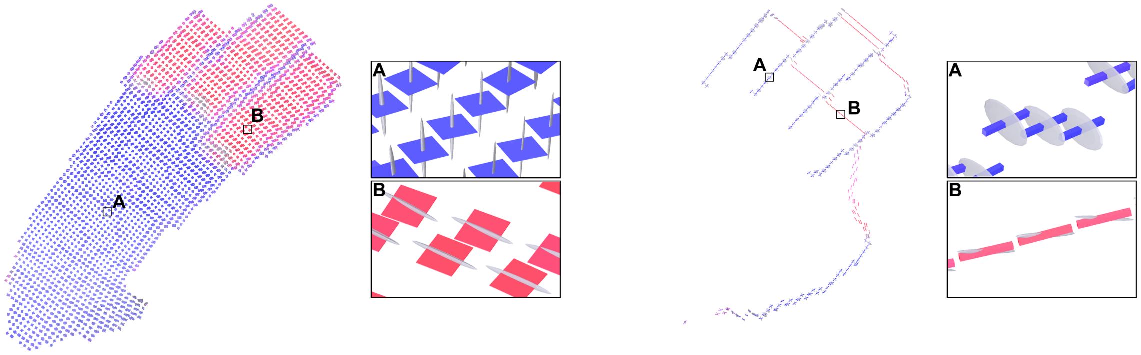
$$A(n_1, n_2) = \frac{\pi}{|m_1 \cos\theta + m_2 \sin\theta| \times |m_1 \sin\theta - m_2 \cos\theta|} \tag{2.4}$$

where,

$$\begin{aligned} m_1 &= \left\{ \frac{n_{1x}}{u_1}, \frac{n_{1y}}{u_2}, \frac{n_{1z}}{u_3} \right\}, \\ m_2 &= \left\{ \frac{n_{2x}}{u_1}, \frac{n_{2y}}{u_2}, \frac{n_{2z}}{u_3} \right\}, \\ \theta &= \frac{1}{2} \arctan\left(\frac{2 m_1 \cdot m_2}{m_1 \cdot m_1 - m_2 \cdot m_2} \right). \end{aligned}$$

The score of a skeletal curve with tangent vector c is therefore the ratio $\frac{L(v_3)}{L(c)}$, and the score of a skeletal surface with tangent vectors n_1, n_2 is the ratio $\frac{A(v_2, v_3)}{A(n_1, n_2)}$. Note that the score is bounded between $[0, 1]$, where 1 corresponds to the direction of minimum grayscale variation locally at p. To avoid possible numerical instability when evaluating scoring functions for very small values of u_1, u_2 and u_3, we note the limit of these functions are well defined when one or more of the eigenvalues approach zero. In practice, we treat any eigenvalue smaller than a threshold (such as 0.00001) as zero and directly apply the limit formula.

Due to the use of a rectilinear grid, the tangent orientation of the skeletal surface or curve at a voxel, if computed locally, will assume a limited number of directions restricted by the axis-aligned voxel faces and edges (i.e. six if using 6-connectivity). To overcome this limitation, we obtain these orientations by computing a best-fitting line or plane to all voxel faces or edges in a neighborhood of the voxel p.



(a) Scored surfaces

(b) Scored curves

Figure 2.7: Voxel scores on the skeletal surfaces (a) and curves (b) in Figure 2.5 (b,d), showing the ellipsoidal representation of grayscale variations and the tangent orientation of surfaces and curves. Blue and red indicate high and low scores.

In Figure 2.7, we show the scores computed for the surfaces and curves resulting from the initial skeletonization steps in Figures 2.5b and 2.5d. Note that lower scores (colored red) effectively indicate skeletal geometry (e.g. surfaces and curves between the fingers) that do not correspond well to actual shape components in the grayscale volume.

Numerical stability of the scoring function: The ellipsoidal representation of the structure tensor has a clear geometric shape in the limit cases when the eigenvalues approach zero, and can be used to explicitly derive the limit of scoring functions in these conditions.

When u_1 approaches zero so does u_2 and u_3 (as $u_1 \geq u_2 \geq u_3$). In this case we treat the ellipsoid as a sphere, where all embedded line segments have the same length, and all embedded cross sections have the same area. Therefore, the value of the scoring functions (the ratio between the minimal and actual length / area) will always evaluate to one.

$$\lim_{u_1 \to 0} \frac{L(v_3)}{L(c)} = 1, \tag{2.5}$$

$$\lim_{u_1 \to 0} \frac{A(v_2, v_3)}{A(n_1, n_2)} = 1. \tag{2.6}$$

In the case where u_2 approaches zero, so does u_3, reducing the ellipsoid to a needle with an infinitesimally small circular cross section (also can be interpreted as an infinitely long cylinder). Here, the minimal line (or surface) is the projection of the actual line (or surface) onto the plane defined by the surface normal v_1, reducing the scoring functions to the following vector dot products:

$$\lim_{u_2 \to 0} \frac{L(v_3)}{L(c)} = c \cdot \frac{(v_1 \times (c \times v_1))}{\|v_1 \times (c \times v_1)\|}, \tag{2.7}$$

$$\lim_{u_2 \to 0} \frac{A(v_2, v_3)}{A(n_1, n_2)} = (n_1 \times n_2) \cdot v_1. \tag{2.8}$$

When u_3 approaches zero, the ellipsoid reduces to a cylinder with an infinitesimally small height and an ellipse shaped cross section where u_1 and u_2 are the length of the axes. The minimal line is the projection of the actual line onto the v_3 vector, reducing the curve scoring function to the following vector dot product:

$$\lim_{u_3 \to 0} \frac{L(v_3)}{L(c)} = c \cdot v_3. \tag{2.9}$$

The surface score on the other hand, reduces to the product of two ratios; the first being the ratio between u_2 and the length of the curve within the ellipsoid in the l_1 direction, and the second being the ratio between l_2 and its projection onto the v_3 vector. l_1 and l_2 are orthogonal unit vectors which both lie on the actual surface. Additionally, l_1 also lies in the plane defined by the normal v_3.

$$\lim_{u_3 \to 0} \frac{A(v_2, v_3)}{A(n_1, n_2)} = \frac{\sqrt{u_1^2 \cos^2(\theta) + u_2^2 \sin^2(\theta)}}{u_1} \times (l_2 \cdot v_3), \tag{2.10}$$

where,

$$\begin{aligned} l_1 &= \frac{v_3 \times n}{\|v_3 \times n\|}, \\ l_2 &= n \times l_1, \\ \theta &= \arccos(v_2 \cdot l_1), \\ n &= (n_1 \times n_2). \end{aligned}$$

Therefore, we can assume stable behavior for our scoring functions, and use the corresponding limit values when u_1, u_2 or u_3 approach near-zero values.

Feature extraction

Given scored skeleton voxels (e.g. Figure 2.7), we next need to identify pieces of surfaces (in step 2) or curves (in step 4) consisting of high-scored voxels. Ideally, the final skeleton should consist of clean, recognizable surfaces and curves that are free of extraneous features such as small branches and islands. To this end, we first remove all voxels that score below a threshold. We find that the threshold of $\frac{1}{\sqrt{3}}$ (the ratio of the edge length over the diagonal length of a unit cube) works well for both surfaces and curves. Next, we utilize the morphological opening operator in [54] designed for skeletal curves and surfaces to remove extraneous skeleton features. Given user-specified size parameters $\varepsilon_s, \varepsilon_c$, this operator removes surface branches with radius smaller than ε_s and curve branches shorter than ε_c. The final results of skeleton pruning for the hand example are shown in Figures 2.5c and 2.5e.

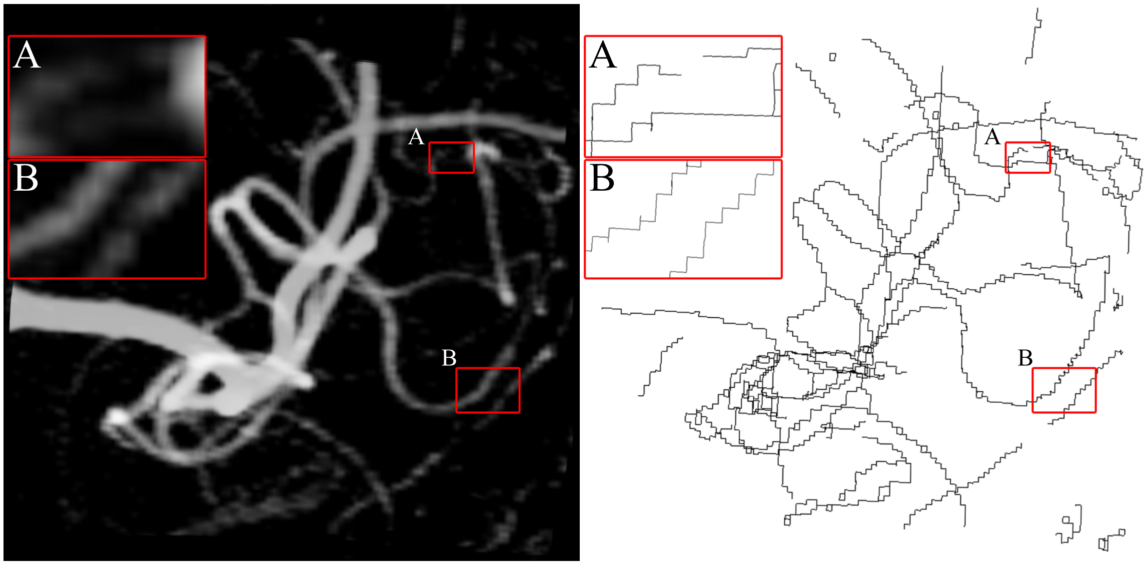
As shown in Ju *et al.* [54], the choice of $\varepsilon_s, \varepsilon_c$ controls the minimum size of the surface or curve feature in the final skeleton. This number typically only depends on the grid resolution and the type of subject being imaged. In our experiments, we use $\varepsilon_s = \varepsilon_c = 5$ except for imaged subjects made up of only rod-like parts (e.g., blood vessels), where we set $\varepsilon_s = \infty$, and subjects made up of only plate-like parts (e.g., cortical bones), where we set $\varepsilon_c = \infty$.

2.5 Results

We demonstrate our algorithm on a set of medical data produced by MRI, CT and cryo-EM imaging, where the biological structure of interest consists of rod-like and/or plate-like components.

Figure 2.8 shows the results of our method on an MRI scan of blood vessels in the human head. Observe from the close-up views, that, without relying on a particular threshold value, our technique was able to capture vessels at a wide range of gray levels and thicknesses, some of which are not even visible to the naked eye.

Figure 2.9 shows both the intermediate and final results of our method on a CT scan of blood vessels. The usefulness of pruning based on grayscale variations is illustrated in the close-up view between two vessels, where a skeletal curve is generated during initial



(a) Volume

(b) Geometric skeleton

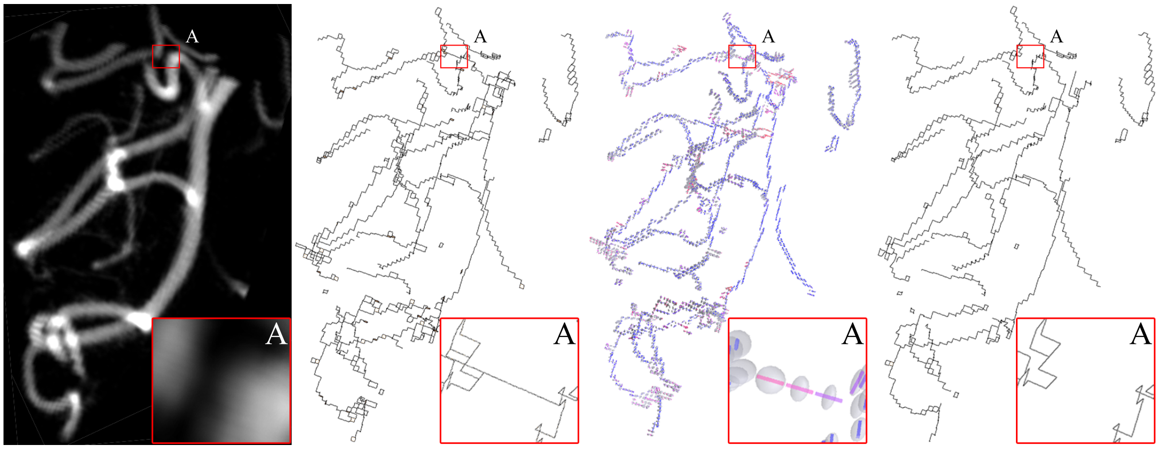
Figure 2.8: An MRI scan of blood vessels in the human head (a), the skeleton generated by our algorithm (b) that captures thin blood vessels that are barely visible to the naked eye (A, B).

skeletonization (as the segmented surface is connected at a low gray level), but receives a low score 2.9b as the curve exhibits a large grayscale variation, and finally gets removed 2.9d.

Figure 2.10 shows another example where our method computes a surface skeleton of cortical bones in a CT scan of the human foot. As seen in the cross-sections (c,d), our technique accurately captures the shell-shape of the cortical bones and preserves their hollow nature. Note that the skeleton is computed independent of any thresholds, and hence is capable of capturing both bright and dark potions of the cortical shell well.

As our overall goal is to understand molecular structures, we present two examples of skeletonization of protein volumes imaged using cryo-EM in Figure 2.11 (simulated). As mentioned earlier, the rod-like and plate-like parts of these volumes correspond well to the secondary structure elements of the protein, including α-helices (rod-like), β-sheets (plate-like) and loops (rod-like). Observe from Figure 2.11c that our method is capable of capturing shape components that correlate well with the actual protein structures shown in 2.11d.

We additionally compare our method with a previous method by Yu *et al.* [124] for computing skeletal curves and surfaces specifically in cryo-EM data. Yu's method also relies



(a) Volume (b) Initial skeleton (c) Scored ellipsoids (d) Pruned skeleton

Figure 2.9: A CT scan of blood vessels in the human head (a), the skeletal curves obtained after initial skeletonization (Step 3) (b) and pruning (Step 4) (d), and the voxel scoring during pruning (c).

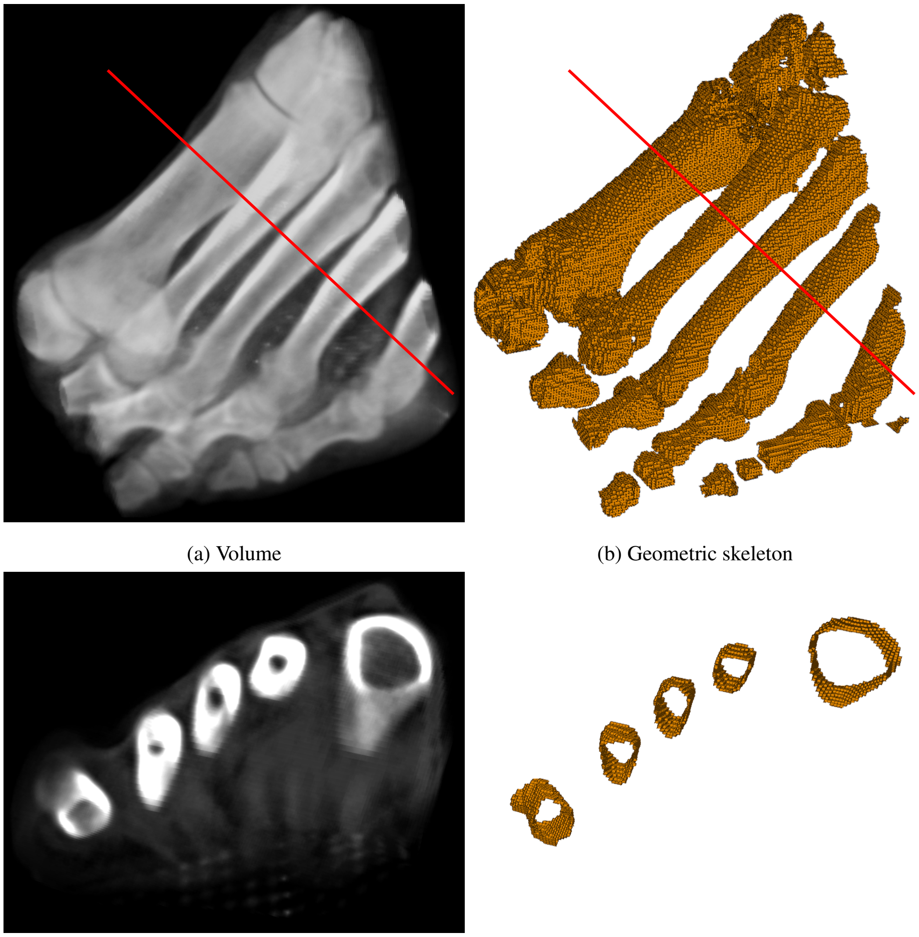
on the structure tensor for extracting the skeletal geometry, but requires explicit knowledge about the typical thickness of α-helices and β-sheets as well as their brightness level in order to locate the corresponding rod-like and plate-like parts in the volume. In comparison with the result of Yu's method in Figure 2.11b, our method, without any domain-specific knowledge (i.e. the typical thickness of α-helices and β-sheets, or their brightness levels), additionally extracts skeletal curves that correspond to loop structures in the protein. The difficulty Yu's method faces when identifying loops arises from the fact that loops lack a uniform thickness and are often at low gray levels in a cryo-EM volume.

Data Set	Dimensions	Step 1		Step 2		Step 3		Step 4		Total
	(voxels)	Time	Voxels	Time	Voxels	Time	Voxels	Time	Voxels	Time
Hand	$129 \times 129 \times 129$	4.12	3640	4.40	1981	3.85	2331	1.65	2202	14.04
Protein 2ITG	$64 \times 64 \times 64$	6.76	4736	3.18	142	6.79	1198	0.73	659	17.48
Protein 1TIM	$96 \times 96 \times 96$	16.87	8954	5.93	978	17.04	3067	1.75	1735	41.61
Protein 1BTV	$128 \times 128 \times 128$	34.29	12232	8.48	747	34.76	3777	2.73	1910	80.27
Blood Vessels (CT)	$121 \times 71 \times 66$	11.67	6608	0.59	0	11.78	4737	1.85	1757	25.90
Blood Vessels (MRI)	$101 \times 82 \times 111$	11.34	10313	0.71	0	11.41	7753	2.75	2662	26.23
Bones	$150 \times 128 \times 128$	33.50	143617	76.29	78178	29.67	105708	9.98	78178	149.44

Table 2.1: Time taken (in seconds) for each step of the algorithm (see Section 3.2) and the number of skeleton voxels after each step.

Table 2.1 shows the breakdown of the time for each step in our algorithm[4]. The time complexity of the initial skeletonization process is $O(ng)$, where n is the number of voxels, and g is the number of distinct gray-levels in the grayscale volume. The pruning process

[4] All experiments were performed on a PC with a 3GHz Pentium-D CPU and 4GB of memory (our implementation runs on a single thread, thus utilizes only one of the cores of the CPU)



(a) Volume

(b) Geometric skeleton

(c) Cross section of the volume

(d) Cross section of the skeleton

Figure 2.10: A CT scan of a human foot (a), the skeleton generated by our algorithm (b) that captures the cortical bones as surfaces, and a cross-section view (c,d).

has a time complexity of $O(\varepsilon s)$ where s is the number of voxels in the initial skeleton, and ε is the minimum size of the curve or surface feature in the final skeleton.

2.6 Conclusion and discussion

In this chapter, we discussed an innovative approach for skeletonization of density-like grayscale volumes, for the purpose of shape understanding. Our method does not require any explicit segmentation of the volume, is robust under the presence of noise, and is capable of extracting skeletal surfaces and curves corresponding to plate-like and rod-like

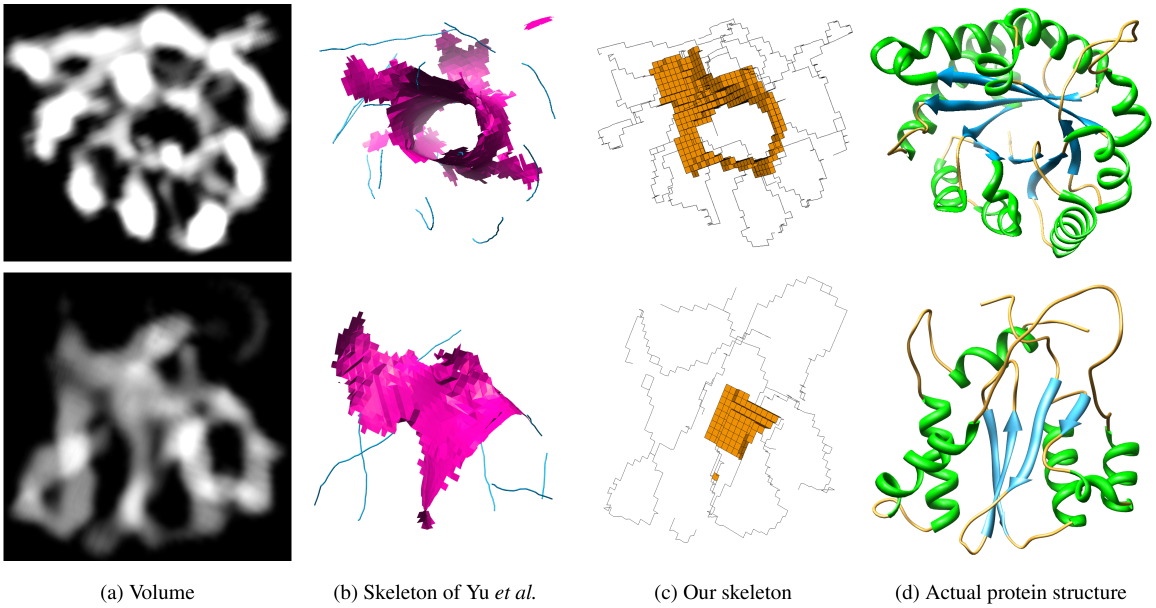

(a) Volume (b) Skeleton of Yu *et al.* (c) Our skeleton (d) Actual protein structure

Figure 2.11: Simulated cryo-EM volumes of proteins 1TIM (above) and 2ITG (below) at 8Å resolution (a), skeletons computed by the method of Yu *et al.* [124] (b) and our method (c), and the ground-truth structure of these proteins (d).

grayscale shape components. We tested our technique on synthesized and medical data-sets to demonstrate its behavior in different application domains.

While emphasizing shape representation, the resulting skeleton of our method may not exhibit the desired topology of the imaged subject. For example, as can be observed in the top left curve of Figures 2.11c and 2.11d, the skeleton contains extraneous loops and broken curves. This topological noise is mainly due to the sensitivity of morphological thinning to image noise in the initial skeletonization stage. Unlike a solid model, the topology of the imaged subject in the grayscale volume is not well defined, and a correct topology often needs to be defined by a human expert. We were able to overcome this limitation using an interactive skeletonization method we developed [2] that allows the user to point, click and sketch using a mouse and a 2D screen, to quickly create a visually accurate geometric skeleton. This method also uses the structure tensor based scoring function to better analyze the underlying shape of the density volume, and suggest possible skeletal paths that satisfy the interactive user constraints.

Limitations: The assumptions based on the observations of Section 2.1.1 limit the applicability of our technique to density-like grayscale volumes where features of interest are in high-density areas. Although typical in medical imaging, they may not apply to other data (e.g., a photograph of a scene). We would like to explore the extension of our algorithm to a more general set of data. One possible solution is to explore mapping functions that convert grayscale volumes in a different form to those satisfying our assumptions. However, the assumptions hold true for density volumes obtained using Cryo-EM, and therefore has been effectively used in literature for secondary structure identification purposes [126, 28] via SSEHunter [9].

As described in the earlier section, the performance of the initial skeletonization step is dependant on the number of distinct gray-levels of the volume. While the performance can be improved by discretizing the range of gray-levels, performing iterative thinning for each single gray-level is still time-consuming. To make the process more efficient, a possible alternative that can be explored is to perform only one iterative thinning step from low-density regions to high-density regions, while adjusting the shape and topology preservation criteria in binary thinning to the grayscale data.

Chapter 3

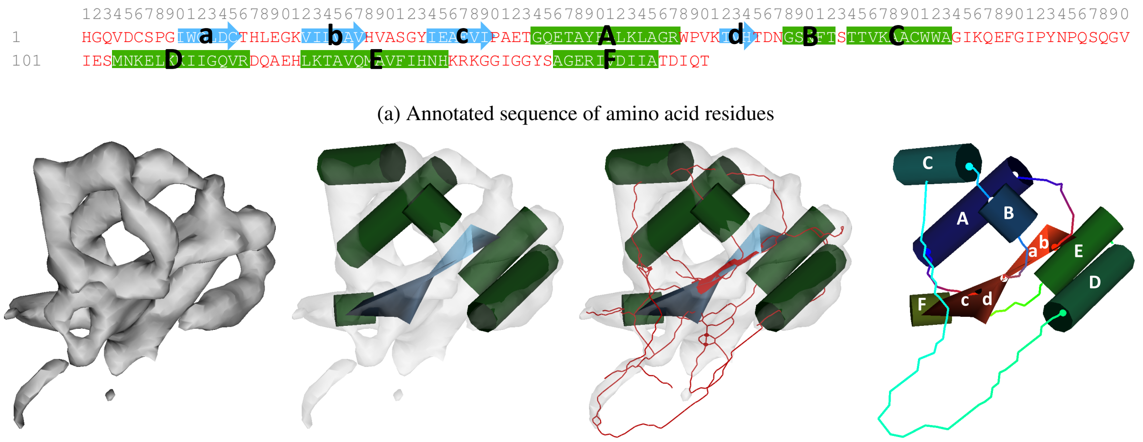
Correspondence between observed and predicted SSEs

3.1 Introduction

As discussed in the first chapter, the state of the art in cryo-EM based single particle reconstruction [130] can only produce density volumes at intermediate resolutions, and thus cannot be directly used to determine the locations of amino acid residues. However, as seen in Figure 3.1b, secondary structure elements are easily observed at intermediate resolutions due to their characteristic tubular and plate-like shape. This has led to the development of many manual [131] and automatic techniques such as SSEHunter [9], HelixHunter [51], SheetMinter [58] and SheetTracer [59] that use geometric skeletons, template-based cross correlation and heuristics to locate the *observed* SSEs within the density volume. Figure 3.1c displays the results of one such method (SSEHunter). For more details on detecting SSEs in cryo-EM density volumes readers are directed towards the comprehensive survey by Chiu *et al.* [25].

With the use of modern large scale DNA sequencing efforts such as the Human Genome Project [84], obtaining the sequence of amino acid residues of a protein has become a very accurate and efficient task. Subsequently, techniques such as PSIPred [53], JPred [27] (Figure 3.1a), Scratch [24] and many others have been developed that accurately and efficiently *predict* which amino acid residues in the sequence that might form SSEs.

Finding the correspondence between these *observed* SSEs in the volume and the *predicted* SSEs in the sequence is very important for molecular modeling as it can be used to formulate an initial pseudo-backbone of the protein, shedding light on its actual structure.


1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
1 HGQVDCSPGIWQLDCTHLEGKVIIVAVHVASGYIEAEVIPAETGQETAYFLKLAGRWPVKTVHTDNGSNFTSTTVKAACWWAGIKQEFGIPYNPQSQGV
101 IESMNKELKKIIGQVRDQAEHLKTAVQMAVFIHNHKRKGGIGGYSAGERIVDIIATDIQT
a
b
c
A
d
B
C
D
E
F
C
A
B
a
b
E
D
F
c
d

(a) Annotated sequence of amino acid residues

(b) Density volume

(c) Secondary structure elements

(d) Geometric skeleton

(e) SSE Correspondence

Figure 3.1: The inputs to our method are the protein sequence where α-helices (green) and β-strands (blue) are *predicted* using JPred [27] (a), and the 3D volume obtained by cryo-EM (b), where possible locations of SSEs have been detected using SSEHunter [9](c). Our method computes the correspondence between the two sets of SSEs (e) by matching the 1D sequence with a skeletal representation of the volume (d).

3.1.1 Problem statement

As a result, the computational problem that we will address here is the *correspondence* between the SSEs predicted from the sequence, and the ones observed in the density volume. As illustrated in Figure 3.1e, such a correspondence would establish a coarse 3D protein structure consisting of a chain of helices that sheds light on how the protein folds in 3D. It is important to note that this correspondence may not be a bijection. Due to noise in a typical density volume, an SSE detection algorithm may fail to find the locations of all the SSEs within that volume and may also identify false SSEs.

In the past, the SSE correspondence problem has only been studied in the work of Wu *et al.* [122]. Wu employed an exhaustive combinatoric search to find, amongst all permutations of SSEs in the density volume, an ordering that best matches the protein sequence. Note that this brute-force algorithm has a factorial time complexity. According to their experiments, this method is only practical for very small inputs, taking 1.5 hours and 16 hours to find the correspondence of a 3-helix and 8-helix protein respectively. In the first version of our work [3] we achieved much better performance (i.e. 5 seconds for a 20-helix protein) by formulating the correspondence problem as a subgraph-isomorphism. However, that method focused on only the α-helix correspondence, and therefore could not be used to generate accurate pseudo-backbones for proteins with β-sheets.

3.1.2 Method

The central theme of our approach is to cast the SSE correspondence problem as that of *shape matching* between the 1D sequence and the 3D volume. The key observation is that the search space of possible SSE correspondences can be much reduced if the shape of the density volume and the sequence are taken into consideration. That is, the successive SSEs in the density volume must be connected by paths through high-density regions, and the lengths of these SSEs and loops must match those in the sequence.

The key that makes such a matching possible is the modeling of both the 1D and 3D shapes as graphs that encode the lengths of SSEs as well as their connectivity. In particular, the graph representing the density volume is obtained by computing a *geometric skeleton* using the method described in Chapter 2 that encodes the topology of the high-density regions (Figure 3.1d). Using the shape representations, SSE correspondence reduces to a constrained error-correcting graph-matching problem, that seeks the best-matching simple paths among two graphs. Using a best-first search algorithm, the optimal match can be found in an efficient manner.

When applied to an extensive suite of test data, our method was shown to be capable of identifying the correct SSE correspondence with no or minimal user-intervention for small and medium size proteins. For example, Figure 3.1e shows the correspondence computed by our method for the 2ITG protein of the Human Immunodeficiency Virus (HIV). Our shape-matching approach improves the efficiency of an otherwise exhaustive search [122] by several orders of magnitude, obtaining the correspondence of proteins with more than 25 SSEs in under 40 seconds. In addition, the availability of the skeleton allows us to plot a path on the skeleton that connects successive SSEs, suggesting a possible pseudo-backbone of amino acid residues.

3.1.3 Contributions

In summary, we see our work making the following contributions to shape modeling, matching and computational biology:

- We introduce a common shape representation for both protein sequences and density volumes as attributed relational graphs, that is suitable for structural matching.

- We formulate a constrained error-correcting matching problem between attributed graphs, that differs from previously known exact and inexact matching problems. In addition, we develop an optimal solution based on a best-first search.
- We present a novel and efficient computational approach for solving an open problem in structural biology, that achieves orders of magnitude speedup over the best available method and makes model building from cryo-EM volumes much easier for medium-size proteins.

3.2 Previous work

Shape representation for matching: Shape representations, or *descriptors*, have been widely employed in graphics and computer vision for matching purposes. Generally, such representations can be classified into two classes. *Global* shape representations, often used in shape retrieval from a large repository of models, aim at computing a compact set of feature vectors of an entire object for fast comparison between objects [22, 96, 127]. We would refer interested readers to the survey [96] for descriptions and comparisons of these descriptors. Note that these global descriptors seldom provide local feature information and are thus generally unsuitable for partial matching; that is, finding a portion of an input object that matches a model object.

In contrast, *local* shape representations describe geometric features of an object (possibly at multiple scales) and are designed for partial matching and object alignment. Some examples of local descriptors include SIFT features [67], local spherical harmonics [41], salient surface features [42], curvature maps [43], and skeletons [99]. In this paper, we utilize the skeleton descriptor to translate the shape of an iso-surface in the density volume into a graph structure that can be used to identify connectivity among helices. Such a skeleton can be efficiently generated from a discrete volume by iterative thinning [11, 16, 79, 100]. For our experiments, we first generate a binary skeleton using the method of Ju et al. [54], and thereafter improve its connectivity using the grayscale skeletonization method [4] described in Chapter 2.

Graph matching: In pattern recognition and machine vision, graphs have long been used to represent object models such that object recognition reduces to graph matching. Here we only give a brief review of graph matching problems and methodologies and refer the reader to the excellent surveys [20, 29] for the rich volume of matching techniques.

In general, graph matching problems can be divided into exact matching and inexact matching. Exact matching aims at identifying a correspondence between a model graph and (a part of) an input graph, that can be solved using sub-graph isomorphism [110, 30] or graph monomorphism [118]. However, since real-world data is seldom perfect and noise-free, inexact or error-correcting matching is desired in a large number of applications. As in the work of Bunke [18], error-correcting matching can be formulated as finding the bijection between two subgraphs from the model and input graph that minimizes some error function. This error typically consists of the cost of deforming the original graphs to their subgraphs and the error of matching the attributes of corresponding elements in the two subgraphs. Note that, in most applications, the topology of the optimally matching subgraphs (e.g., whether it is connected, a tree, a path, etc.) is generally unknown. Such matching is said to be *un-constrained*, since the minimization of the error function is the only goal.

The most popular algorithms for error-correcting graph matching are based on best-first and A* searches [77]. These algorithms are optimal in the sense that they are guaranteed to find the global optimal match. However, since the graph matching problem itself is NP-complete, the actual computational cost can be prohibitive for large graphs. To this end, various types of heuristic functions have been developed to prune the search space [108, 93, 19, 91, 118]. Other methods such as simulated annealing [44], neural networks [38], probabilistic relaxation [26], genetic algorithms [113], and graph decomposition [73] can also be used to reduce the computational cost. Observe that all of these optimization methods are developed for un-constrained matching where the matched subgraphs can assume any topology.

For our problem domain, we know that the sequence will always be a linear chain of connected secondary structure elements. We can use this observation to develop a specialized form of subgraph isomorphism that benefits from this reduced search space.

3.3 Overview

As mentioned earlier, the central theme of our approach is to cast the SSE correspondence problem as a specialized form of subgraph-isomorphism. We first analyze the sequence of amino acid residues together with its predicted SSE annotations to construct a *Protein sequence graph* that is very sparse and linear. The *Density volume graph* is constructed by analyzing the observed SSEs in the density volume, and by using the geometric skeleton to identify their possible connectivity. Due to noise and the lack of high resolution in cryo-EM densities, the geometric skeleton can often have many alternate paths, and therefore, this graph is most often dense in nature. Section 3.4.1 describes in detail how each of these graphs are constructed.

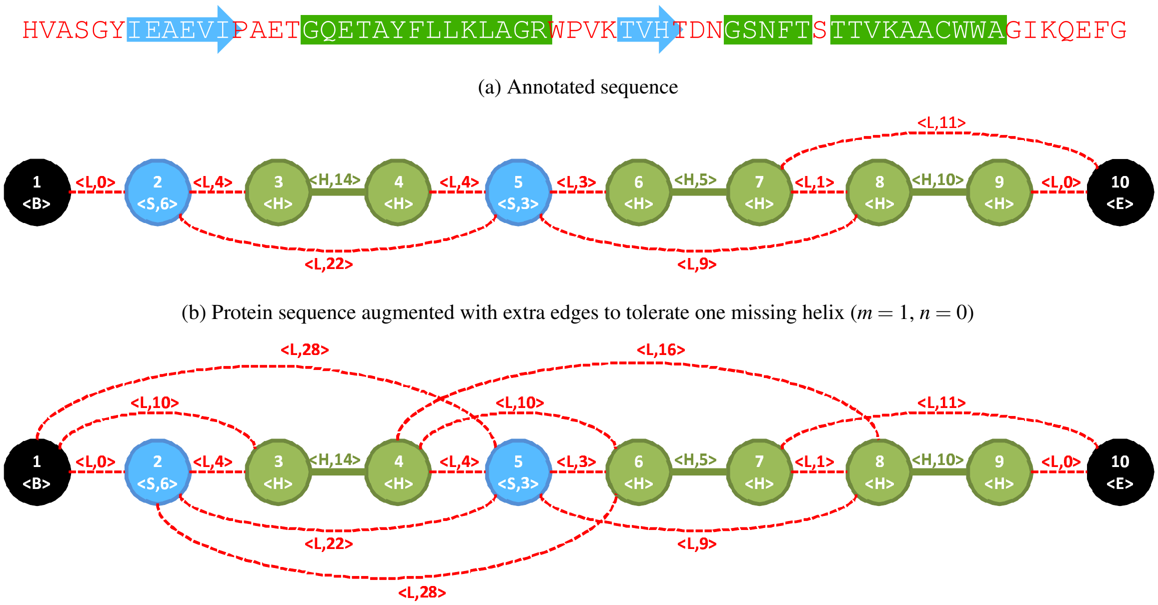
Once the graphs have been constructed, our task is to find how we can map the protein sequence graph to the density volume graph while being robust to SSE detection errors as well as connectivity errors in the density volume. For this purpose, we use a best-first search algorithm complemented by an SSE attribute based cost function to quickly find a set of likely correspondences. Furthermore, due to the nature of the best-first search, we are guaranteed that the solutions found are indeed ones that have the globally minimum costs [77]. Section 3.4.2 describes this search and associated cost functions in detail.

3.4 Method details

3.4.1 Shape representation using graphs

To solve the SSE correspondence problem as stated in Section 3.1.1, we first seek a common shape representation of both the 1D protein sequence and the 3D density volume that is suitable for matching. In particular, such representation should encode the lengths of each helix and strand as well as their connectivity. Here we introduce such a representation using attributed relational graphs (ARG).

In general, an ARG G consists of a 4-tuple $< V_G, E_G, \alpha_G, \beta_G >$, where V_G is a non-empty set of vertices ($|V_G|$ denotes the number of vertices), $E_G = V_G \times V_G$ is a set of edges between pairs of vertices, and α_G, β_G are attribute functions respectively on vertices and edges. Below we detail the meaning of these graph components when describing a protein sequence or a density volume, and conclude this section with a brief summary. Note that the graphs are specifically designed to tolerate the low-resolution and noise in a density volume.



(a) Annotated sequence

(b) Protein sequence augmented with extra edges to tolerate one missing helix ($m = 1$, $n = 0$)

(c) Protein sequence augmented with extra edges to tolerate one missing helix and one missing sheet ($m = 1$, $n = 1$)

Figure 3.2: The sequence of amino acid residues making up the 2ITG protein of the HIV virus (a), and the corresponding attributed relational graphs that can tolerate up to 1 missing sheet (b), or one missing helix and one missing sheet (c) where the vertices and edges have been colored by their attributes. Portions of the sequence have been omitted for demonstration purposes, for the full sequence please refer to Figure 3.1a.

Protein sequence graph

We represent the α-helices and β-strands in the primary sequence using a collection of vertices and edges in the protein sequence graph. Each helix is denoted by two vertices, and each strand by one[5]. These vertices are augmented by two additional terminal vertices denoting the two ends of the protein. To reflect the linearity of the sequence, we index the vertices in V_S in ascending order $\{1,\ldots,2r+s+2\}$ where r is the total number of helices, s is the total number of strands, and 1 and $2r+s+2$ are the two termini of the protein. For matching purposes, the different types of vertices are also distinguished by their attributes: $\alpha_S(x)$. For each $x \in V_S$, $\alpha_S(x)$ returns a 2-tuple $\{\alpha_{S_1}(x), \alpha_{S_2}(x)\}$. The first attribute $\alpha_{S_1}(x)$ assumes H, S, B or E if x represents an end of a helix, a strand, the beginning or the end of the protein. The second attribute $\alpha_{S_2}(x)$ is applicable to only strand vertices, and maintains

[5] As we are interested in the directionality of the α-helix assignment we use two graph vertices that correspond to each end point of the Helix. On the other hand multiple β-strands are associated with a single β-sheet, and the lack of resolution does not allow us to determine directionality. Therefore, we denote β-strands with only a single vertex.

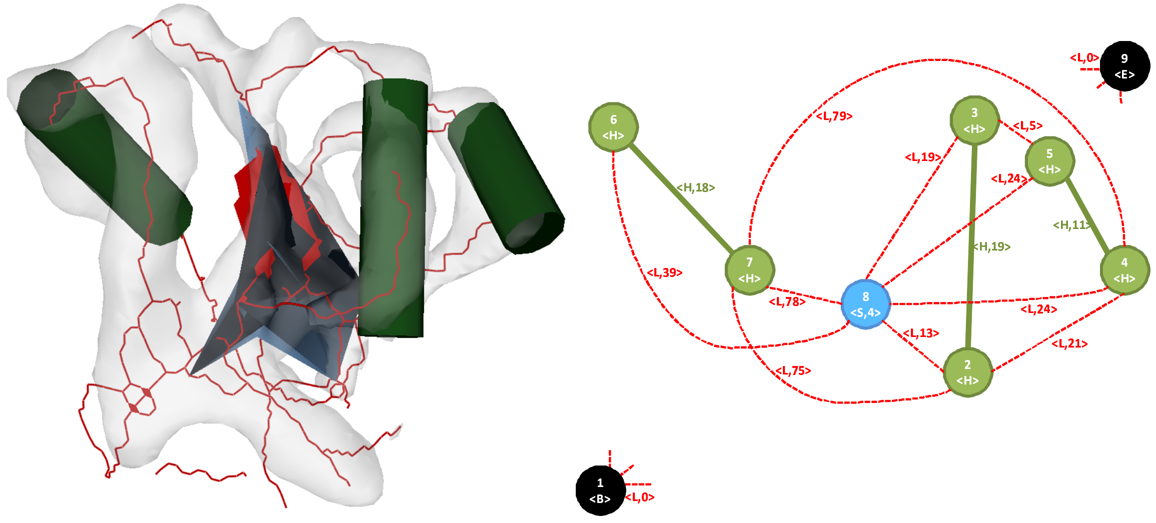
the strand length as the number of amino acid residues in the sequence. An example of vertices and attributes is shown in Figure 3.2b for the sequence in 3.2a.

To encode the lengths of SSEs and their connectivity, a *helix edge* is formed between the two ends of each helix, and a *loop edge* is formed between all other neighboring vertices in the sequence(as shown in Figure 3.2b). For strands, we do not annotate an edge, as the length information already encoded using the $\alpha_{S_2}(x)$ attribute described earlier. Note that these edges form a simple path with different edge types. The attribute function $\beta_S(x,y)$ for each edge $\{x,y\}$ returns a 2-tuple: $\beta_{S_1}(x,y)$ indicates the edge type, being H or L when $\{x,y\}$ is a helix edge or loop edge, and $\beta_{S_2}(x,y)$ maintains the length of that helix or loop as the number of amino acid residues in the sequence. Note that the graph is undirected, that is, $\beta_{S_k}(x,y) = \beta_{S_k}(y,x)$ for $k = 1,2$.

Due to the noisiness and the low resolution of the density volume, SSE detection in the volume may not be able to find all secondary structure elements of that protein. To be able to establish an error-correcting matching in the presence of missing elements, we augment the graph with loop edges bypassing a sequential group of α-helices and/or β-strands. For each sequential group of such SSEs beginning at vertex j and ending at vertex k, such a loop edge connects vertices $\{j-1, k+1\}$. We include loop edges bypassing $1\ldots m$ helices and $1\ldots n$ strands, where m and n are user-specified maximum numbers of helices and strands that could be missing in the volume. The attribute $\beta_{S_2}(x,y)$ for each new loop edge is set to be the total number of amino acids in the sequence bypassed by the edge. Figure 3.2b shows an example with $m = 1$, $n = 0$, and Figure 3.2c shows an example with $m = 1$ and $n = 1$. Note that after such an addition, any simple path in the graph connecting vertices with ascending indices still consists of a sequence of vertices and edges that represents an ordered subset of SSEs in the protein sequence. Due to the inherent limitations in accurately detecting SSEs in the density volume, we assume n to be ≥ 1 in our experiments (Section 3.5).

Density volume graph

As in the sequence graph, the volume graph C consists of two vertices for each detected α-helix, one vertex for each detected β-sheet, and two terminal vertices for the entire protein. The different types of vertices are distinguished using the vertex attribute function α_{C_1}, that assumes H, S, B or E for the helix vertices, sheet vertices, beginning vertex or end vertex



(a) Volume, skeleton and SSEs

(b) Density volume graph

Figure 3.3: The density volume, skeleton, and detected SSEs (a), and the corresponding attributed relational graph (b) where the two terminal vertices 1 and 9 are connected to every other vertex via loop edges. Three helices have been omitted for demonstration purposes, for the full annotation refer Figure 3.1d.

of the protein. In the case of sheet vertices, a second attribute α_{C_2} maintains the expected individual strand length for the given sheet, which is estimated by comparing the relative sizes of the different sheets. Unlike the sequence graph, where there is an explicit ordering of vertices, the indices of vertices in V_C do not imply any ordering.

To encode helix information, vertices representing the two ends of a helix are connected by a helix edge. As in the sequence graph, the edge attribute function β_C returns a 2-tuple, where β_{C_1} assumes H or L indicating a helix or loop edge, and β_{C_2} returns the length information. For a helix edge $\{x,y\} \in E_C$, $\beta_{C_2}(x,y)$ is the Euclidean length of the detected helix in the density volume, which can be normalized by the resolution of the volume to approximate the number of amino acids in the helix [9]. An example of such edges are shown in green in Figure 3.3b representing the helices detected in the density volume in 3.3a.

Unlike the sequence graph, the density volume does not explicitly provide the needed connectivity among detected helices and sheets. However, as stated earlier, two structures (helices and/or strands) at successive positions in the sequence are more likely to be connected in 3D through regions in the volume with high density. As a result, we seek a representation that depicts the topology of such high-density regions. To this end, we extract a morphological *skeleton* of the density using a combination of erosion-based binary

[54] and grayscale [4] skeletonization techniques. Such skeletons can be robustly generated even from noisy surfaces while preserving the solid topology, and an example is shown in Figure 3.3a.

Given the skeleton, we form loop edges as shown in Figure 3.3b. First, we link every two vertices in the graph that represent two different structures (helices or sheets) and are connected by a path on the skeleton, as long as the path does not pass through a helix. When multiple paths exist between two structures, the shortest is taken. Note that, due to noise present in the volume, these skeleton paths may not capture all the necessary connectivity among structures. To this end, we additionally create a loop edge between ends of every two structures whose Euclidean distance is within a user-specified value ε. Finally, to complete the graph, a loop edge is created between each terminal vertex and every non-terminal vertex. The edge attribute $\beta_{C,2}$ for the above three classes of loop edges are set to the length of the skeleton path, the Euclidean distance, and zero respectively (normalized by the resolution of the volume as in [9]).

Summary: Here we briefly summarize the common meanings of the graph components $< V_G, E_G, \alpha_G, \beta_G >$ in a sequence graph ($G = S$) and in a volume graph ($G = C$).

- Vertices V_G: A helix vertex represents one of the two ends of an α-helix. A strand vertex (in the sequence graph) represents a β-strand and a sheet vertex (in the volume graph) represents a β-sheet. A terminal vertex represents one of the two terminals of the protein.
- Edges E_G: A helix edge connects two ends of an α-helix. A loop edge connects ends of two structures (helices or sheets) or between a structure and a protein terminal.
- Vertex attribute:
  - $\alpha_{G_1}(x)$: Returns H, S, B or E if x is a helix vertex, strand vertex, the start terminal or the end terminal.
  - $\alpha_{G_2}(x)$: For a strand vertex in the sequence graph, returns the length of a strand. For a sheet vertex in the volume graph, returns the expected length of a strand.
- Edge attribute:
  - $\beta_{G_1}(x,y)$: Returns H or L if $\{x,y\}$ is a helix edge or a loop edge.

- $\beta_{G_2}(x,y)$: Returns the length of the edge $\{x,y\}$, measured as the number of amino acid residues (in the sequence graph) or as the expected number of amino acid residues in the skeleton path (in the volume graph).

3.4.2 Constrained graph matching

Given two graphs representing the secondary structure elements (helices and strands/sheets) in the sequence and in the volume, we show that finding the correspondence between the two sets of structures reduces to a constrained graph matching problem. We first define:

Definition 1 *A* chain *of an ARG G is a sequence of nodes $\{v_1,\ldots,v_n\} \subseteq V_G$ that form a path in G. A chain is* ordered *if $v_1 = 1, v_n = |V_G|$, and $v_i < v_{i+1}$ for all $i \in [1,n-1]$. A chain is* simple *if $v_i \neq v_j$ for all $i,j \in [1,n-1]$.*

For example, an ordered chain in the sequence graph consists of a sequence of nodes and edges depicting a linked sequence of helices and strands. A correspondence between structures in the sequence and the volume is therefore a bijection between an ordered, simple chain in the sequence graph and a chain in the volume graph. Note that the definition of chain allows establishing partial correspondence between a subset of the structures in both the sequence and the volume. More generally, the problem can be defined for any pair of attributed relational graphs:

Problem 1 *Let S,C be two ARGs. Find an ordered, simple chain $\{p_1,\ldots,p_n\} \subseteq V_S$ and chain $\{q_1,\ldots,q_n\} \subseteq V_C$ that minimize the matching cost:*

$$\sum_{i=1}^{n} c_v(p_i,q_i) + \sum_{i=1}^{n-1} c_e(p_i,p_{i+1},q_i,q_{i+1}) \tag{3.1}$$

where c_v, c_e are any given functions evaluating the cost of matching node p_i with q_i or edge $\{p_i,p_{i+1}\}$ with $\{q_i,q_{i+1}\}$.

Comparing to previously studied graph matching problems such as exact graph (or subgraph) isomorphisms, inexact graph matching and maximum common subgraph problems [47], Problem 1 is unique in that it seeks best-matching subgraphs from two graphs that have a particular shape. Given such constraints, previous graph matching algorithms that are guided only by error-minimization can not be directly applied.

Cost functions

Here we explain our choice for the two cost functions c_v, c_e in Equation 3.1 when matching the sequence graph and the volume graph. Note that the algorithm we present in the next section works for any non-negative cost function.

Each cost function measures the similarity of the attributes associated with two vertices or two edges. The vertex cost function has two purposes: it ensures that two matched vertices are of the same type, and for a strand-sheet vertex pair, it computes the difference between the length of the strand and the expected strand length for that sheet. The vertex cost function is defined as:

$$c_v(x,y) = \begin{cases} |\alpha_{S_2} - \alpha_{C_2}|, & \text{if } \alpha_{S_1}(x) = \alpha_{C_1}(y) = \text{`S'} \\ 0, & \text{if } \alpha_{S_1}(x) = \alpha_{C_1}(y) \neq \text{`S'} \\ \infty, & \text{otherwise} \end{cases} \tag{3.2}$$

The edge cost function enforces type matching and computes the length difference between two helix edges or two loop edges, and is defined as:

$$c_e(x,y,u,v) = \begin{cases} |\beta_{S_2}(x,y) - \beta_{C_2}(u,v)|, & \text{if } \beta_{S_1}(x,y) = \beta_{C_1}(u,v), \\ & \text{and } y = x+1. \\ |\beta_{S_2}(x,y) - \beta_{C_2}(u,v)| + \gamma_S(x,y), & \text{if } \beta_{S_1}(x,y) = \beta_{C_1}(u,v), \\ & \text{and } y > x+1. \\ \infty, & \text{otherwise.} \end{cases} \tag{3.3}$$

Here, the γ_S term penalizes missing helices and sheets in the volume graph and is set to be a weighted sum of the length of helices and strands bypassed by a link edge. For a link edge in the protein sequence connecting nodes x and y, we compute the penalty as:

$$\gamma_S(x,y) = \omega_h \sum_{\substack{x < i < y-1, \text{ and} \\ \beta_{S_1}(i,i+1) = \text{`H'}}} \beta_{S_2}(i,i+1) \quad + \quad \omega_s \sum_{\substack{x < i < y, \text{ and} \\ \alpha_{S_1}(i) = \text{`S'}}} \alpha_{S_2}(i) \tag{3.4}$$

where ω_h and ω_s are user-specified weights that adjust the influence of missing helices and missing strands in this penalty term.

An optimal best-first search algorithm

In this section, we present a best-first search algorithm for solving Problem 1. Our method extends the tree-search paradigm popularized in computing unconstrained error-correcting graph matching and is guaranteed to find the optimal match.

To find a match between two graphs, a tree-search algorithm starts out from an initial, incomplete match and incrementally builds more complete matches. To find matching chains in graphs S,C, we first consider a partial match as a sequence of node-pairs

$$M_k = \{\{p_1,q_1\},\ldots,\{p_k,q_k\}\}$$

where $\{p_1,\ldots,p_k\}$ and $\{q_1,\ldots,q_k\}$ are the initial portion of some ordered, simple chain in S and some chain in C. Based on the definition of chains and our matching goal of minimizing cost functions, elements of M_k must satisfy the following requirements:

- **Vertex requirement**: For all $i \in [1,k]$:

  $$p_1 = 1, \quad p_i \in V_S, \quad q_i \in V_C, \quad \text{and} \quad c_v(p_i,q_i) \neq \infty,$$

  and for all $j \in [1,k], \quad i \neq j, \quad \alpha_{C_1}(j) \neq$ 'S':

  $$q_i \neq q_j.$$

  In other words, the only vertices that may repeat in M_k are sheet vertices in the volume graph, and vertices in each pair must be of the same type.

- **Edge requirement**: For all $i \in [1,k-1]$:

  $$p_i < p_{i+1}, \quad \{p_i,p_{i+1}\} \in E_S, \quad \{q_i,q_{i+1}\} \in E_C, \quad \text{and} \quad c_e(p_i,p_{i+1},q_i,q_{i+1}) \neq \infty.$$

  In words, $\{p_1,\ldots,p_k\}$ must form an ordered chain, and the two edges connecting the two nodes in neighboring pairs in M_k must be of a same type.

Starting with an empty match $M_0 = \emptyset$, the search algorithm incrementally builds longer matching chains. Specifically, we define an *expansion* of a partial match M_k as a new partial match $M_{k+1} = M_k \cup \{\{p_{k+1},q_{k+1}\}\}$ such that the added nodes p_{k+1},q_{k+1} satisfy the node requirement and the added edges $\{p_k,p_{k+1}\},\{q_k,q_{k+1}\}$ (for $k > 0$) satisfy the edge

```
// Finding the optimal common chain in $S, C$

$ChainMatch(S, C)$
// $Q$ is a min heap
// The key of each element $M \in Q$ is $g(M)$
    $Q \leftarrow \{M_0\}$
    Repeat
        $M_k \leftarrow$ Pop($Q$)
        // $M_k$ has the form $\{\{p_1, q_1\}, \ldots, \{p_k, q_k\}\}$
        If $p_k = |V_S|$
            Return $M_k$
        Repeat for each expansion $M_{k+1}$ from $M_k$
            Insert($Q, M_{k+1}$)
```

Figure 3.4: Pseudo-code for our best-first search based algorithm.

requirement. Note that usually a M_k can be expanded into multiple M_{k+1}. A match M_k is *complete* (i.e., no more expansion can be done) if $p_k = |V_S|$.

Observe that the search procedure essentially builds a tree structure with M_0 at the root of the tree, expanded partial matches M_k at the *k*th level of the tree, and complete matches at the tree leaves. Our goal is therefore to find the complete match that minimizes the matching error defined in Equation 3.1.

Best-first search

To avoid an exhaustive tree search to find the optimal complete match, we adopt the best-first search algorithm that prioritizes the expansion of incomplete matches using the cost function. We denote the cost function for partial match M_k as $g(M_k)$. The best-first search algorithm works by maintaining all un-expanded partial matches in a priority queue and only expanding the partial match with the best (smallest) cost function value. Figure 3.4 outlines the pseudo-code of the algorithm.

Observe from Figure 3.4 that the algorithm returns the first complete match that it finds. Due to the nature of the node expansion, where the current lowest cost match is always expanded first, the first complete match is guaranteed to be the *optimal* match. For our purposes, we extend the algorithm to continue expanding after the first match is found. Based on the same best-first proof, the subsequent match (and the next match thereafter, *ad*

Protein	Volume Size	Sequence		Density Volume		
	(d^3)	Helix count	Strand count	Sheet count	Missing helices	Missing sheets
1UF2	96	4	-	-	-	-
2ITG	64	6	4	1	-	-
1IRK	96	9	9	3	1	1
1WAB	64	9	5	1	2	-
1DAI	64	9	9	3	-	2
1BVP	128	10	14	3	-	-
3LCK	64	12	7	2	5	-
1TIM	96	12	8	1	3	-
GroEL (Apical Domain)	100	5	8	2	-	-
RDV P8	96	14	8	3	2	1

Table 3.1: Data used to evaluate our method for finding the correspondence between SSEs.

infinitum) is guaranteed to be the next-best match. This allows us to find a set of the lowest cost matches, ordered by the ascending order of their costs.

3.5 Results

In this section, we discuss the performance of our method on an extensive suite of protein data. For the majority of these data sets, we observed that our method was capable of finding the correct SSE correspondences without any user intervention. However, for density volumes of poor quality, the optimal graph matching may not represent the actual SSE correspondence, and user-interaction is needed to yield the correct result. For this purpose, we present the user with statistical information gathered from the top matches, to give them a better idea of the most likely interaction choices.

3.5.1 Setup

Our experiment consists of ten cryo-EM volumes at 4Å-10Å resolution, eight of which are simulated from the actual atomic model obtained from the Protein Data Bank [34] and two which are authentic cryoEM reconstructions (RDV P8 at 6.8Å, GroEL-Apical domain at 4.2Å[6]). These structures, while not an exhaustive representation of those found in the Protein Data Bank, do represent commonly occurring folds of the major families of protein structure. Table 3.1 shows the number of helices and strands in the protein sequence, the number of sheets in the density, and the number of missing helices and sheets in the density (given as the parameters m and n when creating the sequence graph).

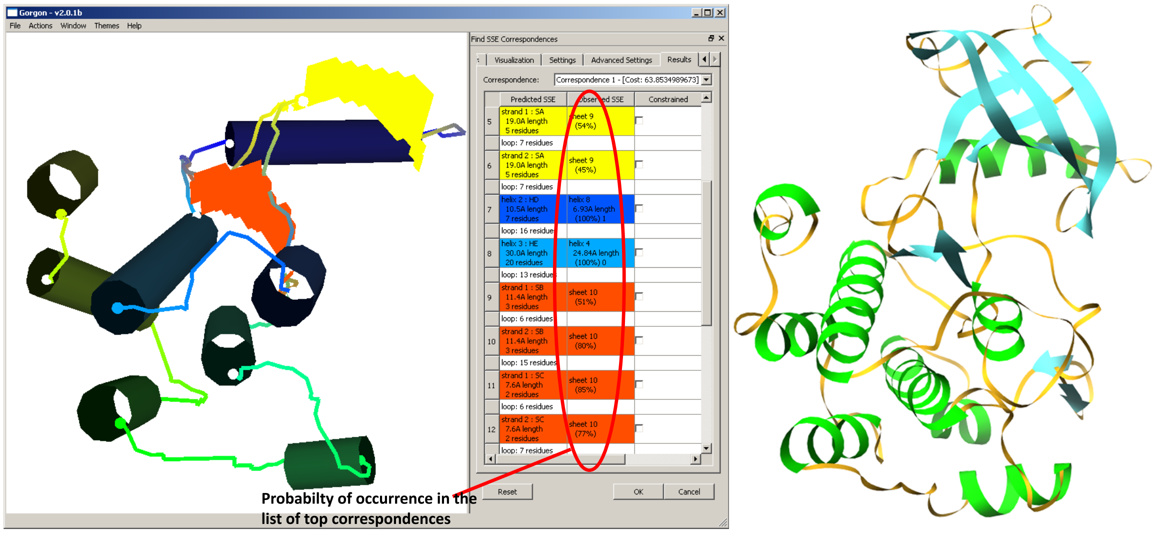
[6] EMDB numbers for these authentic reconstructions are 1060 (RDV P8) and 5001 (GroEL)

In each example, we utilize the protein sequence data from the Protein Data Bank, the SSEs in density volumes detected using SSEhunter [9], and the skeleton created using the methods of Ju *et al.*[54] and Abeysinghe *et al.*[4] (Chapter 2). The matching result is presented as a correspondence between SSEs in the sequence with those in the density volume. In our most noisy data set, (RDV P8) an Euclidean distance threshold of $\varepsilon = 10\mathring{A}$ was used for creating extra edges in the volume graph to allow for missing connectivity in the geometric skeleton. In all our experiments the missing helix and sheet penalty terms in equation 3.4 are set to $\omega_h = 5, \omega_s = 5$.

3.5.2 Evaluation method

To evaluate the accuracy of matching, we compare the SSE correspondence computed by our method with a manual labeling of the SSEs in the density volume based on the known atomic structure (for simulated data) or a structural homologue (for authentic data). To improve on a binary evaluation of whether or not our method finds the correct correspondence, we compute a list of candidate correspondences between the SSEs in the sequence graph and the volume graph, ranked by their matching costs. This can be done easily in the best-first search framework by terminating the search only after a number of complete matches (e.g., 35) have been found. In our experiments, the accuracy of our method is reported as the ranking of the manual-labeled correspondence in this candidate list.

Quantifying the benefit to the user: In Figure 3.5, we see our method used to identify the SSE correspondence for the 1IRK protein. As we will discuss later, this type of data set is challenging due to missing SSE elements and similar lengths of loops, strands and helices. Although we do not detect the correct strand correspondence in our candidate list, we see that the pseudo-backbone generated for the first candidate is almost identical to the ground truth. Furthermore, we provide users with statistics (probability of occurrence of a given pairing in the list of top candidates) that allow them to make reasonable assumptions when interactively refining the search. Due to this reason, the rank by itself is not a good indicator of the accuracy of our method, and we make use of the evaluation function E_s (described below) which is a better estimate of accuracy that also captures the interactive nature of our approach.



Correspondence: Correspondence 1 - [Cost: 63.8534989673]

	Predicted SSE	Observed SSE	Constrained
5	strand 1 : SA 19.0A length 5 residues	sheet 9 (54%)	□
	loop: 7 residues		
6	strand 2 : SA 19.0A length 5 residues	sheet 9 (45%)	□
	loop: 7 residues		
7	helix 2 : HD 10.5A length 7 residues	helix 8 6.93A length (100%) 1	□
	loop: 16 residues		
8	helix 3 : HE 30.0A length 20 residues	helix 4 24.84A length (100%) 0	□
	loop: 13 residues		
9	strand 1 : SB 11.4A length 3 residues	sheet 10 (51%)	□
	loop: 6 residues		
10	strand 2 : SB 11.4A length 3 residues	sheet 10 (80%)	□
	loop: 15 residues		
11	strand 1 : SC 7.6A length 2 residues	sheet 10 (85%)	□
	loop: 6 residues		
12	strand 2 : SC 7.6A length 2 residues	sheet 10 (77%)	□
	loop: 7 residues		

Probabilty of occurrence in the list of top correspondences

(a) Results displayed as a pseudo-backbone, and the probabilities for each pairing

(b) Actual C_α backbone

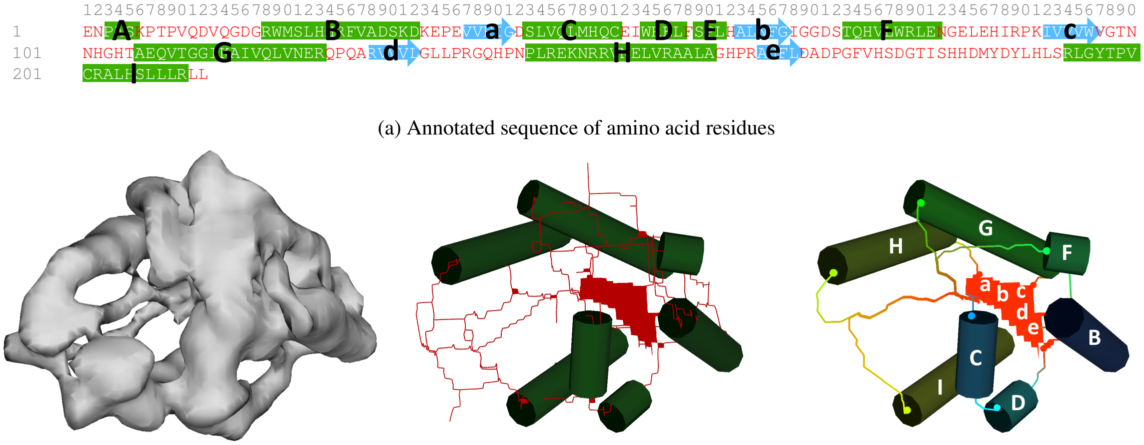
Figure 3.5: Our method used to identify the SSE correspondence of the 1IRK protein of the Human Insulin Receptor(a), where the pseudo-backbone is displayed on the left, and the individual SSE correspondences are displayed on the right together with the probability of that occurrence in the list of candidates. Even though the correspondence does not have a perfect sheet matching, the pseudo-backbone is almost identical to that of the ground truth (b).

Given the probability of occurrence[7] for each pairing $P(\{i,j\})$ of the sequence SSE i and the density SSE j, a naïve user-interaction method is to pick the highest probability pairing for each secondary structure element in the sequence. We compare the correspondence obtained from this method C_n against the ground truth C_g to form an evaluation E_p:

$$E_p(C_n, C_g) = \frac{1}{|C_g|} \sum_{i=1}^{|C_g|} \delta_{C_n[i], C_g[i]} \tag{3.5}$$

where $\delta_{a,b}$ is the Kronecker delta function $\delta_{a,b} = \begin{cases} 1, \text{if } i = j \\ 0, \text{if } i \neq j \end{cases}$. In other words, E_p is the fraction of correct individual SSE pairings if we pick the highest occurrence match for each SSE. Due to the binary nature of E_p this can be an overly conservative measure of accuracy. For example, for each SSE if the probability of the correct pairing is only slightly lower than that of an incorrect match, we would have $E_p = 0$. Due to this reason, we must consider the probability of the correct pairing compared against that of the highest-occurrence match.

[7] The probability of a pairing $P(a)$ is defined as the ratio between the number of occurrences of a in the list of candidates, and the total number of candidates (which in our experiments is set to 35).



(a) Annotated sequence of amino acid residues

(b) Density volume

(c) Detected SSEs and geometric skeleton

(d) SSE Correspondence

Figure 3.6: The annotated sequence of amino acid residues of the 1WAB protein (a), the density volume (b), the detected secondary structure elements together with the skeleton (c), and the correspondence between the two sets of SSEs computed as the optimal match between the sequence and volume graphs (d).

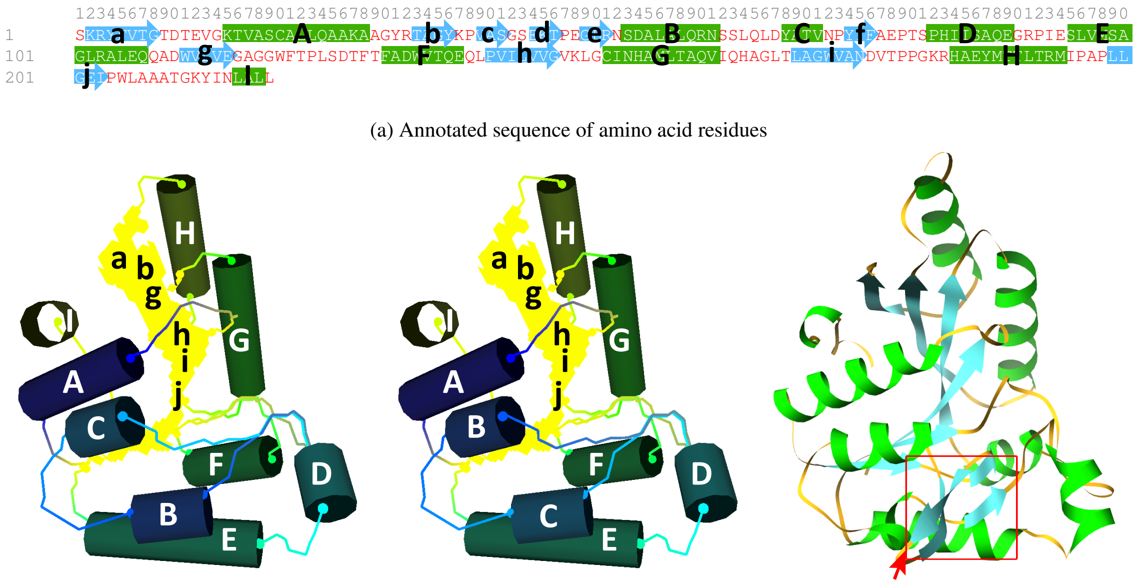
E_s ($E_p \leq E_s \leq 1$) achieves this task by allowing *partial credit* for each pairing.

$$E_s(C_n, C_g) = \frac{1}{|C_g|} \sum_{i=1}^{|C_g|} \frac{P(C_g[i])}{P(C_n[i])}. \tag{3.6}$$

In our experiments we have observed that an E_s score higher than 0.8 indicates a very good list of candidate matches that are only one or two assignments different from the ground truth. Table 3.2 evaluates the accuracy of our method by considering the rank, and the E_s score.

3.5.3 Unsupervised matching

Figures 3.1 and 3.6 show two examples (2ITG and 1WAB) where our method is able to identify a correct SSE assignment as a top-ranked candidate. Observe that our algorithm is robust to noise in the data, such as the two missing helices in the density volume of 1WAB. As a by-product of our algorithm, a pseudo-backbone can be visualized by rendering the skeleton paths represented by the graph edges in the optimally matching chain. This pseudo-backbone serves as a starting point when determining the actual C_α backbone as described in Chapter 5. In Figure 3.5, we compare the pseudo-backbone for the 1IRK protein of the Human Insulin Receptor with the actual C_α backbone. Observe that although



(a) Annotated sequence of amino acid residues

(b) Rank 1 correspondence

(c) Actual correspondence

(d) Two missing β-sheets

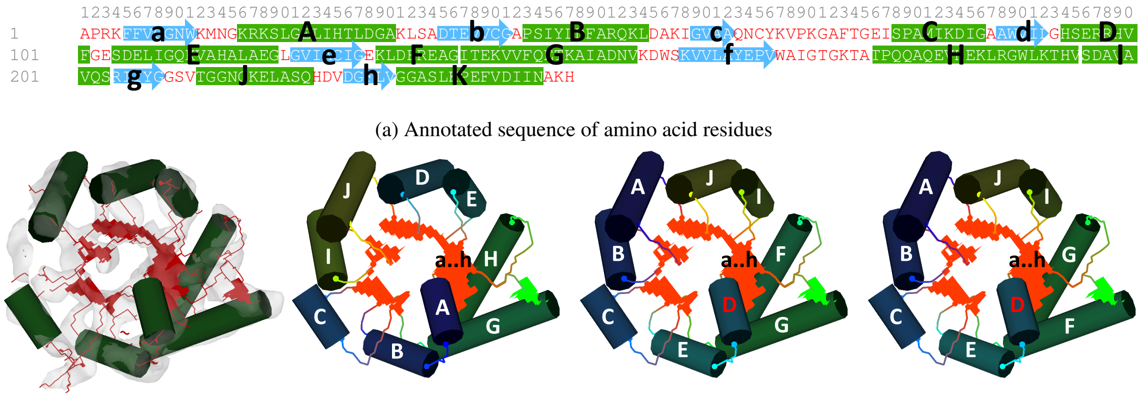
Figure 3.7: The annotated sequence of amino acid residues of the 1DAI protein (a), the optimal correspondence computed by our method (b) and the actual correspondence (c) which ranks 17th in the candidate matches. Observe that our method accurately identifies the missing sheet highlighted on the actual molecular model (d).

the top-ranking correspondence was not able to correctly identify the two missing β-sheets, the backbones are almost identical.

Figure 3.7 shows an example where the correct correspondence (Figure 3.7c) is ranked 17th in the candidate list, and differs from the top candidate (Figure 3.7b) by only the helix assignments *B* and *C*. These two helices exhibit very similar lengths and connectivity, illustrating why graph-matching alone cannot distinguish right from wrong without further user interaction. However, it is worthwhile to note that our method was able to accurately identify the two missing β-sheets highlighted in Figure 3.7d for all but two of the top 35 candidate matches. This is also reflected by the high E_s scores observed for this experiment in Table 3.2.

3.5.4 Interactive matching

In the case where the resolution of the density volume does not provide sufficient shape or topology information of the embedded protein, our shape-matching based approach may not produce the correct correspondence in the candidate list. Proteins that exhibit


1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
1 APRKFFVGGNWKMNGKRKSLGELIHTLDGAKLSADTEVVCGAPSIYLDFARQKLDAKIGVAAQNCYKVPKGAFTGEISPAMIKDIGAAWVILGHSERRHV
101 FGESDELIGQKVAHALAEGLGVIACIGEKLDEREAGITEKVVFQETKAIADNVKDWSKVVLAYEPVWAIGTGKTATPQQAQEVHEKLRGWLKTHVSDAVA
201 VQSTRIIYGGSVTGGNCKELASQHDVDGFLVGGASLKPEFVDIINAKH
a
A
b
B
c
C
d
D
E
e
F
G
f
H
I
g
J
h
K
J
D
E
I
a..h
H
C
A
B
G
A
J
I
B
a..h
F
C
D
G
E
A
J
I
B
a..h
G
C
D
F
E

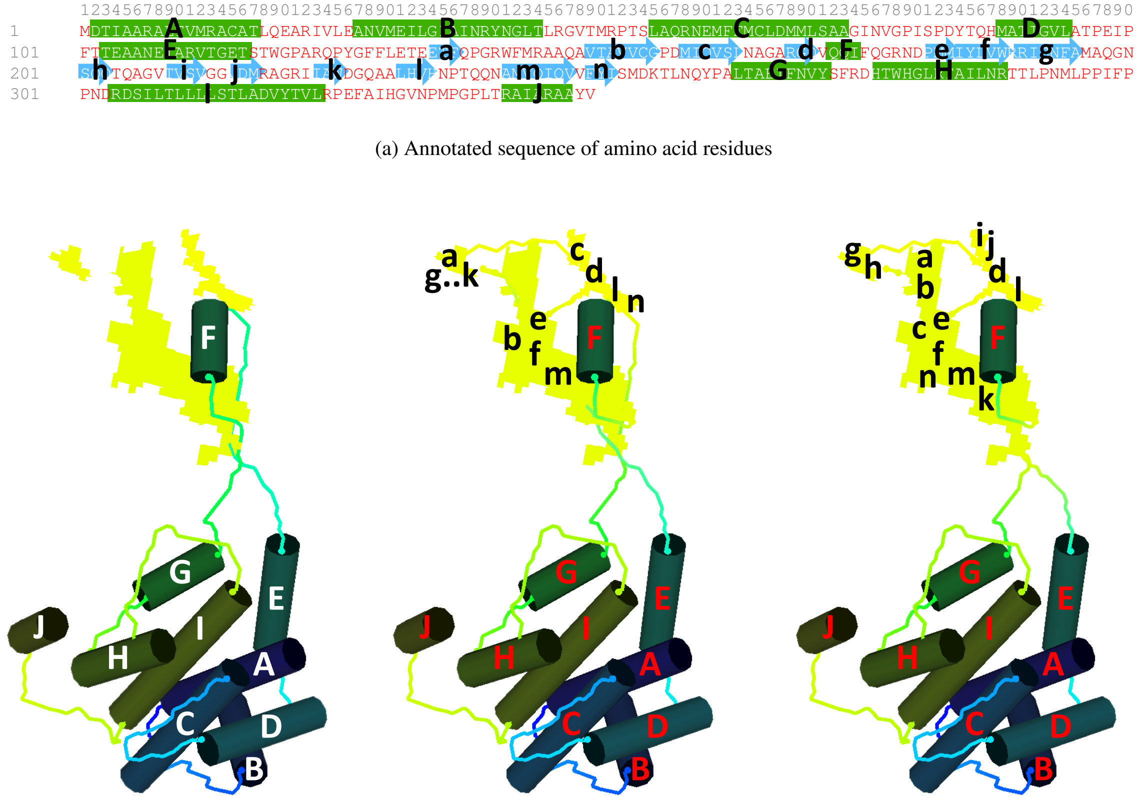
(a) Annotated sequence of amino acid residues

(b) Density, skeleton and SSEs (c) Optimal correspondence (d) Optimal after constraint (e) Actual correspondence

Figure 3.8: The annotated sequence of amino acid residues of the 1TIM protein (a), the optimal correspondence computed by our method (b) and the actual correspondence (c) which ranks 9th in the candidate matches. Observe that our method accurately identifies the missing sheet highlighted on the actual molecular model (d).

symmetrical elements such as similar-length helices that surround a beta barrel (Figure 3.8), or similar-sized β-sheets that are located close together (Figure 3.9) also pose a challenge to our method due the large amount of possible correspondences that have the same cost as defined in Equation 3.1. To overcome this limitation, we allow the user to manually assign matching constraints based on their biological knowledge of the spatial arrangement of SSEs. Specifically, the user may designate the correspondence between a small subset of helix edges and/or sheet vertices in the sequence graph and the volume graph. Such information can be translated into additional edge and/or node attributes (e.g. $\beta_{S,1}(\{x,y\}) = \beta_{C,1}(\{u,v\}) = H_k$ if edge $\{x,y\}$ and $\{u,v\}$ are the kth corresponding pair) to enforce such explicit matching in the best-first search.

Figure 3.8 shows an example of the 1TIM protein found in chicken muscle where the correct SSE correspondence was not found in the initial candidate list without any user constraints (we computed a list of 35 top matches, the optimal match is shown in Figure 3.8c). After a user specified constraint (D), the correct correspondence was found ranking 25th in the candidate list as shown in Figures 3.8d and 3.8e. The reason that graph matching was not able to identify the correct correspondence without user constraints is that the SSEs in the protein exhibit similar lengths and are spatially close-by, hence pose challenges to matching which is primarily based on SSE and loop lengths. Interactive constraints, that carry domain knowledge, can be used in these cases to provide anchor points to guide our method towards a more accurate correspondence. In this example, the constraint is picked



(a) Annotated sequence of amino acid residues

(b) Helix-only correspondence (c) Optimal correspondence after helix constraints (d) Actual correspondence

Figure 3.9: The annotated sequence of amino acid residues of the 1BVP protein (a), and the optimal correspondence computed by our method where the helix correspondence is correct (b). The optimal correspondence after the helices have been constrained (c), and the actual correspondence (d).

by analyzing the sequence to find the shortest of the only helix pair that has a loop segment separating them.

For proteins where the β-sheets are clustered away from the α-helices our method can predict the α-helix correspondence with very high accuracy. For example, for the 1BVP protein of the Blue-Tongue Virus shown in Figure 3.9, our method was able to predict the α-helix correspondence as the optimal cost match. However, due to the clustered nature of the β-strands, the correct β-strand correspondence does not appear in the top list of candidates. For proteins of this nature, we provide the user with a 2-stage interaction approach, where they can first find the α-helix only correspondence. Once they are satisfied with the helix correspondence, they can constrain the helices and continue to search for the β-strand correspondence guided by the probabilities of occurrence (Figure 3.5).

Protein	User	Helix only		Helices and Strands			Execution time (s)	
	Constraints	Rank	E_s	Rank	E_s(Strand only)	E_s(Composite)	First match	Top 35 matches
1UF2	-	1	0.96	1	-	0.96	0.00	0.00
2ITG	-	1	1.00	1	1.00	1.00	0.00	0.01
1IRK	-	1	1.00	>35	0.76	0.92	0.65	0.82
1WAB	-	11	0.89	11	1.00	0.91	0.65	1.14
1DAI	1	17	0.82	17	1.00	0.89	0.21	0.32
1BVP	-/10*	1	1.00	>35	0.58	0.83	0.86	1.50
3LCK	-	7	0.95	>35	0.71	0.89	17.01	21.04
1TIM	1	25	0.91	25	1.00	0.93	11.60	15.41
Groel	-	1	1.00	>35	0.90	0.95	0.07	0.11
RDV P8	6	12	0.96	>35	0.88	0.94	36.54	38.32

Table 3.2: The number of user constraints specified for each experiment, the rank and E_s score when considering only the α-helix correspondence, the rank and E_s scores when considering both α-sheet as well as β-strand correspondences, and the execution times to find the first correspondence and the list of the top 35 candidates. *For 1BVP the experiment was conducted by first finding the helix correspondence using no constraints, and then constraining all the helices to find the β-strand correspondence.

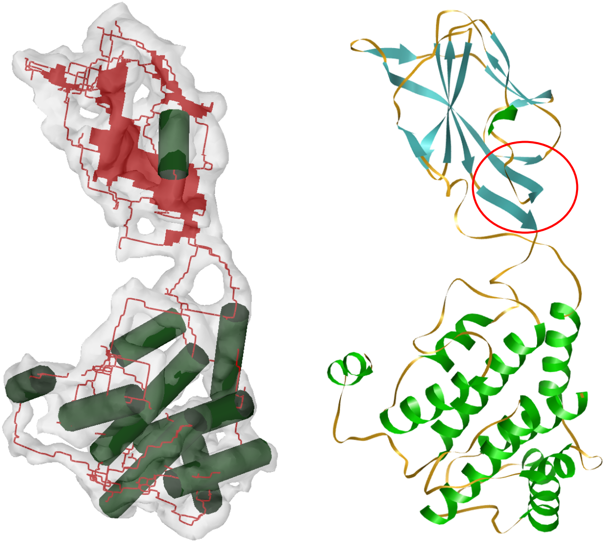
Due to the accumulation of error in the search process because of the inherent ambiguities present in low-resolution imaging, we observe that the amount of user constraints needed to obtain a high-ranked correct correspondence increases with the decreasing accuracy (in terms of resolution as well as noise) of the density map. The shape of the protein is also a factor, as proteins with rotational symmetry require a few user constraints to act as anchor points. As discussed later, the worst-case computational cost of our approach is exponentially proportional to the number of SSEs in the protein. Therefore, an increasing number of user constraints can to be used to reduce the computational cost when performance is a critical factor.

Although user constraints demand a time investment by a domain expert, we note that the time needed to specify these constraints is much smaller compared to the time needed if the user was to specify all the helix correspondences.

3.5.5 Performance

The result for all 10 proteins are presented in Table 3.2, showing the number of user-specified constraints used, the rank of the ground truth when considering only the helix correspondences, the rank of the ground truth considering all of the SSEs, the execution times, and the associated E_s scores for the list of top 35 candidates.

Observe from the tables that the graph matching approach in combination with the domain-specific strategies allow accurate identification of protein structure with no or a small amount of human input depending on the quality of the density volume. Also observe



(a) Volume and detected SSEs (b) Actual molecular model

Figure 3.10: The density volume and detected SSEs for the 1BVP protein (a), and the actual molecular model (b). Observe that in the SSEs detected using SSEHunter (a), the β-sheet highlighted in the molecular model has been incorrectly detected as being part of the larger sheet is located immediately above.

that for proteins where the ground truth correspondence did not appear in the list of candidates, the E_s scores are still high implying that the statistical information can be effectively used to guide the user towards the correct correspondence. The only exception to this observation is 1BVP where its E_s score is 0.58 even after constraining the helix elements. In this case, a set of β-strands that appear as a separate sheet in the sequence are incorrectly identified by SSEHunter as belonging to a large β-sheet (Figure 3.10). In this situation, our method performs poorly due to the mismatch between the expected strand length, and the actual strand length, leading to the low E_s score observed in Table 3.2.

Also note that the time taken to perform a computation is almost negligible in human terms ($< 40(s)$ for RDV P8), which facilitates a smooth user-interactive functionality[8].

3.6 Conclusion and discussion

In this chapter we reported a novel application of shape modeling and matching to roughly register the sequence of amino acid residues to an intermediate resolution density volume

[8] Experiments were performed on a PC with a 3GHz Pentium D CPU and 4GB of memory (our implementation runs on a single thread, thus utilizes only one of the cores of the CPU)

obtained using electron cryo-microscopy. We translated the biological problem into a computational one by representing the shapes of biological data (e.g., protein sequence and density volume) as attributed relational graphs. We solved the SSE correspondence problem using graph matching, and we demonstrated the effectiveness of the method on authentic as well as simulated data sets. One of our main contributions is an optimal algorithm for constrained error-correcting graph matching, which will be useful in other shape-matching tasks where the sought match has a linear shape.

Limitations: One of the limitations of our graph matching algorithm, like other best-first search based graph isomorphism techniques, is its high computational cost (both time and memory) for large graphs. In particular, our implementation of the method has difficulty in handling proteins with more than 25 SSEs without a fairly large number (> 6) of user-specified constraints. In the future, we plan to explore variants of the best-first search, including A*, iterative-deepening and memory-bounded A*, that are better suited for solving the graph matching problem on large data sets.

As demonstrated in Figure 3.10, our method is very sensitive to one type of β-sheet mispredictions in the density volume where multiple sheets are incorrectly identified as one. In the future we plan to explore better alternatives for SSE prediction from density volumes, and also incorporating techniques into our graph matching so that it can tolerate these mispredictions in a much more robust manner.

In our current method, user constraints are specified in the form of known SSE correspondences. There are also situations where different forms of user interaction may be more convenient to specify and more effective. For example, in proteins such as 1TIM (Figure 3.8) users can easily specify a pair of SSEs that occur one after the other by looking at the sequence. However, due to the symmetry they are not able to anchor them to a specific section in the sequence. For these cases, we intend to provide an interaction mode that allows users to constrain SSEs in the density to be neighboring elements.

Another beneficial interaction mode is to trace the most likely paths in 3D space of the strands making up each β-sheet. While the density does not contain adequate resolution for this purpose, we can approximate the paths of the strands by finding the shortest path on the geometric skeleton that connect the β-sheet end points. As incorrect configurations are very likely to contain paths that cross-over (which is not something seen in nature), this can be a very valuable visual clue when guiding the user towards an accurate correspondence.

We can take this one step further and try to incorporate this observation into our cost metric. However, to obtain reasonable results with such a cost metric the β-sheet detections would have be more accurate than they presently are.

The reason for lower success rates at lower resolutions often stem from the bad quality of the geometric skeleton obtained. These skeletons often contain many incorrect paths, and can lead to wrong estimates when using a shortest-path based value to approximate the loop length between SSEs in the density graph. In the future we wish to annotate the loop edges of the density graph with not only the length of the shortest path, but also a statistical measure of all the possible paths that bridge the same pair of SSEs. This can lead to a more accurate estimate of the loop lengths, and thus a more accurate cost measure.

Finally, our evaluation method assumes the existence of a manually labeled correspondence. This begs the question: in a practical application of this method, how do we identify the correct correspondence within the candidate list when no “ground-truth” is available? Indeed, this is a common problem in structural biology. Many structure prediction algorithms produce a gallery of structures that range in accuracy. The end user is often required to evaluate the model in the context of other data. The ranking achieved by our program is at least on par with the best algorithms if not significantly better. However, a fully automated approach for finding the most physically accurate correspondence based from a (much larger) list of candidates can be very beneficial. We are currently working on a method towards this goal and is discussed in more detail in Section 6.1.

A note on protein flexibility: Finally, we would like to note that while cryo-EM is well suited for imaging large macromolecular complexes in near-native solution conditions, the method ultimately reconstructs only a single snapshot of the assembly for a given set of images. In the event that there is some intrinsic flexibility in the molecule, the corresponding regions within the density map will appear less well resolved and have more ambiguous density values. Based on empirical evidence, most flexibility on the order of helix or sheet shifts are not easily identifiable until sufficiently high resolutions are reached (typically better than 7Å-8Å resolution). We envision that, given density maps of higher resolution our technique could produce potential secondary structure topologies through regions of disorder that may not have been readily detectable by visual observation.

Chapter 4

α-helix registration for flexible fitting

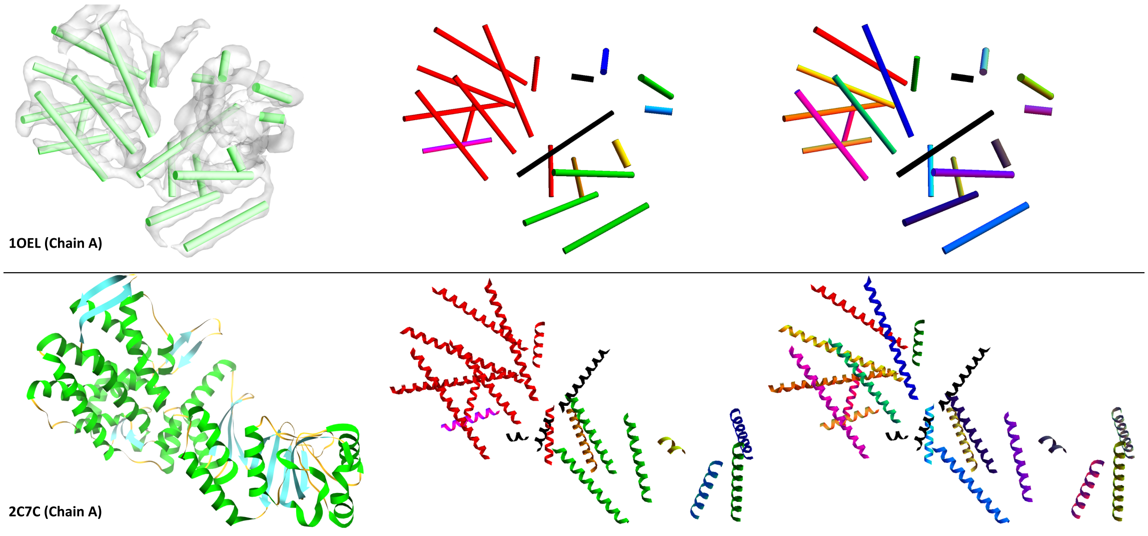
4.1 Introduction

The monumental developments towards solving atomic structures in the recent years, and the wide-spread use of the Protein Data Bank (PDB) [10] has resulted in a large number[9] of high-resolution structures being made publicly available. The significant majority of these structures have been solved using X-Ray crystallography and NMR spectroscopy, and are thus much smaller in scale than the typical macromolecular complex imaged using cryo-EM. Nevertheless, fitting these high-resolution structures to the low-resolution density map of the much larger complex may provide a reasonable approximation of its structure [78, 92], and is therefore a valuable tool for structural biologists.

Traditionally this problem of fitting a high-resolution model into the density map was based on the assumption that the protein undergoes only a rigid deformation (positional/rotational changes) [119]. While this was a reasonable assumption for maps obtained using X-Ray crystallography (as all molecules must be in a crystallisable conformation), it is no longer true for cryo-EM. This is because cryo-EM is capable of obtaining density maps at many different conformations, and these conformational changes are often non-rigid in nature with deformations similar to that of articulated bodies (i.e. hinge-like motions) [89, 74].

To overcome this limitation many flexible-fitting methods have been proposed in the recent past [102, 98, 106]. They start with an initial rigid-body alignment, and thereafter perform an energy minimization constrained by the density volume and the molecular *Van der Waals* forces. As these methods operate at the *micro-level* (i.e. atoms or amino acid residues) there are many degrees of freedom [61], and therefore are very computationally

[9] As of March 20th 2010, the PDB contains 64098 structures of which 55,361, 8,296 and 279 have been solved using X-ray crystallography, NMR spectroscopy and cryo-EM respectively



(a) Structural homologues (b) Isometric α-helix clusters (c) α-helix registration

Figure 4.1: Cryo-EM density volume, and α-helices of the 1OEL-chain A (a-above), and a high-resolution model of the 2C7C-chain A (a-below) which has a similar shape to that of 1OEL. Our method can be used to identify the isometric helix clusters (b) as well as the registration of each individual helix (c) to form a macro-level alignment of the two proteins.

intensive, taking days and often weeks to complete. More importantly, they are affected by local minima in the search space, and are accurate only when the initial rigid body alignment is reasonably close to the final conformation, with a very small degree of non-rigid deformation. Exacerbating the problem is that the failure case for rigid body alignment is also when the protein undergoes a large amount of non-rigid deformation, leading to bad initializations. For example, for the proteins shown in Figure 4.1a that undergo hinge-like unrolling deformations, a rigid-body alignment would result in approximately 50% of the map being misaligned. In these cases the *micro-level* flexible fitting methods can benefit greatly with an initialization that is accurate at the *macro-level* (i.e. secondary structure elements).

Our goal is to efficiently compute the *macro-level* deformation from a model to the map, accounting for both global (positional/rotational) and local (hinge motion) differences. To compute the deformation we consider only α-helices; for four reasons:

1. They are few, making the computation more efficient.
2. They tend to stay the same length among structural homologues and proteins at different conformations.

3. When undergoing hinge-like deformations they most often stay congruent[10] to a small cluster of α helices (most often in the same molecular domain).
4. They can be robustly detected from cryo-EM maps [9].

Our approach proceeds in two steps:

Step 1: Compute the deformation that takes helices in the high-resolution model to those in the cryo-EM map. Assuming hinge-motions dominate the shape change, we consider deformations consisting of a set of isometric transformations between clusters of helices.

Step 2: Propagate the helix deformations to the residues on sheets and loops, resulting in an initial alignment of the model with the map. The propagation should be smooth while respecting the density information.

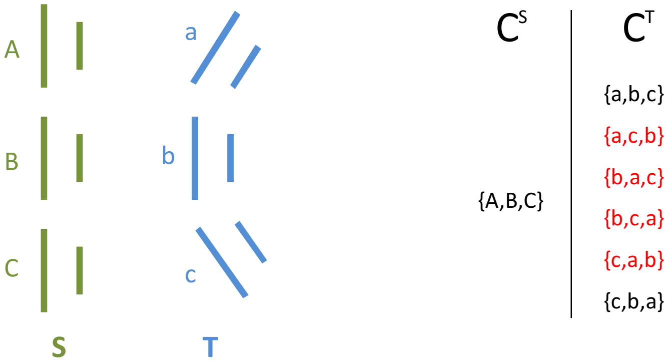
In this chapter, we address Step 1, leaving Step 2 to be addressed as future work. Step 1 is a geometric problem falling under the broad category of feature registration with a special assumption that the protein deforms in a quasi-isometric manner (or like an articulated body).

4.1.1 Problem statement

Given two sets of helices, one from the high-resolution model and the other from the cryo-EM map, 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*) transformation. Isometric transformations are more general than rigid-body ones, and additionally include reflections (by a plane or by a point). We consider this more general class of transformations due to the fact that reflections can be one of the many possible differences between structural homologues [80].

More specifically, consider the two sets of helices S, T that may have different cardinalities. The difference could be attributed to inaccuracy of helix prediction in either the high-resolution structure or the cryo-EM map, the occasional breaking and merging of helices in homologues or during conformational changes, or mapping a sub-unit S to a larger complex

[10] In geometry, two shapes are congruent if and only if one can be transformed onto the other by an isometry, where an isometry is a combination of translations, rotations and reflections.



C^S	C^T
{A,B,C}	{a,b,c}
	{a,c,b}
	{b,a,c}
	{b,c,a}
	{c,a,b}
	{c,b,a}

Figure 4.2: Two sets of helices 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 set.

T (or vice versa). Our goal is to identify a set of helix clusters $C^S = \{C_1^S, \ldots, C_k^S\}$ in S and another set $C^T = \{C_1^T, \ldots, C_k^T\}$ in T with the following properties:

1. **Quasi-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.
2. **Disjoint:** For any $i \neq j \in [1,k]$, $\quad C_i^S \cap C_j^S = \emptyset \quad$ and $\quad C_i^T \cap C_j^T = \emptyset$.
3. **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 helices. Also, no additional clusters can be added to the sets C^S or C^T while satisfying (1,2).

Note that such matching cluster sets C^S, C^T may not be unique. For example, if S or T contains *symmetric* parts, a group of helices in S may be isometrically mapped to multiple groups in T. This is illustrated in the example in Figure 4.2, 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).

With this scenario in mind, we wish to find among the multiple choices 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 spatially close for any $i \neq j \in [1,k]$, so should be C_i^T, C_j^T, and vice versa.

4.1.2 Method

To identify the matching clusters, our key observation is that a pair of quasi-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 helices with matching length, one in S and the other in T, and each edge represents a pair of helices 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 4.3.

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 4.4.

4.1.3 Contribution

The contribution of this work to the field on feature registration is two fold. First, while registration of point features has been extensively studied before, there has been few works on matching line features. We propose a graph-based formulation of isometric mapping between line features, and a fast algorithm for computing such mapping based on properties of this graph. Second, we propose a heuristic solution to the problem of semi-rigid mapping, particularly in the presence of symmetry, which has been rarely addressed in the past.

In addition, we make a practical contribution to the flexible fitting problem in structural biology by providing a fast and accurate way of aligning secondary structures (helices, in particular) using multiple rigid-body or reflective transformations.

4.2 Previous work

4.2.1 Protein docking

Multi-resolution modeling has been an active field of research in the recent past, leading to many different fitting techniques in two main classes: *rigid-body fitting* that considers the high resolution structure (template) as a rigid body, and attempts to find the best fit within the volume, and *flexible fitting* that allows each atom/amino acid residue of the template to move relative to each other, to better fit the density.

Rigid Body Fitting: Most of the early methods for rigid-body fitting [49, 70] were based on error minimizations (i.e. least squares fit) between observed and calculated structure factors in Fourier space with respect to the rotational and translational parameters of the template. These techniques are most useful when the diffraction amplitudes are the only source of information, or when you want to avoid the numerical complications when transforming into direct space [119]. However, deviations in the structure factors in the Fourier space do not correspond to localized position or orientation changes in direct space, thereby making it very difficult to refine the atomic positions of the template based on the fitting error [71]. Direct space fitting on the other hand provides much better control over localized changes in the structure [119], allows the development of visual tools for interactive docking [75, 5] and can easily incorporate atomic energy constraints [76]. These advantages have led to the development of many fitting techniques that exhaustively search (6 degrees of freedom) for the position and orientation maximizing the cross correlation between the low-resolution density volume and the template[11] [51, 86, 112, 57] in direct space, or its Laplacian [21]. However, as searching for the relative position of the template is far more time consuming in direct space than in Fourier space, these techniques are most often used only after an initial position has been estimated, or the appropriate molecule has been segmented out from the density volume.

The computation cost for the above methods lie in the range of a few hours, and has motivated the development of information-guided fitting techniques that perform rigid body fitting in a matter of minutes. *Mod-EM* [105] and *Fold Hunter* [51, 105] are two such

[11] In these techniques the high-resolution template is blurred to be the same resolution as the cryo-EM scan in order to simplify the cross-correlation function.

methods that use Monte-Carlo based sampling and progressively smaller step sizes, respectively, to achieve fitting times of a few minutes. However, both techniques selectively sub-sample the search space, and are therefore not guaranteed to find the optimal alignment between template and volume. *Situs* [121, 120] finds the fitting in a few seconds using vector quantization by training a topology preserving artificial neural network [69]. However, this method requires that all of the density volume be accounted for by the template(s), and therefore cannot be used with most larger macromolecular complexes. For further information on rigid body fitting methods, we direct the readers towards the excellent survey by Wriggers and Chacón [119].

Flexible Fitting: Most often the shape of a protein within a large assembly undergoes many hinge-like deformations when compared to its crystalized structure [89, 74] (available via the protein data bank). Therefore, the fitting process must allow deformations of the high-resolution model to better fit the volume. Methods such as *FlexProt* [94] and *FATCAT* [123] have been proposed that can efficiently detect the flexible alignment between two high-resolution molecules, but they depend on the relative ordering of secondary structure elements which is not readily available in the case of cryo-EM densities. Wang and Guibas [114] attempts to overcome this limitation by using a geometric approach to find the registration between surface features of the two molecules. However, for cryo-EM densities this method yields poor results as the noisy surface segmentations are often very different from the surface segmentations of the noise-free X-Ray crystallography based densities. To overcome these limitations, the recent past has seen the introduction of many flexible fitting methods such as *NMFF-EM* [103, 102], *NORMA* [98], *RSRef* [37], *S-flexfit* [111], *Flex-EM* [106] and *MDFF* [107]. The input for most of these techniques is a template that is rigidly aligned to the density volume, after which each method uses its own energy minimization routines or molecular dynamics simulations to locally move each individual atom (or amino acid residue) to obtain the best fit in the density while preserving their geometric and chemical properties. While these methods obtain much better alignment than rigid body fitting techniques, they are computationally intensive (Flex-EM takes 92 CPU hours for a 200 residue protein [106]) as they operate on the *micro-level* of the structure (i.e. atoms or amino acid residues), and therefore, have many degrees of freedom within the search space [61]. Furthermore, as they often use gradient-descent based energy minimization routines, they are affected by local minima in the search space, and are accurate only when the initial rigid body alignment is reasonably close to the final conformation, with a very small amount of non-rigid deformation. Therefore, an accurate *macro-level* alignment (i.e. secondary structure elements) can be used to significantly improve the accuracy, and efficiency of these flexible fitting methods.

As mentioned earlier, we can consider part of this *macro-level* alignment problem to be a specialized form of the feature registration problem and therefore, consider the many methods that have been proposed in the computer graphics and vision communities.

4.2.2 Feature registration

Computing the registration between rigid or isomorphic sets of features have 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* [67], *local spherical harmonics* [41], *salient surface features* [42], *curvature maps* [43], *spin images* [52], *geometric hashing* [117], etc. Thereafter, *Iterative closest point (ICP)* [12] or statistical analysis methods such as *RANSAC* [39, 83] are used to find the registration as the lowest-error alignment of the feature sets. These methods are very efficient [7, 48], but are only suited when there is a significant amount of rigid overlap between the two shapes.

To allow for articulated body based deformations, methods [50] have been proposed that use spectral embedding to represent each shape within the same coordinate system. However, these methods depend on a dense sampling of feature points, and cannot be used for the macro-level alignment problem as we have only a very small amount of α-helices (all of which can potentially move independently).

A more recent development utilizes voting mechanisms coupled with a greedy algorithm to quickly find the registration [114, 62]. However, the voting framework for these methods return ambiguous results in the presence of symmetries [125]. Au *et al.* on the other hand proposes a graph-based voting mechanism that is robust under symmetrical conditions, however, it is computationally expensive and can be used only with small (< 20) sets of features [8]. Zheng and Doermann [129] use *spin images* and local neighborhood constraints to find the lowest-error flexible deformation. However their local neighborhood criteria assumes only a small amount of flexing within each local neighborhood and therefore perform poorly in the presence of large transformations between helix clusters. The work of Zhang *et al.* [125] introduces a deformation-driven algorithm that attempts to find the registration between the features detected in the shape extremities by performing a combinatorial search. While they produce promising results, their error measure for each registration involves deforming one shape to align the corresponding features, and then measuring the resulting shape distortion. Computing this deformation is computationally intensive, especially on complex shapes.

Another body of work that is similar to ours involve finding a maximum subgraph isomorphism of the two shapes. This is an NP-hard problem, and heuristics have been proposed to avoid the exponential time complexity. Many such approaches [45, 99, 109, 13] assume an underlying shape topology, and therefore are not suitable for finding the registration between sets of features where the topology is unknown. This is the case for the protein fitting problem, as the connectivity between the secondary structure elements observed in the density volume cannot be accurately determined due to the lack of resolution. In contrast our method utilizes a clique-finding approach in a product graph to find the feature registration within a few seconds, and therefore, does not depend on the shape topology.

4.3 Graph formulation

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

4.3.1 Graph construction

We consider a *product graph* that represents the associative relation between the two sets of helices S and T. Vertices in this graph represent a pair of helices, one from S and one from T that have similar lengths. An edge between two vertices indicates a likely isometric mapping between two helices in S and two helices in T. To facilitate subsequent analysis, we consider each helix to be represented by two *oriented line segments* (OLSs) that have the same length as the helix 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:

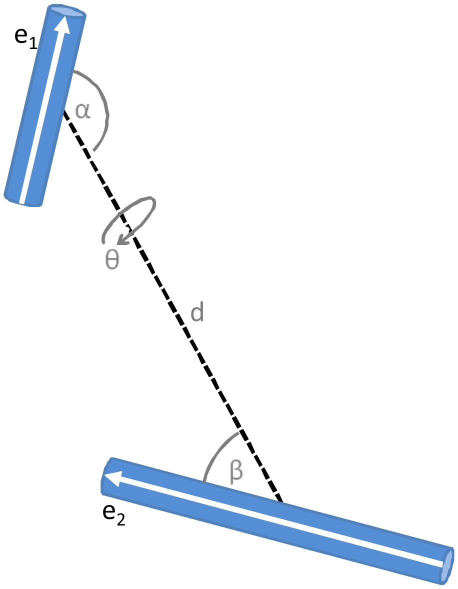


Figure 4.3: The $R(\vec{e}_1, \vec{e}_2)$ descriptor for the oriented line segments $\vec{e}_1$ and $\vec{e}_2$ consists of the distance between their centroids d, and the angles α, β and θ.

Vertex construction: For each pair of OLSs $\vec{e} \in S, \vec{f} \in T$, we can create a vertex in the graph if $\|L(\vec{e}) - L(\vec{f})\| \leq \varepsilon_l$, 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 4.3. 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 rational behind using this particular set of measures is that it is intuitive for users to specify error tolerances in fitting, based on their understanding of how much the helices could have moved or rotated relative to each other after undergoing conformational change.

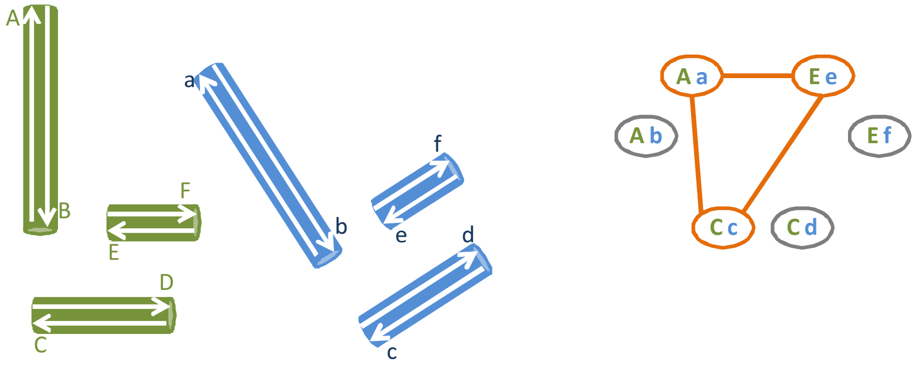


Figure 4.4: The zero-tolerance product graph G_0 (right) constructed for two sets of helices S, T (left), where each helix is represented by two oriented line segments (OLSs) indicated by arrows and labeled by letters. A clique (highlighted in orange) in the graph corresponds to an isometric mapping from a cluster of OLSs in S to another cluster in T.

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 helix in S (and T), and if $\|R(\vec{e}_1, \vec{e}_2) - R(\vec{f}_1, \vec{f}_2)\| \leq \{\varepsilon_d, \varepsilon_a, \varepsilon_a, \varepsilon_\theta\}$, where $\varepsilon_d, \varepsilon_a, \varepsilon_\theta$ are user-specified distance and angular tolerances.

The graph: We denote the graph constructed this way as G_ε, where $\varepsilon = \{\varepsilon_l, \varepsilon_d, \varepsilon_a, \varepsilon_\theta\}$ denotes the various error tolerances. In particular, we use G_0 as a shorthand to denote the graph constructed with zero tolerances $\varepsilon = \{0, 0, 0, 0\}$. In this graph, the vertices and edges represent helices and helix pairs between S and T that are mapped by exact isometries. An example of the graph G_0 is shown in Figure 4.4.

4.3.2 Isometric clusters as cliques

Based on the above construction, we can show that two helix 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), that identifies quasi-isometric clusters as cliques in the non-zero-tolerance product graph G_ε.

We will prove the above equivalence claim in each direction.

Proposition 1 *Let $\{\vec{e}_1,\ldots,\vec{e}_n\} \subseteq S$ and $\{\vec{f}_1,\ldots,\vec{f}_n\} \subseteq T$ 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 helix 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 $\{v_1,\ldots,v_n\} \subseteq G_0$ where each vertex v_i ($i \in [1,n]$) represents the pair $\{\vec{e}_i,\vec{f}_i\}$.*

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 $\{v_1,\ldots,v_n\}$ form a clique in G_0. □

Proposition 2 *Consider a clique $\{v_1,\ldots,v_n\} \subseteq G_0$, 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 $\{\vec{e}_1,\ldots,\vec{e}_n\} \subseteq S$ and $\{\vec{f}_1,\ldots,\vec{f}_n\} \subseteq T$ 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 helix 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 pairwise 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 is a well known fact that an isometric transformation exists between two point sets where all pair-wise distances are preserved. Hence the end points in the cluster $\{\vec{e}_1,\ldots,\vec{e}_n\}$ can be mapped to those in $\{\vec{f}_1,\ldots,\vec{f}_n\}$ 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. □

As an example, the highlighted triangle (a 3-clique) in the graph on the right of Figure 4.4 represents a isometric transform from the OLSs A,C,F in S to a,c,f in T. 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 4.2).

4.4 The algorithm

4.4.1 Overview

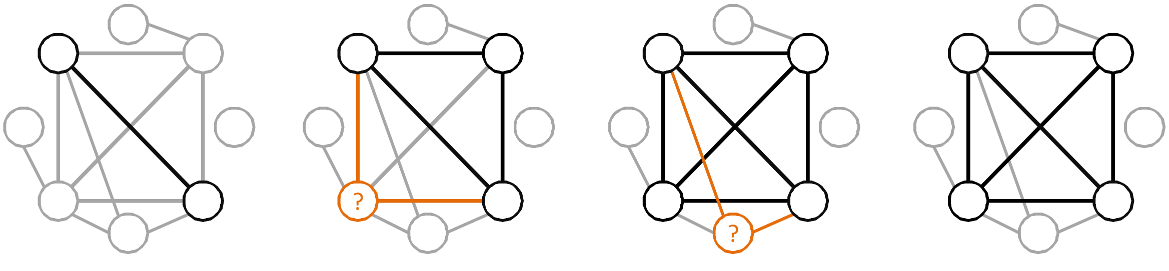
Recall from our problem statement (Section 4.1.1) that our goal is to identify a set of quasi-isometric pairs of helix clusters in S and T that are disjoint, maximal, and preserving the spatial coherence among neighboring clusters in S or T. Now that we formulated a quasi-isometric cluster pair as a clique in the product graph G_ε, the task becomes searching for a set of *maximal* cliques that represent disjoint, spatially coherent cluster pairs. Note that 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 propose a greedy heuristic that finds a locally optimal solution. As we will demonstrate in the next section, the heuristic gives reasonable matching results in all our test cases involving actual protein structures.

Our algorithm is a greedy, best-first tree search. A node in the search tree consists of a set of maximal cliques $Q = \{q_1, \ldots, q_k\} \subseteq G_\varepsilon$ that represent two disjoint, quasi-isometrically matched sets of helix clusters $C^S = \{C^S_1, \ldots, C^S_k\}, C^T = \{C^T_1, \ldots, C^T_k\}$ 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 helices 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 helices that have not appeared in Q. We then expand multiple children nodes, each 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 cost-matrix based procedure instead of the best-first tree search, guided by the same spatial-coherence principle.

In the next section, 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



(a) Highest valence edge (b) Vertex that can be added (c) Vertex that cannot be added (d) Final clique

Figure 4.5: Our triangle based search finds the largest clique in the graph (d) by first finding the highest valence edge (a), and for each vertex that form triangles with this edge we add it to the clique if it shares edges with the vertices already in the clique (b), and discard it if not (c).

structure of our graph. We will then discuss finding symmetric cliques, our cost function, and the cost-matrix based procedure for identifying smaller cliques.

4.4.2 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 [85]. 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 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 4.5 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 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 in the exact isometric matching scenario. 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. □

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 $\{u_1, \ldots, u_l\}$ 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 $\{v_1, v_2, u_1, \ldots, u_l\}$ is a clique.*

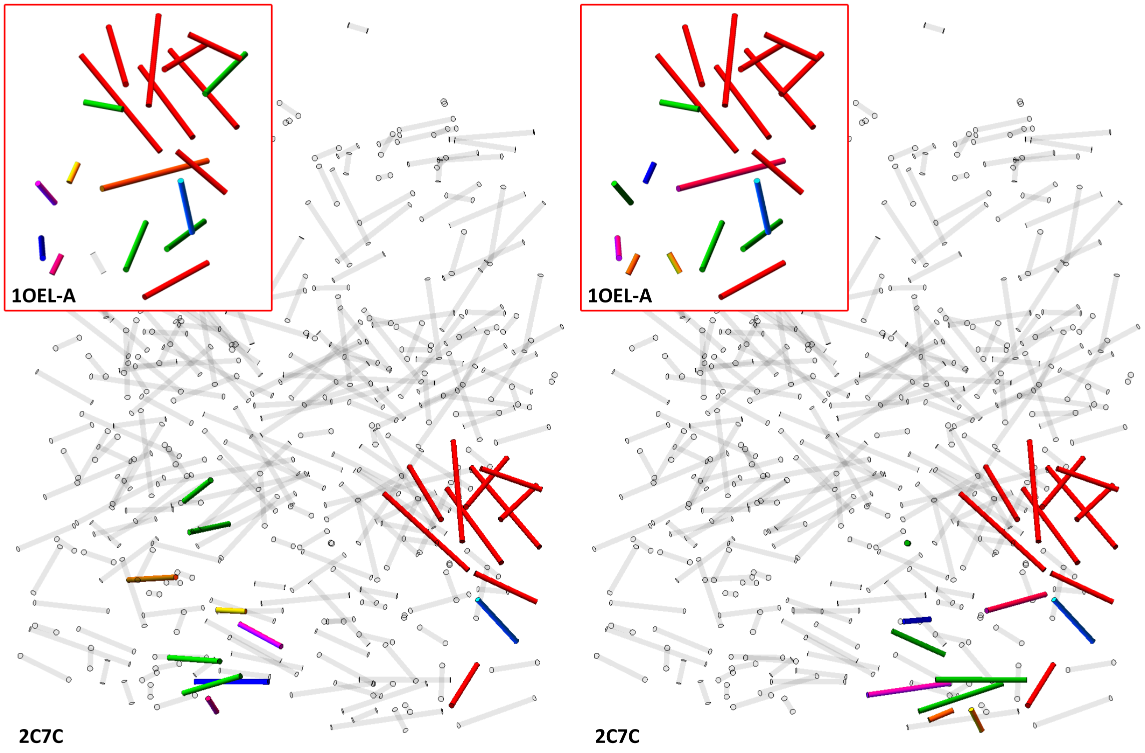
Proof: Since each triple $\{v_1, v_2, u_i\}$ ($i \in [1, l]$) forms a clique, by Proposition 2, there is an isometric transform, denoted as M_i, between $\{\vec{e}_1, \vec{e}_2, \vec{g}_i\}$ and $\{\vec{f}_1, \vec{f}_2, \vec{h}_i\}$. Following Lemma 1, all such M_i for $i \in [1, l]$ must be identical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

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.

4.4.3 Finding symmetric cliques

As mentioned before, 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_ε only finds one matching that involves the largest number of helices, and this matching may not be the optimal one in terms of maintaining spatial coherence (as seen in Figure 4.2). 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 (using the algorithm described above) in some residue graph G that represents a quasi-isometric mapping from a cluster $E = \{\vec{e}_1, \ldots, \vec{e}_l\} \subseteq S$ to another cluster $F = \{\vec{f}_1, \ldots, \vec{f}_l\} \subseteq T$, 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



(a) Registration without spatial coherency

(b) Registration with spatial coherency

Figure 4.6: The result of our method in the presence of symmetry without using the spatial coherency cost (a), and with using spatial coherency (b).

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 represent 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.

4.4.4 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 pairwise cost function between two cliques $p, q \subseteq G_\varepsilon$:

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

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.

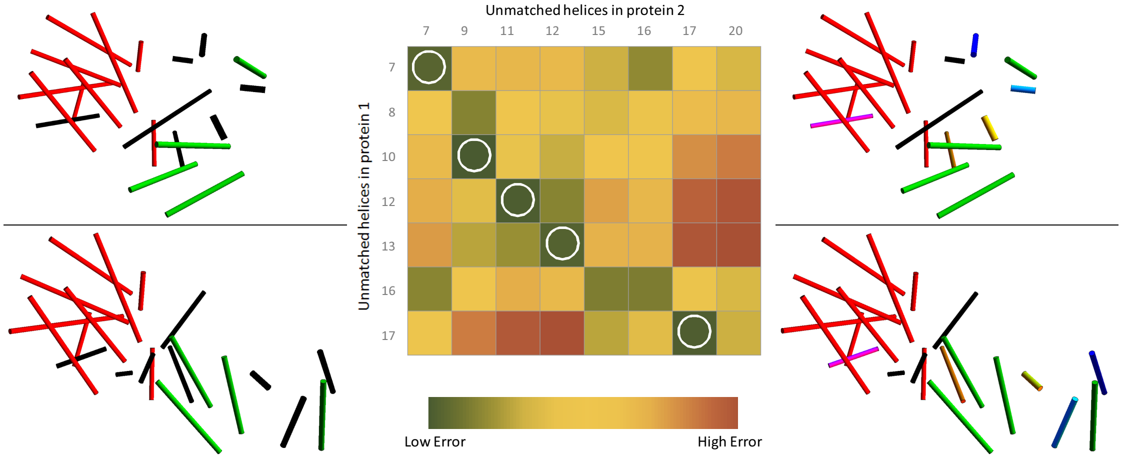
To make our tree search better approximate the optimal solution, we would like the cost function associated with the tree nodes to be non-decreasing as the nodes are expanded. To achieve this effect, we consider the cliques at each tree node as an ordered set $Q = \{q_1, \ldots, q_k\}$ where newly added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

$$H(Q) = \sum_{i=2}^{k} w(q_i) \min_{j<i} h(q_i, q_j) \tag{4.1}$$

where the weight function $w(q) = 1/(1 - e^{-Size(q)/\lambda})$ down-grades the penalty for larger cliques. This is because larger cliques represent isometric mapping between greater number of helices, 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. Figure 4.6 shows a comparison of our method being used with and without the spatial coherency cost function.

4.4.5 Finding single helix 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, which represent matching between individual helices in S and T that do not belong to any larger isometric clusters, we


Unmatched helices in protein 2
7
9
11
12
15
16
17
20
Unmatched helices in protein 1
7
8
10
12
13
16
17
Low Error
High Error

(a) Clusters after best-first search

(b) Cost-matrix

(c) Clusters after single-helix registration

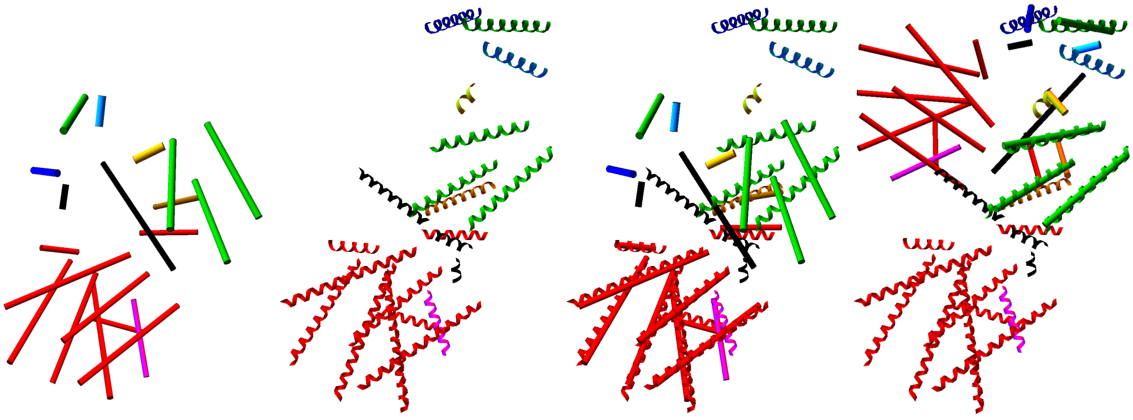
Figure 4.7: Isometric clusters identified after the best-first search (a) where black indicates un-assigned helices, the cost matrix for the single-helix registration (selected helix pairs have been highlighted with white circles) (b), and the final set of isometric clusters after the single-helix matching step.

resort to a greedy cost-matrix based scheme 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 helices in S and T after the best-first tree search. Each entry in the matrix records the spatial non-coherence of matching a helix x in S and a helix y in T with respect to the already matched clusters. More specifically, denote the cliques found by tree search as Q, and the four graph vertices representing the pairwise matching between the two OLSs for x and the two OLSs for y as $v_1, \dots, v_4$, the entry of the matrix is set to be

$$\min_{i \in [1,4]} H(Q \cup \{v_i\})$$

where H is the cost function in Equation 4.1. If none of these vertices exist in the graph, the cost is set to be ∞. Figure 4.7b shows such a cost matrix. Given this matrix, we match up the helices that give the lowest cost in the matrix, eliminate the corresponding row and column from the matrix, and repeat this process until all helices in S or T are assigned, or a maximum cost threshold has been exceeded.



(a) α-helices of 1OEL-A (b) α-helices of 2C7C-A (c) Aligned by red cluster (d) Aligned by green cluster

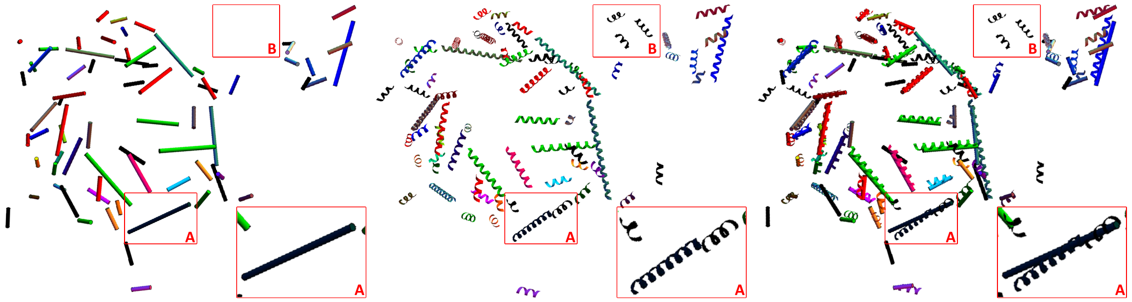
Figure 4.8: α helices detected in the 1OEL-chain A and 2C7C-chain A proteins shown as cylinders (a) and coils (b) are colored by the isometric clusters identified using our method. We can align the two shapes by the red (c) or green cluster (d) to visually inspect their isometric alignment. (helices that have no registrations are displayed in black)

4.5 Results

We are interested in the problem of finding the quasi-isometric registration between the sets of α-helices obtained from a high-resolution model and those identified in the density volume. However, to validate our results we must compare against an established ground truth and therefore, we demonstrate our method on a set of protein data obtained from the Protein Data Bank [10]. For each experiment, we pick two high-resolution models that are structurally similar and have similar sequences. Thereafter, we extract OLSs from their α-helix annotations, and use the sequence alignment to establish a ground truth registration. It is this registration that we use to validate the accuracy of our results.

Figure 4.8 shows the results of our method on two proteins. As you can see from 4.8a and 4.8b the protein has two main isometric clusters, where the top cluster in 2C7C (green) unrolls out from the lower cluster (red). Our method accurately identifies these isometric clusters, and furthermore determines the correct registrations for the helices that move independent of both main clusters. Figure 4.1c shows the individual helix registrations obtained from our method[12].

[12] In this chapter (apart from Figures 4.1c and 4.10-bottom) we have a common coloring theme to demonstrate our results. Each helix (either represented as a cylinder or coil) is colored based on the isometric cluster that it is assigned to. This allows quick visual identification of the corresponding isometric clusters of each protein. The only deviations to this norm are in Figure 4.1c where each helix is colored based on its



(a) α-helices of 3E8K

(b) α-helices of 2GP1

(c) 3E8K and 2GP1 rigidly overlayed

Figure 4.9: α-helices detected in the 3E8K (a) and 2GP1 (b) proteins of the HK97 Bacteriophage colored by the identified isometric clusters. The two proteins are overlayed (c) to show their differences. Observe that our method has successfully tolerated α-helix identification errors where one helix has broken into multiple helices (region A), and where helices in one protein are not present in the other (region B).

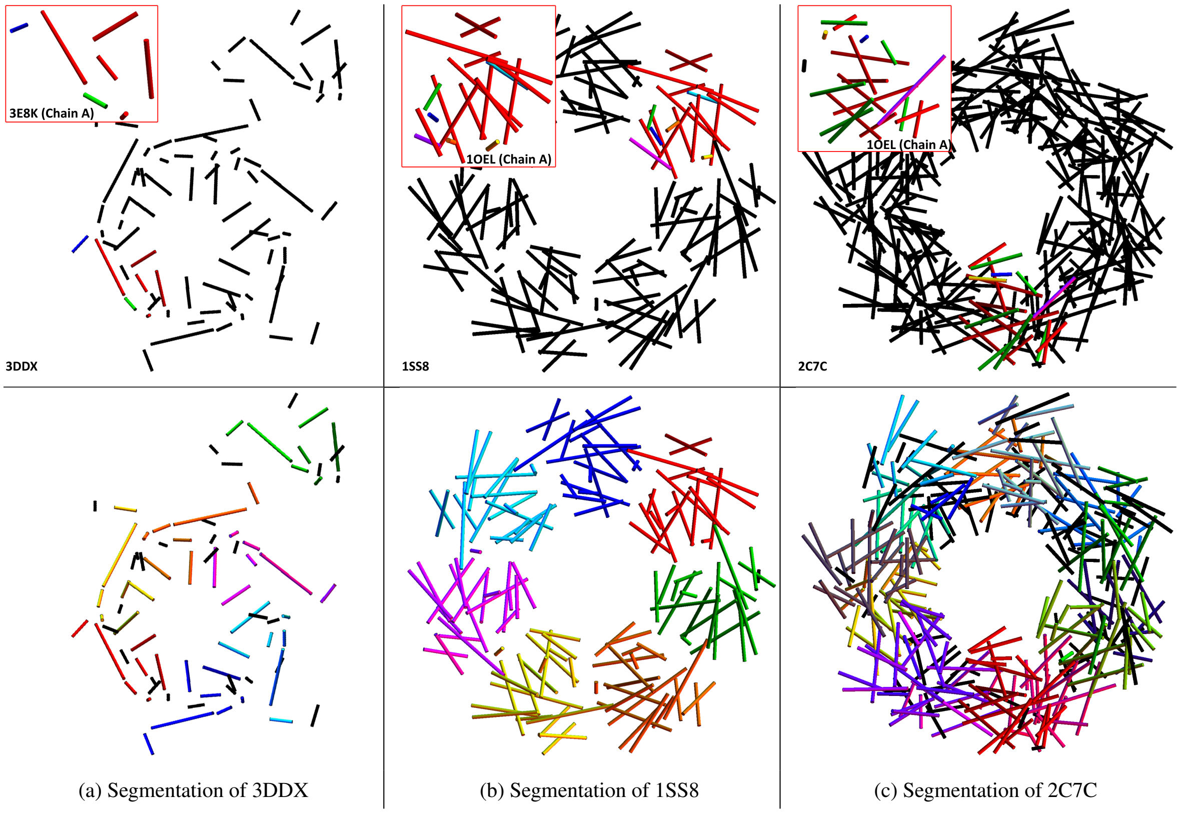
	Experiment	Ground truth	Correctly found		Not found		Incorrectly
		reg. count	reg.	%	reg.	%	found reg.
1	3E8K-A 2GP1-A	6	6	100.00%			
2	3E8K-A 3DDX-A	7	7	100.00%			
3	1OEL-A 1SS8-A	19	19	100.00%			
4	1OEL-A 2C7C-A	20	18	90.00%	2	10.00%	
5	3E8K-A 2GP1	46	45	97.83%	1	2.17%	2
6	3E8K-A 3DDX	49	49	100.00%			
7	3E8K 2GP1	57	55	96.49%	2	3.51%	2
8	1OEL-A 1SS8	133	133	100.00%			7
9	1OEL-A 2C7C	331	242	73.11%	89	26.89%	7

Table 4.1: The results of our method compared against the ground truth.

α-helix identification errors can pose a significant challenge when determining their registration. For example a helix identified in one shape can be detected as multiple helices in the other shape, or not be seen at all. As shown in Figure 4.9 our method is capable of robustly tolerating these errors, and does not register those helices.

In Figure 4.10 we can see how finding the α-helix registrations can be used segment proteins based on their symmetrical elements (chains). To achieve this, we first find the registration of the single-chain protein with the multi-chain molecule (Figure 4.10(above)). We can then remove the associated vertices from the residual graph G_ε, and iterate the process until all symmetrical elements have been found as seen in Figure 4.10(below). Recall from Figure 4.1 that the 1OEL-A and 2C7C-A registration finds multiple isometric clusters in each protein, and observe that our method has correctly found the proper grouping of these

unique registration, and Figure 4.10-bottom where the symmetrical chain elements are colored separately to demonstrate their segmentation.



(a) Segmentation of 3DDX

(b) Segmentation of 1SS8

(c) Segmentation of 2C7C

Figure 4.10: Our method used to segment helices detected in proteins 3DDX (a), 1SS8 (b) and 2C7C (c) by their symmetrical elements (chains). The one-to-one registration after our method is run once (above), and the segmentation after multiple iterations (below).

clusters even in the presence of large amounts of symmetry (Figure 4.10c). This is due to the spatial non-coherence cost function defined in Equation 4.1.

In Table 4.1, we compare the results of our method against the ground truth. For a wide variety of shapes, we are able to find the helix registrations with a very high accuracy (for all experiments $>95\%$ of registrations found are correct). However, our method fails to identify all possible registrations in cases where the shapes undergo a large amount of flexible deformation (i.e. the isometric clusters are very small).

Performance: As seen in Table 4.2 our algorithm is able to identify the α-helix registration for large molecules in under four minutes[13], which is orders of magnitude faster than flexible fitting methods. This allows our method to be used as a much more accurate

[13] All experiments were performed on a PC with 3GHz Pentium D CPU and 4GB of memory (our implementation runs on a single thread, thus utilizes only one core)

	Experiment	Time (seconds)			
		Graph const.	Best-first	Cost matrix	Total
1	3E8K-A 2GP1-A	0.01	0.00	0.00	0.01
2	3E8K-A 3DDX-A	0.01	0.00	0.00	0.01
3	1OEL-A 1SS8-A	0.04	0.00	0.00	0.04
4	1OEL-A 2C7C-A	0.09	0.01	0.01	0.11
5	3E8K-A 2GP1	0.93	0.15	0.01	1.09
6	3E8K-A 3DDX	1.28	0.17	0.03	1.48
7	3E8K 2GP1	32.07	5.43	0.10	37.6
8	1OEL-A 1SS8	9.90	1.48	0.06	11.44
9	1OEL-A 2C7C	151.04	51.34	0.71	203.09

Table 4.2: The times taken by our method to find the α-helix registrations.

initialization for flexible fitting, at a practically negligible time cost. The time complexity for the initial graph construction is $O(n^4)$, where n is the number of helices in a protein. In the future we anticipate using a hashing mechanism to reduce this time complexity to $O(n^2)$. Finding the correct set of isometric clusters is an NP-Hard problem, and therefore, has a worst-case exponential time complexity. However, in our experiments we have seen that the cost function used to guide the search performs reasonably well and terminates within a few seconds for large data sets. The cost-matrix based single-helix matching step has an $O(m^2r)$ time complexity where r is the number of isometric components found in the best-first search, and m is the number of unassigned helices.

4.6 Conclusion and discussion

In this chapter, we proposed an automatic method to find the registration between α-helices of a high-resolution molecular model with the α-helices observed in a density volume. Additionally we can also find the isometric clusters of α-helices where each cluster defines a unique isometric transformation in 3D space that aligns the two registered sets of helices. In the future we envision utilizing this registration to deform the high-resolution model to approximately fit the density volume, and thereafter use that alignment as an initialization for *micro-level* flexible fitting routines. Our method is efficient, accurate, and is robust to errors in α-helix detection. We tested our technique on a suite of protein data and demonstrated its accuracy as well as its performance on small to large sized molecules.

Limitations: Our method is based on the assumption that proteins undergo hinge-like deformations and therefore, can be segmented into multiple isometric clusters. However, for some shapes (Figure 4.10c) there can be a significant subset of helices that deform independent of all other helices, and therefore violates our assumption. Our method regards these as errors in helix detection and annotation, and fails to find the registration for these helices. In the future, we would like to explore relaxing the rigidity constraints to allow a significant amount of flexing in the helices as long as they maintain the same relations to their neighbors.

Currently we only find the registration between the α-helices in the high-resolution model and the density. This can lead to potential errors when deforming proteins that contain β-sheets. In the future we would like to extend our method to also support different feature types (surfaces in this case), leading to much more accurate registrations. Section 6.1 in Chapter 6 provides more details in this direction.

Chapter 5

Gorgon: An interactive molecular modeling system

5.1 Introduction

In the previous chapters, we described computational techniques that can be used to tackle three stages of the protein modeling pipeline. These techniques together with the many other methods in literature provide a plethora of options for the structural biologist. However, they most often appear as individual isolated systems, and many hours need to be spent learning, tweaking and managing data between them. Furthermore, most systems are based on command-line arguments and scripting languages, and therefore visualizing their results often require the use of another system. In recent years, many molecular visualization and analysis tools have been developed [81, 35]. However, they focus on molecules imaged using X-Ray crystallography, and do not apply well to intermediate resolution cryo-EM density volumes.

5.1.1 Problem statement

We are interested in the problem of developing a modeling system specifically geared towards building molecular models from intermediate resolution density volumes, most often obtained using electron cryo-microscopy or low-resolution X-Ray crystallography. We want the system to be visual, interactive, support a wide variety of data formats, and be extensible such that new computational techniques can be easily incorporated and can work in unison with the other methods available in the system.

5.1.2 Contributions

In order to address the above-mentioned challenges we have developed “Gorgon”, an interactive molecular modeling system that is available for the Windows, MacOS and Linux platforms as independent 32 and 64 bit binaries. Gorgon can be freely downloaded for academic and non-profit use from http://gorgon.wustl.edu and currently[14] has 191 registered users from 27 different countries. We make the following key contributions to the field of structural biology:

- Visual, easy-to-use interface for all stages of the *de novo* molecular modeling pipeline, with a focus on intermediate resolution density volumes.
- Incorporation of many tools that can be used for molecular modeling [9, 3, 54, 2, 4]
- Extensible plug-in architecture allowing users to interface with existing techniques, or add their own methods and scripts.
- Support for a wide variety of input and output data formats.

5.2 Previous work

A simple search on the internet for “volumetric data visualization” yields hundreds of commercial and academic tools such as *Slicer-Dicer* [82], *VolView* [56] and *VolVis* [97], that are freely available, and are widely used in many different disciplines. However, most of them can only be used to visualize density volumes, and not for modeling molecular structures. On the other extreme are tools such as *Modeller* [36], *Rosetta* [31] and *EMAN* [68] that allow users to create molecular models using only command-line arguments or scripting languages. While most of these tools now have visualization routines, they are often only for display purposes, and cannot be used to interactively build molecular models.

The most commonly used software packages that allow users to visually and interactively model molecular structures include *UCSF Chimera* [81] and *Coot* [35]. While these packages provide a large amount of functionality for molecular modeling, they focus on near-atomic resolution (1-2.5Å) density volumes obtained using X-Ray Crystallography or NMR Spectroscopy, and are not well suited for intermediate resolution cryo-EM density

[14] The registered user count is current as of the 14th of March 2010.

volumes. In contrast, Gorgon provides a visual, interactive and intuitive molecular modeling system that focuses on modeling structures that are imaged using cryo-EM, and features a set of novel automatic algorithms that are uniquely designed for these intermediate resolution density volumes (3.5-10Å).

5.3 Main features

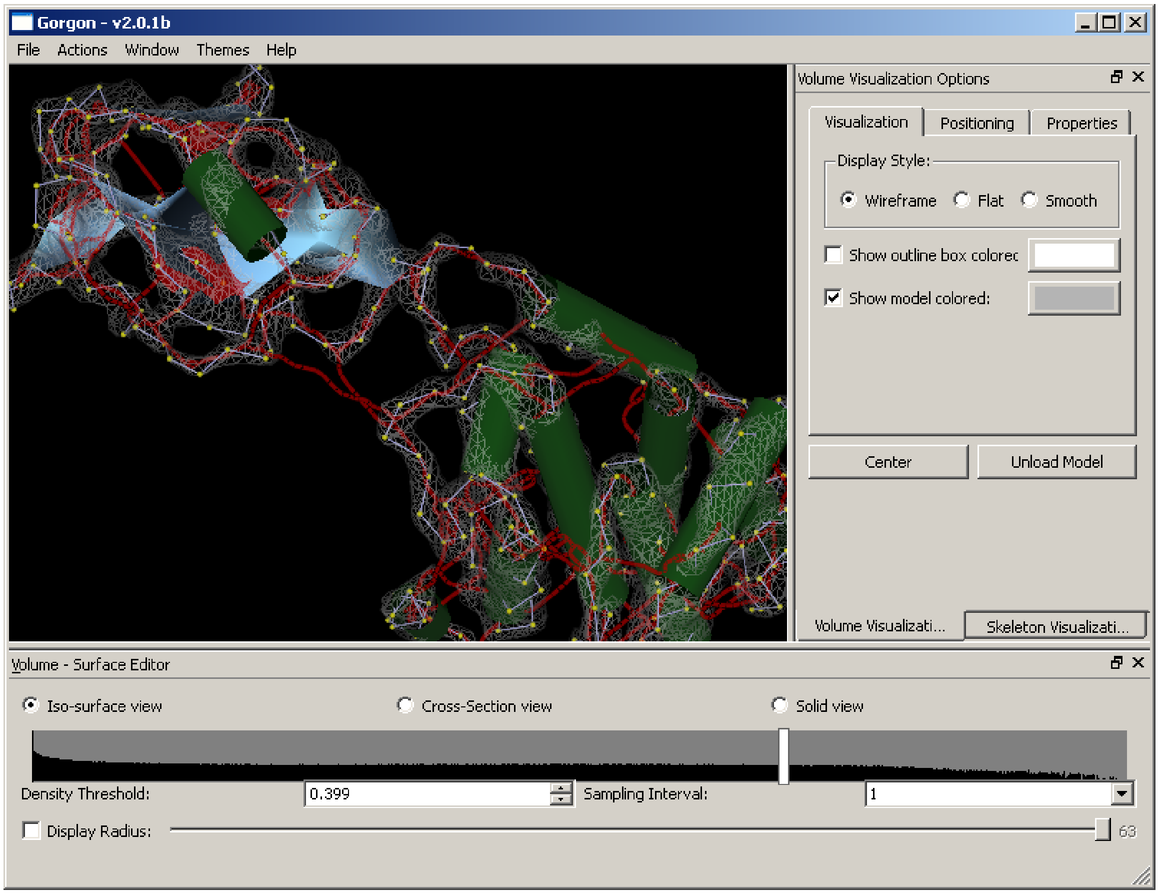
5.3.1 Visualizing molecular data

We provide an integrated visualization platform for volumetric data, geometric skeletons, secondary structure elements and C_α backbones. As seen in Figure 5.1, these different stages of the modeling pipeline can be visualized together, thereby allowing interactive visual refinement of the model being generated.

Volumetric data can be visualized as Iso-surfaces (using the Marching Cubes algorithm by Lorenson and Cline [65]), wire-frame meshes, volume renderings or cross sections. This allows users to load and visually analyze a wide variety of data sets not limited to cryo-EM density maps of proteins, and get a better understanding of their underlying 3D structure. As seen in Figure 5.1 geometric skeletons are visualized as non-manifold meshes that contain edge and face elements, while α-helices and β-sheets are shown as cylinders and surfaces respectively. We provide a ball and stick visualization method of C_α backbones where the bond is colored based on its deviation from the statistical median distance of 3.8Å (i.e. bond is colored blue if it is too short, and red if too long). We can also simulate side-chains allowing users to easily compare their backbone against the density volume for a rough visual alignment.

5.3.2 Computing geometric skeletons

As described earlier, geometric skeletons are powerful shape descriptors that can be used to simplify and understand the underlying structure of density volumes. In Gorgon, we provide three geometric skeletonization routines that can be used based on the data-set being analyzed.



Display Style: ● Wireframe ○ Flat ○ Smooth

□ Show outline box colorec

☑ Show model colored:

● Iso-surface view ○ Cross-Section view ○ Solid view

Density Threshold: 0.399 Sampling Interval: 1

□ Display Radius: 63

Figure 5.1: The 1BVP protein of the Blue Tongue Virus displayed in Gorgon, where the volume is shown as a transparent wire-frame, the skeleton as a red mesh, α-helices as green cylinders, β-sheets as blue surfaces, and the C_α backbone as a ball (yellow) an stick (light-blue) model.

- **Binary skeletonization:** Here we implement the method of Ju *et al.* [54] that can efficiently create geometric skeletons given a threshold to segment the density. Based on this segmentation, a skeleton will be generated that accurately captures its shape and topology. This method is quick and efficient, but is dependent on the threshold used, and should be used only when you can clearly segment the volume using that single threshold.

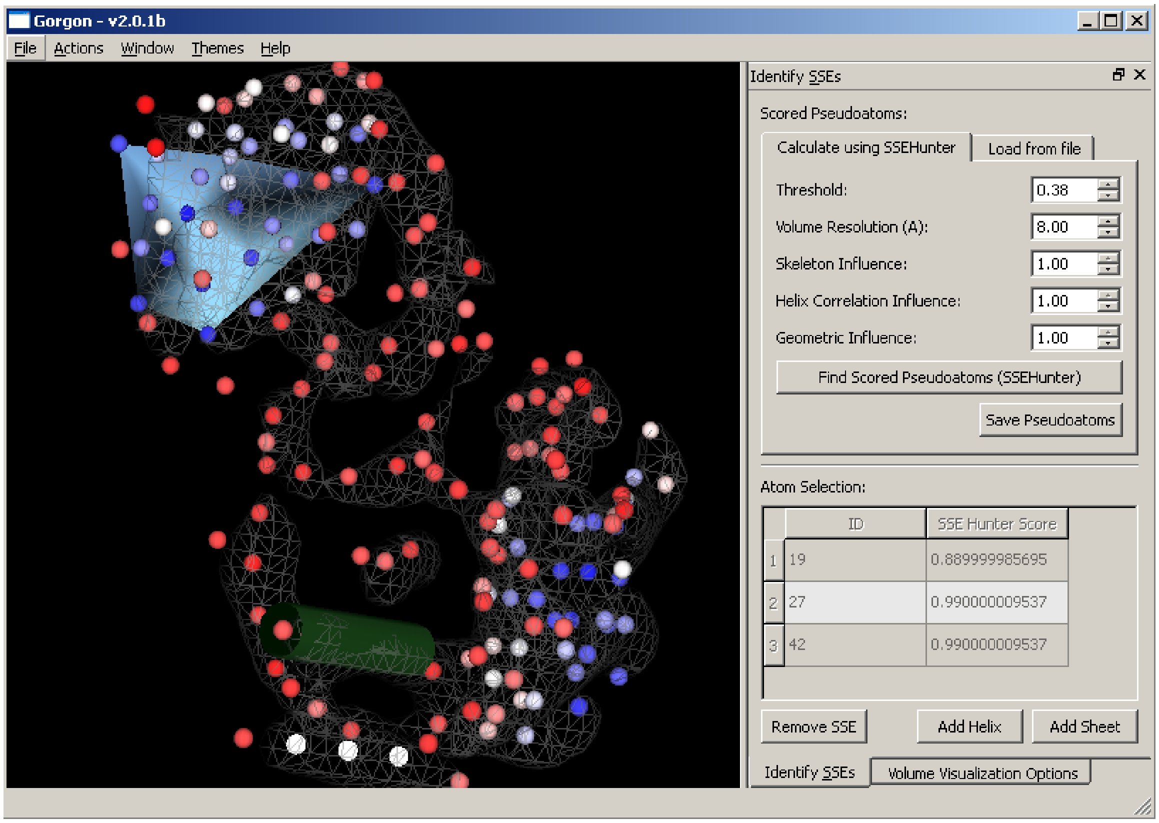
- **Grayscale skeletonization:** Here we implement the work described in Chapter 2 [4]. This method does not require a specific threshold, and can effectively find the shape and topology of a density volume even when the features are seen at different thresholds. In the presence of a large amount of noise (i.e. volumes obtained using photo-acoustic imaging), this method can generate incorrect loop segments, and broken connectivity. For these data sets, we suggest using the interactive skeletonization method described next.

- **Interactive skeletonization:** This method implements our work [2] that allows a user to quickly create a geometric skeleton by using the interactive user interface provided by Gorgon. With a few mouse clicks and sketches, a user can quickly create a visually accurate skeleton even in the most noisy data sets.

The three skeletonization methods described above should be used based on the noise-level and resolution of the density volume being analyzed. For this purpose, we suggest the following approach for generating skeletons of cryo-EM density volumes. First, obtain a skeleton by performing binary skeletonization using a relatively high threshold that allows you to visually observe the individual secondary structure elements without them grouping together. This would generate a high-quality skeleton for the SSEs, but will not contain all of the loop information. To obtain this loop information, we suggest performing grayscale skeletonization while maintaining the skeleton created in the earlier step. In our experiments, we have found that this approach leads to the best quality skeletons that can then be used for finding the correspondence between the observed SSEs and the ones predicted from the sequence.

5.3.3 Annotating secondary structure elements

For the annotation of secondary structure elements, we implement SSEHunter [9], a widely used tool for this purpose that was first made available in command-line form in the EMAN [68] application suite. This method utilizes a geometric skeleton, cross-correlation routines, as well as geometric shape measures to generate a set of annotated pseudo-atoms as seen in Figure 5.2. Gorgon colors these pseudo-atoms based on their likelihood of being part of an α-helix (red) or a β-sheet (blue), which in-turn makes it easier for a user to interactively select them and create the SSEs that they represent. Furthermore, we allow the user to fit the created SSEs to the underlying density volume. This allows the creation of much more accurate SSE annotations that can increase accuracy when computing the correspondence between these SSEs and the ones predicted from the sequence.



Threshold: 0.38

Volume Resolution (A): 8.00

Skeleton Influence: 1.00

Helix Correlation Influence: 1.00

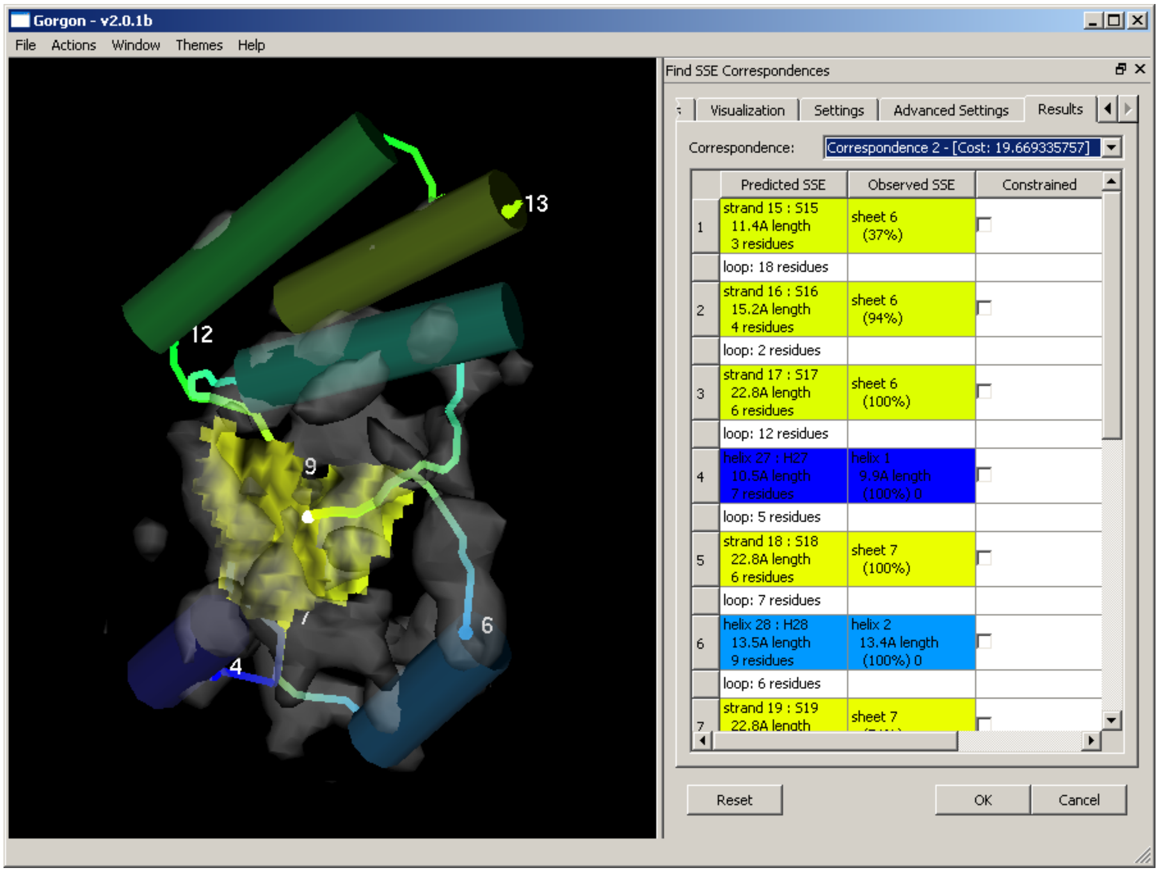
Geometric Influence: 1.00

	ID	SSE Hunter Score
1	19	0.889999985695
2	27	0.990000009537
3	42	0.990000009537

Figure 5.2: The pseudo-atoms scored based on their SSE likelihood are shown as red (possible α-helix), blue (possible β-sheet) and white (uncertain) spheres. In this figure the user has interactively created an α-helix and a β-sheet.

5.3.4 Finding the correspondence between SSEs observed from the density and predicted from the sequence

In 2008, we first released Gorgon to the public with an implementation of a search that allowed a user to find the correspondence between the observed and predicted α-helices of a protein [3]. Since then we have incorporated the α-helix and β-sheet correspondence method described in Chapter 3. This method allows a user to quickly find a set of the most likely correspondences, visually compare them, constrain correspondences that they are certain of, and iterate the process if needed. An additional benefit of incorporating the β-sheets in this correspondence search is that it allows us to trace out the skeletal paths that connect the corresponding elements. With this, we can obtain a pseudo-backbone that is used to visually guide the user when building the actual C_α backbone trace. Figure 5.3 shows a correspondence generated for the GroEL apical domain.



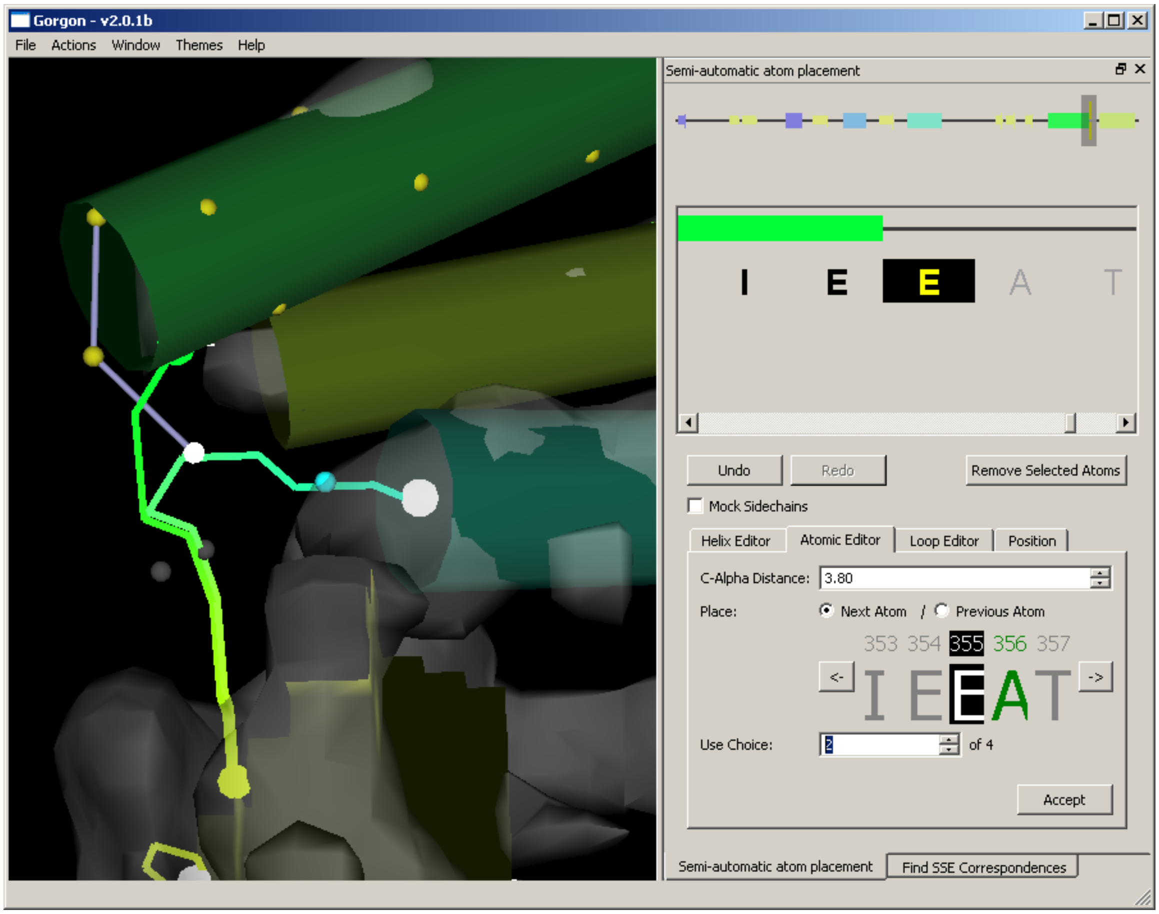
Correspondence: Correspondence 2 - [Cost: 19.669335757]

	Predicted SSE	Observed SSE	Constrained
1	strand 15 : S15 11.4A length 3 residues	sheet 6 (37%)	□
	loop: 18 residues		
2	strand 16 : S16 15.2A length 4 residues	sheet 6 (94%)	□
	loop: 2 residues		
3	strand 17 : S17 22.8A length 6 residues	sheet 6 (100%)	□
	loop: 12 residues		
4	helix 27 : H27 10.5A length 7 residues	helix 1 9.9A length (100%) 0	□
	loop: 5 residues		
5	strand 18 : S18 22.8A length 6 residues	sheet 7 (100%)	□
	loop: 7 residues		
6	helix 28 : H28 13.5A length 9 residues	helix 2 13.4A length (100%) 0	□
	loop: 6 residues		
7	strand 19 : S19 22.8A length	sheet 7	□

Figure 5.3: Here we show the correspondences found between the observed and predicted SSEs for the GroEL apical domain. The volume is shown as a transparent iso-surface, and the SSEs and pseudo-backbone paths are colored based on their corresponding location in the sequence.

5.3.5 Semi-automatic methods for building a C_α backbone

Building an accurate C_α backbone requires a significant time investment and understanding of density features. Size, complexity and quality of the density map all affect the model building process. Even the most experienced users may not be able to build a reliable model in poorly resolved regions of density maps. Therefore, the time and ease of building a *de novo* model is most often proportional to the map quality, complexity of the structure and experience of the user. Gorgon attempts to integrate and streamline the model building process while at the same time enhancing model accuracy using the following complementary methods that utilize the density-sequence correspondence as well as other domain-specific geometric measures:



□ Mock Sidechains

C-Alpha Distance: 3.80

Place: ● Next Atom / ○ Previous Atom

Use Choice: 2 of 4

Figure 5.4: Gorgon is being utilized to build a C_α backbone guided by the correspondence. C_α atoms have been placed for one of the α-helices, and a loop is being built using the atomic editor.

- **Helix editor - C_α placement of α-helices:** Given the correspondence between the observed helices in the volume and the predicted helices in the sequence, we know which region in the density those amino acid residues need to be placed. We use that spatial information together with the known bond-angles to let the user place C_α with a press of a button.

- **Loop editor - Interactive C_α placement of loops and β-strands:** For this, we utilize a technique similar to that of our interactive skeletonization routine [2]. The user first selects a range of amino acid residues from the sequence, and then interactively clicks the start point in the 3D volume visualization area and moves the mouse towards the end point. Based on this input, Gorgon suggests a C_α backbone that aligns well with the underlying density volume. If the suggestion is wrong, the user can quickly correct it using a mouse-drawn sketch. This method can be used to quickly place large loop segments, and also allows the user to visually measure its bond-length accuracy based on the coloring scheme (i.e. bonds that are too long get colored red, and too short get colored blue).

- **Atomic editor - Manual, iterative C_α placement:** If the user wishes to incrementally build the C_α backbone, we allow them to do so using this manual method. The user must first place C_α atom at a known location (most often obtained using the helix editor), and then Gorgon suggests possible C_α atom locations based on the distance to the selected atom, and the geometric skeleton in that local neighborhood. The user can approve the most likely location and proceed to place the next residue.
- **Placement - Manual refinement of 3D locations:** We also provide a set of widgets to give the user a precise method of changing the location (and orientation) of the C_α atoms that have already been placed. This is most useful when you want to slightly move the C_α atoms to better fit the density volume.

Figure 5.4 shows the manual, iterative C_α placement in practice. Anecdotally, the construction of GroEL model by hand required approximately three months to construct. With Gorgon, an experienced user can build a model of similar quality for the apical domain ($\sim 40\%$ of the entire protein) within a few hours.

5.3.6 I/O support and extensibility

Input Output Data Formats: We currently support a wide array of input/output data formats. These formats are listed in Table 5.1.

Plugin framework: We have developed an open, extensible and easy to use framework that allows users and third party developers to write their own plug-ins for Gorgon. A user-written plug-in has access to all the internal Gorgon methods, libraries and visualization techniques. Currently we bundle only a single plug-in with Gorgon that acts as both a documented template for third-party developers as well as a version checker looking for newer Gorgon releases. In the future, we envision the development of plug-ins that allow Gorgon to interface with popular molecular research tools and libraries such as EMan[68], Modeller[36] and Rosetta[31], allowing the structural biology community to

Extension	Description	Input	Output
Density Volumes			
MRC	Cryo-EM density volumes	✓	✓
CCP4	X-Ray crystallography density volumes	✓	✓
MAP	X-Ray crystallography density volumes	✓	✓
RAW	Raw 3D float arrays	✓	✓
PTS	3D point clouds	✓	
NB	3D Mathematica list		✓
BMP	Bitmap image set		✓
OFF	Surface mesh		✓
TNS	Local directionality vector field		✓
Geometric Skeletons			
MRC	Binary density volumes	✓	✓
OFF	Surface mesh	✓	✓
SSE Annotations			
VRML	VRML Cylinders and surfaces	✓	✓
WRL	VRML Cylinders and surfaces	✓	✓
SSE	DeJaVu SSE annotations	✓	✓
Sequence and C_α Atom Locations			
PDB	Protein Data Bank files	✓	✓
SEQ	Annotated sequence information	✓	✓

Table 5.1: The input/output file formats supported by Gorgon.

utilize the plethora of computational tools available in these frameworks via the intuitive and interactive Gorgon interface.

5.3.7 Other features

Geometric routines: To give users better flexibility, we also make available other useful geometric tools such as Laplacian smoothing, volume cropping, resizing, down-sampling and normalization.

Session support: Molecular modeling is a multi-step, time-consuming process. Therefore session support is an essential functionality that would allow users to save their current work so that it can be resumed later, or shared between collaborators. We provide a transparent session functionality that allows users to save their work-in-progress as well as camera and other visualization information. The session information is stored as remarks using the PDB file format, giving the users the option of using partial backbone information with other molecular modeling tools.

Sequence based SSE prediction: While Gorgon does not natively implement any sequence based SSE prediction algorithms, we provide an interface that allows a user to dispatch their sequence information to the JPred [27], PSIPred [53], and Scratch [24] servers. In the future we would like to integrate such methods into Gorgon in a much more seamless manner, providing users a much better experience.

5.4 System design and implementation

5.4.1 Design

Gorgon was designed specifically to handle the modeling pipeline of a single molecule at any given time. For each data element in this pipeline (i.e. volume, skeleton, set of SSEs, C_α backbone) we have a visualizer that is responsible for the user-interface elements, and a renderer that is responsible for the OpenGL rendering routines. We enforce each Visualizer and Renderer class to be a singleton (only one instance exists at any given time), thereby ensuring that only data related to a single molecule is loaded at any given time.

Complex activities that require multiple user interactions and intensive computation need to be kept separate from the remainder of the application to allow independent implementation as well as to minimize memory consumption. For this purpose, we model these activities as engines, where each engine is responsible for implementing a specific functionality, and is instantiated only when necessary.

All menu driven tasks in Gorgon are modeled using the command design pattern, and handled in the MenuManager and ActionManager classes. This lets each action to be implemented in an independent fashion while also allowing the same functionality to be used multiple times throughout the application without code duplication. Additionally, actions and menu items can be queried using their unique, statically-assigned identifiers. This makes it easier for plug-ins and scripts to access and trigger internal Gorgon functionality.

For molecular modeling, it can be useful to observe a model from different perspectives. For this purpose we have designed Gorgon in such a way that multiple cameras can be used

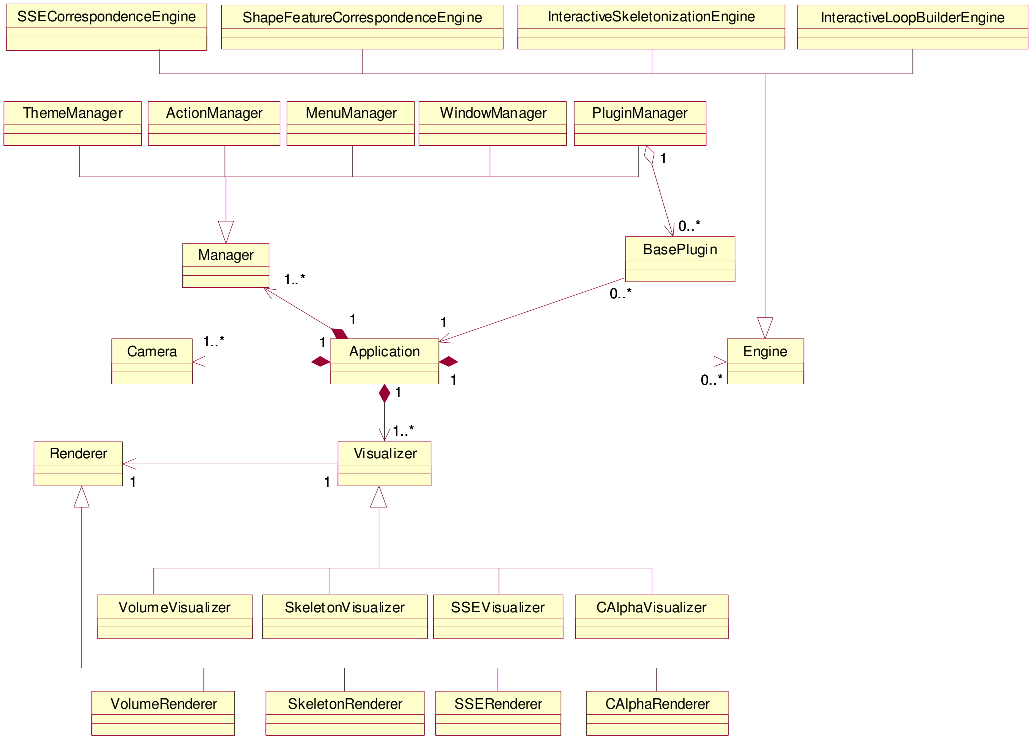


Figure 5.5: The UML model showing the high-level design elements of the Gorgon system.

to observe the same scene from different viewpoints. While the current implementation only makes use of one such camera, we expect to utilize this design to provide a 3D viewing experience in the future. Figure 5.5 shows a high-level UML model that captures the more important design elements of the Gorgon system.

5.4.2 Implementation

The implementation decisions for Gorgon were based on four major requirements: interactive performance, platform independence, scripting support and code reusability. We achieved the performance goals by implementing all of the computationally intensive routines using C++, and by also using efficient data structures such as priority queues, octtrees, hash-maps etc. While C++ allows high-performance computation, it does not offer an easy-to-use model for building user-interface elements. For this purpose, we use Python

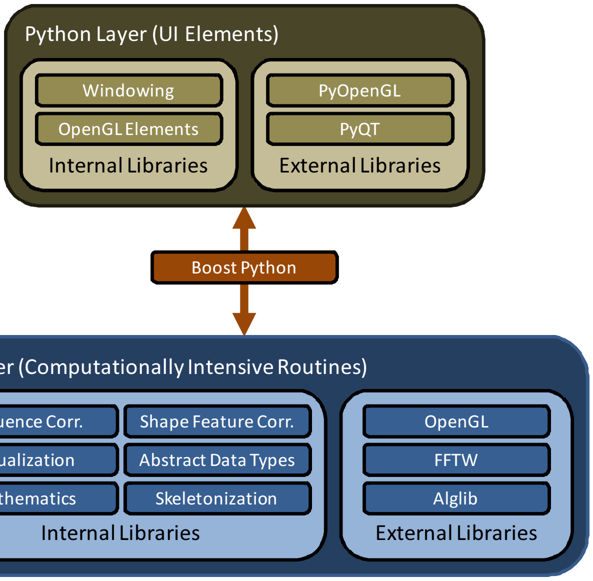
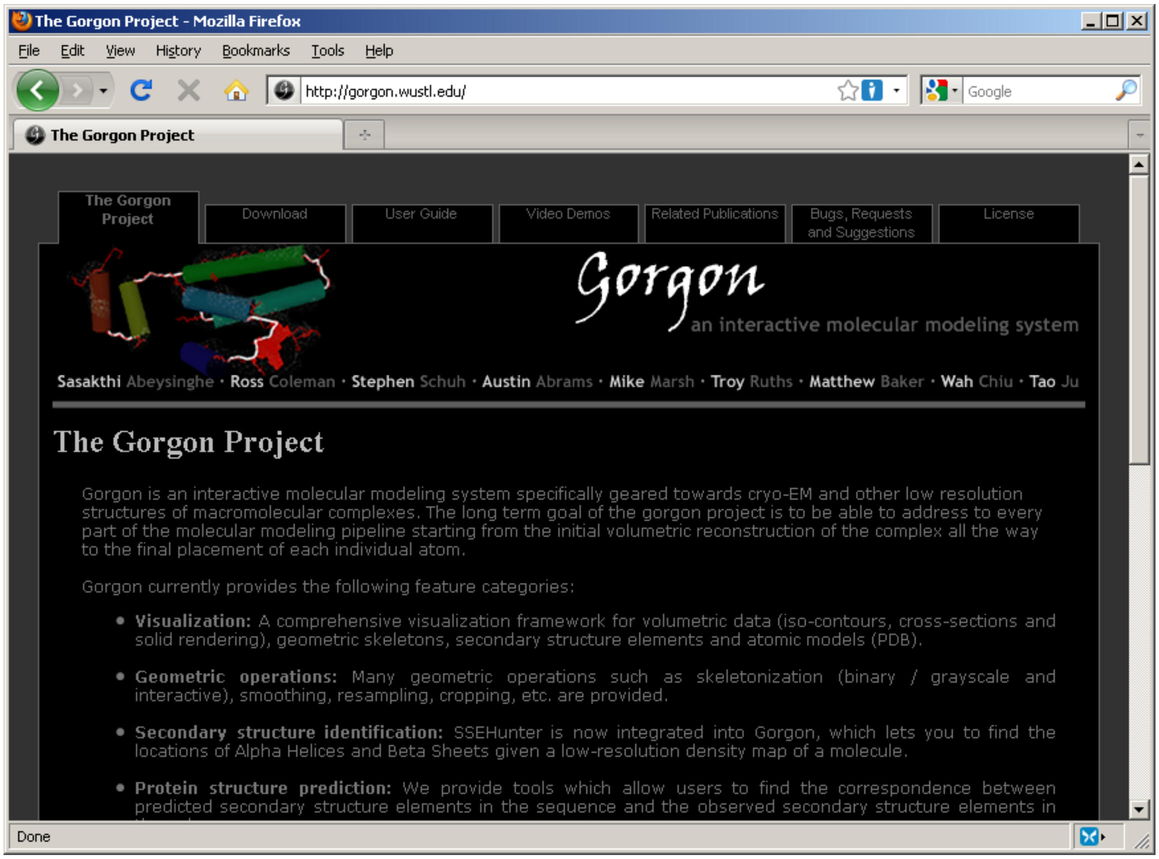


Figure 5.6: The C++ and Python layers, and the libraries used within each layer of Gorgon.

together with PyQT that has the added benefit of utilizing a modern operating system specific look-and-feel while being platform independent. The use of Python also allowed us to provide scripting support that let users write their own tools as well as interface with external software.

Implemented using 52,460 lines of code in 113 files, we can conceptually structure our C++ implementation layer into six main components. The *Abstract Data Types* component implements many computationally efficient data structures and algorithms, while the *Mathematics* component implements mathematical constructs such as complex numbers, Eigen decomposition, singular value decomposition, and also provides an interface to external math libraries such as Alglib and FFTW. The *Visualization* component utilizes OpenGL to implement the rendering routines and the *Skeletonization*, *Sequence Correspondence* and *Shape Feature Correspondence* components implement the methods outlined in Chapters 2, 3 and 4 respectively.

The python layer consists of 14,226 lines of code in 67 files, and houses two main components. The *Windowing* component implements platform-independent window management and messaging functionality, and the *OpenGL elements* component uses PyOpenGL to enable simple rendering options such as camera parameters and model colors. Finally, we utilize Boost-Python to interface between the C++ and Python layers. Figure 5.6 outlines


The Gorgon Project - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://gorgon.wustl.edu/
Google
The Gorgon Project
The Gorgon Project
Download
User Guide
Video Demos
Related Publications
Bugs, Requests and Suggestions
License
Gorgon
an interactive molecular modeling system
Sasakthi Abeysinghe · Ross Coleman · Stephen Schuh · Austin Abrams · Mike Marsh · Troy Ruths · Matthew Baker · Wah Chiu · Tao Ju
Done

The Gorgon Project

Gorgon is an interactive molecular modeling system specifically geared towards cryo-EM and other low resolution structures of macromolecular complexes. The long term goal of the gorgon project is to be able to address to every part of the molecular modeling pipeline starting from the initial volumetric reconstruction of the complex all the way to the final placement of each individual atom.

Gorgon currently provides the following feature categories:

- **Visualization:** A comprehensive visualization framework for volumetric data (iso-contours, cross-sections and solid rendering), geometric skeletons, secondary structure elements and atomic models (PDB).
- **Geometric operations:** Many geometric operations such as skeletonization (binary / grayscale and interactive), smoothing, resampling, cropping, etc. are provided.
- **Secondary structure identification:** SSEHunter is now integrated into Gorgon, which lets you to find the locations of Alpha Helices and Beta Sheets given a low-resolution density map of a molecule.
- **Protein structure prediction:** We provide tools which allow users to find the correspondence between predicted secondary structure elements in the sequence and the observed secondary structure elements in

Figure 5.7: A screenshot of the Gorgon website.

this layered implementation structure as well as the internal components and external libraries used within Gorgon.

5.5 Maintenance and distribution

The source code for Gorgon is maintained using a CVS repository. This allows parallel development, version history, release tagging and other useful functionality essential for working in a team environment. Gorgon is developed as a collaboration between Washington University in St. Louis and Baylor College of Medicine, and the development team currently consists of four graduate students[15] and three faculty advisors[16]. We also have

[15] Sasakthi Abeysinghe, Stephen Schuh, Ross Coleman, Austin Abrams

[16] Tao Ju, Matthew Baker, Wah Chiu

]

two alums[17], who have contributed to the development of Gorgon, but are no longer working on the project.

To facilitate alpha testing, we have created shell scripts that automatically create executables and make them available via the Gorgon website on a nightly basis. These scripts first download the latest source code from CVS, generate platform-dependant make files using CCMake, compile them using their native C++ compilers[18], and then package the Python code and dependencies using Py2Exe for Windows, Py2App for MacOS and Freeze for Linux. This automated nightly-build process allows us to quickly identify cross-platform build errors, and lets interested users and research collaborators immediate access to the most recent features.

For the general audience, Gorgon is freely available for academic and non-commercial use under the creative commons public license. We maintain an annual release cycle with major releases occurring once every year, and minor releases approximately once every six months. Gorgon is currently in its third release cycle and is available for download as stand-alone 32 or 64 bit executable for the Windows, MacOS and Linux platforms via our website.

The Gorgon website, available via http://gorgon.wustl.edu, is a comprehensive resource for all Gorgon users. Here we have all major and minor releases, sample data sets, a user guide, video tutorials and a bug tracking system that allows users to submit errors or feature requests. To-date, we have 191 registered users from 27 different countries who have downloaded Gorgon.

5.6 Conclusion and discussion

In this chapter we discussed Gorgon, an interactive molecular modeling system that can be used to visualize, analyze and quickly build C_α backbone models from intermediate resolution cryo-em density volumes. Currently in the 2nd major release, Gorgon has 191 registered users from 27 different countries, and is freely available to download for the Windows, MacOS and Linux platforms as 32 and 64 bit binaries. To the best of our knowledge Gorgon has been so far used to model at least two molecular structures [28, 126].

[17] Mike Marsh, Troy Ruths

[18] To compile our C++ source code we use Microsoft Visual Studio for Windows, and GCC for MacOS and Linux.

As described, Gorgon contains a number of unique utilities that allow for the construction of reliable and robust protein structural models from intermediate and near-atomic resolution density maps. In Gorgon, model construction is based on establishing a sequence to structure correspondence using secondary structure elements. While shown to be relatively accurate, there are several caveats. For building *de novo* backbone models, the density map must contain secondary structure elements that can be clearly identified. While loops connecting these secondary structure elements may not be unambiguous, density must be present. Therefore, the resolution of the density map to be modeled must be high enough to resolve these features. As all maps vary in composition, quality and resolution, it is difficult to assign a resolution cut-off for building models. Certainly model building is easier and more reliable at higher resolutions than at resolutions where the pitch of an α-helix or separation of strands in a β-sheet is not visible. However, it is possible to build reliable models even at lower resolutions depending on the resolvable features in the map. It should also be noted that there are potential complications in the highest resolution structures. In particular, feature detection with SSEHunter can be problematic, as the β-sheets look more like a series of parallel loops rather than a thin flat plate.

Future work: Gorgon currently supports a *de novo* approach for protein modeling, and does not use homology information when available. In the future, we envision incorporating tools that can mine the protein data bank for structural homologues, and then utilize the shape feature correspondence algorithm described in Chapter 4 to provide rigid and flexible fitting routines to quickly and accurately build atomic resolution models. We can also take advantage of the wide array of tools (e.g. *Rosetta* [31], *Modeller* [36], *EMAN* [68]) available in the community via the Gorgon plug-in framework to provide a feature rich experience to the structural biologist.

Finally, we would like to leverage on recent advancements in 3D viewing technology (e.g. NVidia 3D Vision System) to provide an actual 3D view of biological data thus making it much easier for users to “see” and therefore model the underlying molecular structure. Section 6.1 outlines the immediate goals towards the next version of Gorgon.

Chapter 6

Conclusion and future work

In this dissertation, we proposed a geometric approach for protein structure prediction from intermediate resolution density volumes obtained using electron cryo-microscopy. More specifically, we address three distinct computational challenges in the molecular modeling pipeline (Figure 6.1), and present *Gorgon*, an interactive molecular modeling system freely available to the public.

The first step towards molecular modeling from cryo-EM is understanding the shape and topology of the density volume. The geometric skeleton is a suitable shape descriptor for this purpose. However, obtaining accurate geometric skeletons from density volumes where the actual segmentation is unknown, and where there is a significant amount of noise, is a challenging computational task. In Chapter 2 we proposed a novel skeletonization method that can capture the shape and topology of the density volume while being robust under noise. The key strength of our method is that it does not require any prior segmentation of the density volume, and therefore can be used to quickly and automatically build geometric skeletons for a wide array of biomedical data sets including, but not limited to cryo-EM.

Given a cryo-EM volume and its geometric skeleton, we can use techniques such as SSE-Hunter [9] to accurately determine the 3D locations of its secondary structure elements. However, to build a molecular model we need an understanding of how those 3D SSEs can be mapped to the actual sequence of amino acid residues. In Chapter 3, we proposed a graph-theory based technique to quickly generate a set of most likely correspondences between the SSEs observed in the volume, and predicted from the sequence. This method is orders of magnitudes more efficient than other techniques, and can also be used in a semi-automatic fashion to incorporate user constraints. Given such a correspondence we then automatically create pseudo backbone trace that can guide a user towards interactively creating the actual C_α backbone.

An alternative approach of creating a high-resolution model from a cryo-EM volume is to fit a structural homologue into the density. The challenge here lies in the fact that proteins undergo many global shape changes based on hinge-like motions when exposed to different bio-chemical stimuli. Due to this reason, high-resolution structural homologues are most often very different in shape to the protein imaged using cryo-EM, requiring *flexible fitting* techniques to find their best fit. While many methods have been proposed for this purpose, they operate at the *micro-level* of atoms and residues, and therefore are very computationally expensive. They are also very sensitive to an initial rigid alignment which is most often very different from the actual shape. In Chapter 4, we proposed a novel technique that can be used to find this *flexible fit* at the *macro-level* of α-helices. Our method in a few seconds can identify isometric clusters of α-helices, determine the transformations needed to align them, and also establish a registration that maps each α-helix in the high-resolution structure to each α-helix observed in the density volume. This *macro-level* registration can then be used to deform the high resolution model at the *micro-level* that can thereafter be used as a better initialization for flexible fitting techniques.

The aforementioned computational methods, together with the other techniques available in literature give the structural biologist a wide choice of tools. However *systems* that incorporate all those tools together allowing users to visually and interactively build molecular structures from cryo-EM volumes are rare, and most often focus on volumes obtained using X-Ray crystallography. For this purpose, in Chapter 5 we designed and developed *Gorgon* as a comprehensive molecular modeling system geared towards the cryo-EM community. Gorgon features many unique computational techniques for molecular modeling, provides a visual, interactive and easy to use interface, and has a complete pipeline for users to start with a cryo-EM density volume, and progressively build the molecular model to the level of a C_α backbone. Furthermore, we provide many I/O formats together with an extensible plug-in framework allowing users to integrate Gorgon with the other external tools that they wish to utilize in their molecular modeling process. Gorgon currently has 191 registered users from 27 different countries, and has been used to model at least two molecular structures [28, 126]

Our work on molecular modeling has led us to several challenges that we plan on addressing in the future, and Figure 6.1 shows how they can be incorporated to the molecular modeling

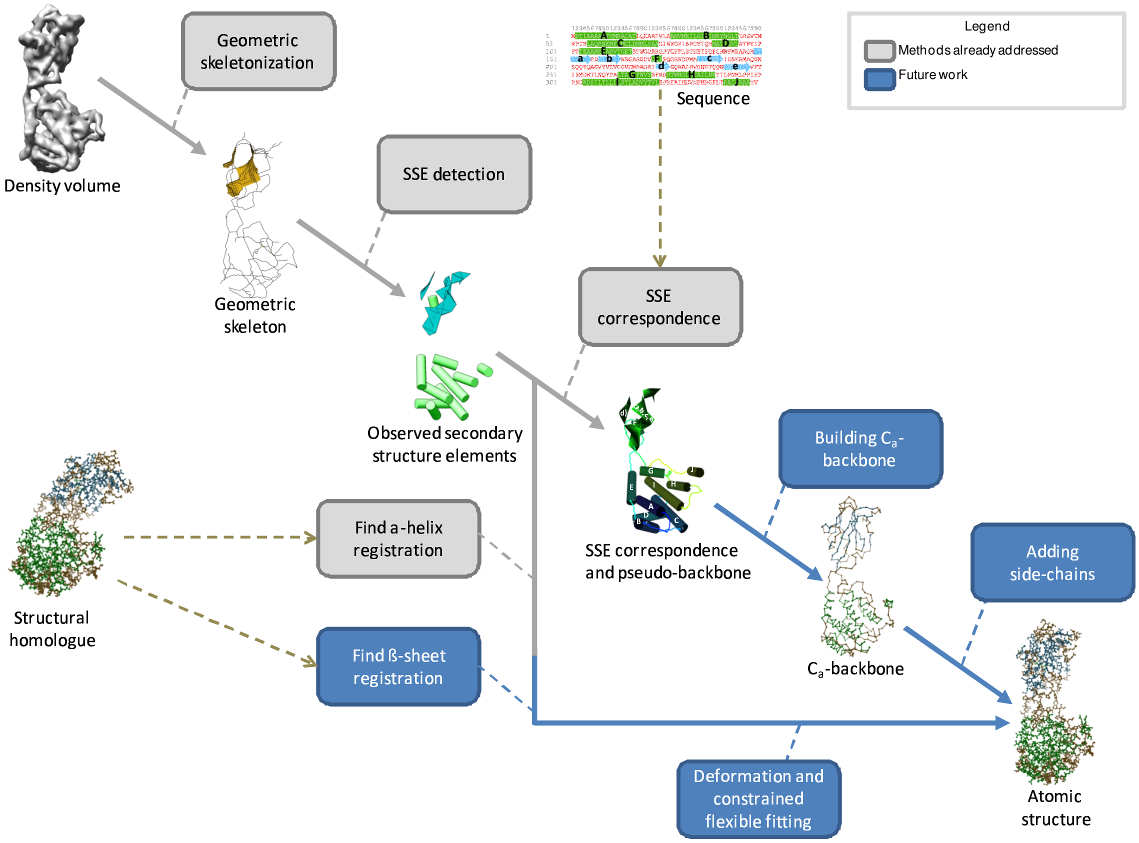

Density volume
Geometric skeletonization
Sequence
Legend
Methods already addressed
Future work
Geometric skeleton
SSE detection
SSE correspondence
Observed secondary structure elements
Building C_a-backbone
Find a-helix registration
SSE correspondence and pseudo-backbone
Structural homologue
Adding side-chains
C_a-backbone
Find ß-sheet registration
Deformation and constrained flexible fitting
Atomic structure

Figure 6.1: Different stages of the molecular modeling pipeline. The methods in gray have been addressed in this dissertation or by collaborators in the past. We envision the methods in blue to be computational challenges to be addressed in the future.

pipeline. We now present these challenges in the context of our work and outline possible solutions.

6.1 Future work

6.1.1 Building physically accurate C_α backbone traces

Using the method described in Chapter 3 we are able to automatically find a set of most likely correspondences between the SSEs observed in the cryo-EM volume and predicted from the sequence. Using each correspondence we generate a pseudo-backbone that can guide users towards interactively building the actual backbone trace. While this interactive process is orders of magnitude faster (hours instead of weeks) than building the backbone without a pseudo-backbone, it still requires a substantial effort by an expert, and motivates the need for a fully automatic backbone generation process. Based on discussions with collaborators[19] in the *ab-initio* modeling community, we have observed that energy minimization techniques can be used to discriminate the difference between physically valid and invalid backbone models with a high degree of accuracy. We are currently working towards using the correspondence search in Chapter 3 to automatically generate a large set (100-1000) of most likely C_α backbone traces, and thereafter using energy minimization techniques from Rosetta [31] to automatically identify the most physically accurate.

6.1.2 Adding side-chains to C_α backbones

Given an accurate C_α backbone trace, there exists methods in literature [104] that utilize density cross-correlation to identify the orientations of side chains based on templates available in Rotomer libraries [66]. While these methods can be used with great accuracy for X-Ray crystallography based density volumes, cryo-EM densities lack the resolution to be used in this form. For this purpose, a more suitable approach would be to first arbitrarily place side-chains on the backbone, and then find the optimal rotations that minimize the global molecular energy. In the future, we would like to investigate the incorporation of such methods into Gorgon, to provide a complete molecular modeling pipeline to build atomic resolution structures from intermediate resolution cryo-EM density volumes.

6.1.3 β-sheet registration for flexible fitting

In Chapter 4, we proposed an approach that can be used to register a high-resolution model with the cryo-EM density based on α-helices. While this registration defines how each high-resolution helix gets mapped to the density volume, we still need a mechanism for mapping β-sheets. Although conceptually our algorithm can be extended to handle different feature types (surfaces in this case), β sheets are more unstable by nature, and can have a wide array of deformations when a molecule undergoes conformational changes. In other words, a β-sheet is more likely to grow/shrink or disappear altogether. This can potentially invalidate the measures we use when evaluating whether feature pairs are isometric. To overcome this we envision using a geometric skeleton induced *neighborhood* criteria

[19] We have an ongoing collaboration with David Baker and Frank Dimaio from the University of Washington towards the problem of automatically determining accurate C_α backbones.

to complement the isometric measures. A method based on this approach would be more robust under large amounts of flexible deformations, and not be limited to quasi-isometric hinge-like motions.

6.1.4 Deforming a high-resolution model based on a SSE registration

Given a *macro-level* registration between secondary structure elements of a high-resolution structural homologue and a cryo-EM density volume, we need to find a set of deformations that can be used to approximately fit the molecule onto the density. Many such techniques exist in the computer graphics community where deformations are smoothly propagated outwards from constrained points (in our case, we need the deformations to be propagated onto the loops while the SSE locations are constrained). However, in the domain of molecular modeling, smooth propagation of the deformation may not be accurate. For example, when undergoing conformational changes, loop segments can bend, unwind and potentially have a completely independent motion to that of the SSEs. For this scenario, a better approach would be to perform the deformation guided by the underlying density. However, density based cross-correlation is a computationally expensive task, and is not suited for this purpose. We envision using a geometric skeleton guided approach for this task, where we propagate the deformation between SSEs by smoothly parameterizing along the geometric skeleton. When given the possibility of multiple skeletal paths, we should chose the one that best satisfies atomic constraints (i.e. C_α atoms should be approximately 3.8Å apart).

Once this deformation has been achieved, we can then use the deformed structure as an initialization for constrained flexible fitting routines [106] to obtain a high-resolution model of the cryo-EM density volume.

6.1.5 Gorgon version 3

Gorgon as described in Chapter 5 is currently in its 2nd major release and has tools to build molecular models at the level of C_α backbones. For the next major release, we envision incorporating the aforementioned work to allow users to build complete atomic-level models. Furthermore, we envision providing plug-ins that allow users to seamlessly integrate Gorgon with other molecular modeling libraries such as *EMAN* [68], *Rosetta* [31] and *Modeller* [36].

Another feature that is most beneficial is the ability to validate the accuracy of models being generated. For this purpose, we envision incorporating tools such as *WhatCheck* [46], *ProCheck* [60] and *MolProbity* [23] that are widely used in the structural biology community for this purpose.

References

[1] Sasakthi Abeysinghe, Ross Coleman, Stephen Schuch, Mike Marsh, Austin Abrams, Troy Ruths, Matthew Baker, Wah Chiu, and Tao Ju. The Gorgon Project. http://gorgon.wustl.edu/, 2008.

[2] Sasakthi Abeysinghe and Tao Ju. Interactive skeletonization of intensity volumes. *The Visual Computer*, 25(5–7):627–635, May 2009.

[3] Sasakthi Abeysinghe, Tao Ju, Matthew L. Baker, and Wah Chiu. Shape modeling and matching in identifying 3D protein structures. *Computer-Aided Design*, 40(6):708–720, June 2008.

[4] Sasakthi S. Abeysinghe, Matthew Baker, Wah Chiu, and Tao Ju. Segmentation-free skeletonization of grayscale volumes for shape understanding. In *Proc. IEEE International Conference on Shape Modeling and Applications SMI 2008*, pages 63–71, 2008.

[5] Rajendra K. Agrawal, Amy B. Heagle, Pawel Penczek, Robert A. Grassucci, and Joachim Frank. EF-G-dependent GTP hydrolysis induces translocation accompanied by large conformational changes in the 70S ribosome. *Nature Structural & Molecular Biology*, 6(7):643–647, July 1999.

[6] Narendra Ahuja and Jen-Hui Chuang. Shape representation using a generalized potential field model. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(2):169–176, 1997.

[7] Dror Aiger, Niloy J. Mitra, and Daniel Cohen-Or. 4-points congruent sets for robust pairwise surface registration. *ACM Transactions on Graphics (TOG)*, 27(3):1–10, August 2008.

[8] Oscar Kin-Chung Au, Chiew-Lan Tai, Daniel Cohen-Or, Youyi Zheng, and Hongbo Fu. Electors voting for fast automatic shape correspondence. In *Computer Graphics Forum (Proc. of Eurographics 2010)*. IEEE, 2010.

[9] Matthew L. Baker, Tao Ju, and Wah Chiu. Identification of secondary structure elements in intermediate-resolution density maps. *Structure*, 15(1):7–19, January 2007.

[10] Helen Berman, Kim Henrick, and Haruki Nakamura. Announcing the worldwide protein data bank. *Nature Structural Biology*, 10(12):980–980, 2003.

[11] Gilles Bertrand. A parallel thinning algorithm for medial surfaces. *Pattern Recogn. Lett.*, 16(9):979–986, 1995.

[12] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, February 1992.

[13] Silvia Biasottia, Simone Marini, Michela Spagnuoloa, and Bianca Falcidienoa. Subpart correspondence by structural descriptors of 3d shapes. *Computer-Aided Design*, 38:1002–1019, 2006.

[14] Harry Blum. A Transformation for Extracting New Descriptors of Shape. In Weiant Wathen-Dunn, editor, *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, Cambridge, 1967.

[15] Alexandra Bonnassie, Françoise Peyrin, and Dominique Attali. Shape description of three-dimensional images based on medial axis. In *ICIP*, volume 3, pages 931–934, 2001.

[16] G. Borgefors, I. Nyström, and Gabriella Sanniti di Baja. Computing skeletons in three dimensions. *Pattern Recognition*, 32(7):1225–1236, 1999.

[17] Gunilla Borgefors. Distance transformations in digital images. *Comput. Vision Graph. Image Process.*, 34(3):344–371, 1986.

[18] H. Bunke. Error correcting graph matching: On the influence of the underlying cost function. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(9):917–922, 1999.

[19] H. Bunke and G. Allermann. Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, 1:245–253, 1983.

[20] Horst Bunke and Bruno T. Messmer. Recent advances in graph matching. *IJPRAI*, 11(1):169–203, 1997.

[21] Pablo Chacón and Willy Wriggers. Multi-resolution contour-based fitting of macromolecular structures. *Journal of Molecular Biology*, 317(3):375–384, March 2002.

[22] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. *Computer Graphics Forum*, 22(3):223–232, 2003. Eurographics 2003 Conference Proceedings.

[23] V. B. Chen, W. B. Arendall III, J. J. Headd, D. A. Keedy, R. M. Immormino, G. J. Kapral, L. W. Murray, J. S. Richardson, and D. C. Richardson. Molprobity: all-atom structure validation for macromolecular crystallography. *Biological Crystallography*, 66(1):12–21, 2010.

[24] J. Cheng, A. Z. Randall, M. J. Sweredoski, and P. Baldi. Scratch: a protein structure and structural feature prediction server. *Nucleic Acids Research*, 33:W72–W76, 2005.

[25] Wah Chiu, Matthew L. Baker, Wen Jiang, Matthew Dougherty, and Michael F. Schmid. Electron cryomicroscopy of biological machines at subnanometer resolution. *Structure*, 13(3):363–372, March 2005.

[26] William J. Christmas, Josef Kittler, and Maria Petrou. Structural matching in computer vision using probabilistic relaxation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(8):749–764, 1995.

[27] Christian Cole, Jonathan D. Barber, and Geoffrey J. Barton. The jpred 3 secondary structure prediction server. *Nucleic Acids Research*, 36:W197–W201, May 2008.

[28] Y. Cong, M.L. Baker, J. Jakana, D. Woolford, E.J. Miller, S. Reissmann, R.N. Kumar, A.M. Redding-Johanson, T.S. Batth, A. Mukhopadhyay, S.J. Ludtke, J. Frydman, and W. Chiu. 4.0-a resolution cryo-em structure of the mammalian chaperonin tric/cct reveals its unique subunit arrangement. In *Proc Natl Acad Sci*, 2010.

[29] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(3):265–298, 2004.

[30] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento. Performance evaluation of the vf graph matching algorithm. In *International Conference on Image Analysis and Processing*, 1999.

[31] Rhiju Das and David Baker. Macromolecular modeling with rosetta. *Annual Review of Biochemistry*, 77:363–382, 2008.

[32] Tamal K. Dey and Wulue Zhao. Approximate medial axis as a voronoi subcomplex. In *SMA '02: Proceedings of the seventh ACM symposium on Solid modeling and applications*, pages 356–366, New York, NY, USA, 2002. ACM Press.

[33] Petr Dokladal, Christophe Lohou, Laurent Perroton, and Gilles Bertrand. A new thinning algorithm and its application to extraction of blood vessels. *Proc. of Biomedsim*, pages 32–37, 1999.

[34] S. Dutta and H.M. Berman. Large macromolecular complexes in the protein data bank: a status report. *Structure*, 13:381–388, 2005.

[35] P. Emsley and K. Cowtan. Coot: model-building tools for molecular graphics. *Acta Crystallogr*, D60:2126–2132, 2004.

[36] Narayanan Eswar, Ben Webb, Marc A Marti-Renom, M. S. Madhusudhan, David Eramian, Min-Yi Shen, Ursula Pieper, and Andrej Sali. Comparative protein structure modeling using modeller. *Currrent Protocols Protein Science*, Chapter 2:Unit 2.9, November 2007.

[37] Felcy Fabiola and Michael S Chapman. Fitting of high-resolution structures into electron microscopy reconstruction images. *Structure*, 13(3):389–400, March 2005.

[38] J. Feng, M. Laumy, and M. Dhome. Inexact matching using neural networks. *Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies, and Hybrid Systems*, pages 177–184, 1994.

[39] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[40] K.S. Fu and J. K. Mui. A survey on image segmentation. *Pattern Recognition*, 13(1):3–16, 1981.

[41] Thomas Funkhouser and Philip Shilane. Partial matching of 3D shapes with priority-driven search. In *Symposium on Geometry Processing*, pages 131–142. IEEE, June 2006.

[42] Ran Gal and Daniel Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Trans. Graph.*, 25(1):130–150, 2006.

[43] Timothy Gatzke, Steve Zelinka, Cindy Grimm, and Michael Garland. Curvature maps for local shape comparison. In *Shape Modeling International*, pages 244–256. IEEE, June 2005. A local shape comparison technique for meshes.

[44] L. Herault, R. Horaud, F. Veillon, and J. J. Niez. Symbolic image matching by simulated annealing. In *Proc. British Machine Vision Conference (BMVC90)*, pages 319–324, 1990.

[45] Masaki Hilaga, Yoshihisa Shinagawa, Taku Kohmura, and Tosiyasu L. Kunii. Topology matching for fully automatic similarity estimation of 3d shapes. In *International Conference on Computer Graphics and Interactive Techniques*, pages 203–212. ACM, 2001.

[46] R.W.W. Hooft, G. Vriend, C. Sander, and E.E. Abola. Errors in protein structures. *Nature*, 381:272–272, 1996.

[47] Radu Horaud and Thomas Skordas. Stereo correspondence through feature grouping and maximal cliques. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(11):1168–1180, 1989.

[48] Fei Hou, Yue Qi, Xukun Shen, Shen Yang, and Qinping Zhao. Automatic registration of multiple range images based on cycle space. *The Visual Computer: International Journal of Computer Graphics*, 25:657–665, 2009.

[49] R. Huber and M. Schneider. A group refinement procedure in protein crystallography using fourier transforms. *Journal of Applied Crystallography*, 18(3):165–169, 1985.

[50] Varun Jain and Hao Zhang. A spectral approach to shape-based retrieval of articulated 3d models. *Computer-Aided Design*, 39(5):398–407, May 2007.

[51] Wen Jiang, Matthew L. Baker, Steven J. Ludtke, and Wah Chiu. Bridging the information gap: computational tools for intermediate resolution structure interpretation. *Journal of Molecular Biology*, 308(5):1033 – 1044, 2001.

[52] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, May 1999.

[53] David T. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *Journal of Molecular Biology*, 292(2):195–202, 1999.

[54] Tao Ju, Matthew L. Baker, and Wah Chiu. Computing a family of skeletons of volumetric models for shape description. *Computer-Aided Design*, 39(5):352–360, 2007.

[55] Gordon Kindlmann, Xavier Tricoche, and Carl-Fredrik Westin. Anisotropy creases delineate white matter structure in diffusion tensor MRI. In *Ninth International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI'06)*, Lecture Notes in Computer Science 4190, pages 126–133, Copenhagen, Denmark, October 2006.

[56] Kitware. Volview. http://www.kitware.com/products/volview.html, 2008.

[57] G J Kleywegt and T A Jones. Template convolution to enhance or detect structural features in macromolecular electron-density maps. *Acta Crystallographica. Section D, Biological Crystallography*, 53(Pt 2):179–185, March 1997.

[58] Yifei Kong and Jianpeng Ma. A structural-informatics approach for mining [beta]-sheets: Locating sheets in intermediate-resolution density maps. *Journal of Molecular Biology*, 332(2):399 – 413, September 2003.

[59] Yifei Kong, Xing Zhang, Timothy S. Baker, and Jianpeng Ma. A structural-informatics approach for tracing [beta]-sheets: Building pseudo-c[alpha] traces for [beta]-strands in intermediate-resolution density maps. *Journal of Molecular Biology*, 339(1):117 – 130, May 2004.

[60] R.A. Laskowski, M.W. MacArthur, D.S. Moss, and J.M. Thornton. Procheck - a program to check the stereochemical quality of protein structures. *J. App. Cryst*, 26:283–291, 1993.

[61] Cyrus Levinthal. Are there pathways for protein folding? *Journal de Chimie Physique*, 65(1):44, 1968.

[62] Yaron Lipman and Thomas Funkhouser. Möbius voting for surface correspondence. *ACM Transactions on Graphics*, 28(3):1–12, August 2009.

[63] A.M. López, F. Lumbreras, and J. Serrat. Evaluation of methods for ridge and valley detection. *Evaluation*, 21(4):327–335, 1999.

[64] Antonio M. López, David Lloret, Joan Serrat, and Juan J. Villanueva. Multilocal creaseness based on the level-set extrinsic curvature. *Comput. Vis. Image Underst.*, 77(9):111–144, 2000.

[65] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics*, 21(4):163–169, 1987.

[66] Simon C. Lovell, J. Michael Word, Jane S. Richardson, and David C. Richardson. The penultimate rotamer library. *Proteins: Structure, Function, and Bioinformatics*, 40(3):389–408, June 2000.

[67] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. Seventh IEEE International Conference on Computer Vision The*, volume 2, pages 1150–1157, 1999.

[68] 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, December 1999.

[69] Thomas Martinetz and Klaus Schulten. Topology representing networks. *Neural Networks*, 7(3):507–522, 1994.

[70] R A Mendelson and E Morris. The structure of f-actin. results of global searches using data from electron microscopy and x-ray crystallography. *Journal of Molecular Biology*, 240(2):138–154, July 1994.

[71] Robert Mendelson and Edward P. Morris. The structure of the acto-myosin subfragment 1 complex: Results of searches using data from electron microscopy and x-ray crystallography. *Proceedings of the National Academy of Sciences*, 94(16):8533–8538, August 1997.

[72] S.S. Mersa and A.M. Darwish. A new parallel thinning algorithm for gray scale images. *Proceedings of the IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing*, pages 409–413, 1999.

[73] B. T. Messmer and H. Bunke. A new algorithm for error-tolerant subgraph isomorphism detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):493–504, 1998.

[74] Kakoli Mitra and Joachim Frank. Ribosome dynamics: insights from atomic structure modeling into cryo-electron microscopy maps. *Annual Review of Biophysics and Biomolecular Structure*, 35:299–317, 2006.

[75] Carolyn A. Moores, Nicholas H. Keep, and John Kendrick-Jones. Structure of the utrophin actin-binding domain bound to f-actin reveals binding by an induced fit mechanism. *Journal of Molecular Biology*, 297(2):465–480, March 2000.

[76] Florian Mueller, Ingolf Sommer, Pavel Baranov, Rishi Matadeen, Matthias Stoldt, Jens Wöhnert, Matthias Görlach, Marin van Heel, and Richard Brimacombe. The 3D arrangement of the 23 s and 5 s rRNA in the escherichia coli 50 s ribosomal subunit based on a cryo-electron microscopic reconstruction at 7.5 resolution1. *Journal of Molecular Biology*, 298(1):35–59, April 2000.

[77] N.J. Nilsson. *Principles of Artificial Intelligence*. Morgan Kaufmann Publishers, 1980.

[78] N H Olson, P R Kolatkar, M A Oliveira, R H Cheng, J M Greve, A McClelland, T S Baker, and M G Rossmann. Structure of a human rhinovirus complexed with its receptor molecule. *Proceedings of the National Academy of Sciences of the United States of America*, 90(2):507–511, 1993.

[79] Kálmán Palágyi and Attila Kuba. A parallel 3d 12-subiteration thinning algorithm. *Graph. Models Image Process.*, 61(4):199–221, 1999.

[80] Sung-Joon Park and Masayuki Yamamura. Two-layer protein structure comparison. In *15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'03)*, 2003.

[81] E.F. Pettersen, T.D. Goddard, C.C. Huang, G.S Couch, D.M. Greenblatt, E.C. Meng, and T.E. Ferrin. Ucsf chimera–a visualization system for exploratory research and analysis. *J Comput Chem.*, 25(13):1605–1612, October 2004.

[82] Pixotec. Slicer dicer. http://www.slicerdicer.com, 1996.

[83] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. In *Computer Vision ECCV 2008*, pages 500–513. ECCV, 2008.

[84] L. Roberts, R. J. Davenport, E. Pennisi, and E. Marshall. A history of the human genome project. *Science*, 291(5507):1195, February 2001.

[85] J.M. Robson. Finding a maximum independent set in time $o(2^n/4)$. Technical report, Laboratoire Bordelais de Recherche en Informatique, 2001.

[86] Alan M. Roseman. Docking structures of domains into maps from cryo-electron microscopy using local correlation. *Acta Crystallographica. Section D, Biological Crystallography*, 56(10):1332–1340, October 2000.

[87] P.K. Saha, B.R. Gomberg, and F.W. Wehrli. Three-dimensional digital topological characterization of cancellous bone architecture. *IJIST*, 11(1):81–90, 2000.

[88] P. K. Sahoo, S. Soltani, A. K.C. Wong, and Y. C. Chen. A survey of thresholding techniques. *Comput. Vision Graph. Image Process.*, 41(2):233–260, 1988.

[89] Helen R. Saibil. Conformational changes studied by cryo-electron microscopy. *Nature Structural Biology*, 7:711–714, 2000.

[90] Andrej Sali. 100,000 protein structures for the biologist. *Nature Structural Biology*, 5:1029–1032, 1998.

[91] A. Sanfeliu and K.S. Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Systems, Man, and Cybernetics*, 13:353–363, 1983.

[92] Rasmus R. Schroder, Dietmar J. Manstein, Werner Jahn, Hazel Holden, Ivan Rayment, Kenneth C. Holmes, and James A. Spudich. Three-dimensional atomic model of f-actin decorated with dictyostelium myosin s1. *Nature*, 364(6433):171–174, July 1993.

[93] L. G. Shapiro and R. M. Haralick. Structural descriptions and inexact matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 3(5):504–519, 1981.

[94] M. Shatsky, R. Nussinov, and H.J. Wolfson. Flexible protein alignment and hinge detection. *Proteins*, 48(2):242–256, 2002.

[95] Damian J. Sheehy, Cecil G. Armstrong, and Desmond J. Robinson. Shape description by medial surface construction. *IEEE Transactions on Visualization and Computer Graphics*, 2(1):62–72, 1996.

[96] Philip Shilane, Patrick Min, Michael Kazhdan, and Thomas Funkhouser. The princeton shape benchmark. In *SMI '04: Proceedings of the Shape Modeling International 2004 (SMI'04)*, pages 167–178, Washington, DC, USA, 2004. IEEE Computer Society.

[97] State University of New York - Stony Brook. Volvis. http://www.cs.sunysb.edu/∼vislab/, 2010.

[98] Karsten Suhre, Jorge Navaza, and Yves-Henri Sanejouand. *NORMA*: a tool for flexible fitting of high-resolution protein structures into low-resolution electron-microscopy-derived density maps. *Acta Crystallographica Section D*, 62(9):1098–1100, September 2006.

[99] H. Sundar, Deborah Silver, Nikhil Gagvani, and Sven J. Dickinson. Skeleton based shape matching and retrieval. In *Shape Modeling International*, pages 130–142, 290, 2003.

[100] S. Svensson, I. Nyström, and Gabriella Sanniti di Baja. Curve skeletonization of surface-like objects in 3d images guided by voxel classification. *Pattern Recognition Letters*, 23(12):1419–1426, October 2002.

[101] Stina Svensson, Ingela Nystrom, Carlo Arcelli, and Gabriella Sanniti di Baja. Using grey-level and distance information for medial surface representation of volume images. *icpr*, 02:20324, 2002.

[102] Florence Tama, Osamu Miyashita, and Charles L Brooks. Flexible multi-scale fitting of atomic structures into low-resolution electron density maps with elastic network normal mode analysis. *Journal of Molecular Biology*, 337(4):985–999, April 2004.

[103] Florence Tama, Willy Wriggers, and Charles L Brooks. Exploring global distortions of biological macromolecules and assemblies from low-resolution structural information and elastic network theory. *Journal of Molecular Biology*, 321(2):297–305, August 2002.

[104] Thomas C. Terwilliger. Automated side-chain model building and sequence assignment by template matching. *Acta Crystallogr D Biol Crystallogr*, 59(1):45–49, January 2003.

[105] Maya Topf, Matthew L. Baker, Bino John, Wah Chiu, and Andrej Sali. Structural characterization of components of protein assemblies by comparative modeling and electron cryo-microscopy. *Journal of Structural Biology*, 149(2):191–203, February 2005.

[106] Maya Topf, Keren Lasker, Ben Webb, Haim Wolfson, Wah Chiu, and Andrej Sali. Protein structure fitting and refinement guided by cryo-em density. *Structure*, 16:295–307, 2008.

[107] Leonardo G. Trabuco, Elizabeth Villa, Kakoli Mitra, Joachim Frank, and Klaus Schulten. Flexible fitting of atomic structures into electron microscopy maps using molecular dynamics. *Structure*, 16(5):673–683, 2008.

[108] W. H. Tsai and K. S. Fu. Error-correcting isomorphisms of attributed relational graphs for pattern recognition. *IEEE Trans. Systems, Man, and Cybernetics*, 9:757–768, 1979.

[109] T. Tung and F. Schmitt. Augmented reeb graphs for content-based retrieval of 3d mesh models. In *Shape Modeling International*, pages 157–166, 2004.

[110] J. R. Ullmann. An algorithm for subgraph isomorphism. *J. ACM*, 23(1):31–42, 1976.

[111] Javier-Ángel Velazquez-Muriel, Mikel Valle, Alberto Santamaría-Pang, Ioannis A. Kakadiaris, and José;-María Carazo. Flexible fitting in 3D-EM guided by the structural variability of protein superfamilies. *Structure*, 14(7):1115–1126, July 2006.

[112] Niels Volkmann and Dorit Hanein. Quantitative fitting of atomic models into observed densities derived by electron microscopy*1. *Journal of Structural Biology*, 125(2-3):176–184, April 1999.

[113] Yuan-Kai Wang, Kuo-Chin Fan, and Jorng-Tzong Horng. Genetic-based search for error-correcting graph isomorphism. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 27(4):588–597, 1997.

[114] Yusu Wang and Leonidas J. Guibas. Toward unsupervised segmentation of semi-rigid low-resolution molecular surfaces. *Geometric Modeling and Processing - GMP 2006*, 4077(2006):129–142, July 2006.

[115] J.S. Weszka. A survey of threshold selection techniques. *Pattern Recogn*, 7:259–265, 1978.

[116] M. Wolf, R.L. Garcea, N. Grigorieff, and S.C. Harrison. Subunit interactions in bovine papillomavirus. *Proc Natl Acad Sci*, 107(14):6298–6303, April 2010.

[117] H.J. Wolfson and I. Rigoutsos. Geometric hashing: An overview. *IEEE Computational Science and Engineering*, 4(4):10–21, 1997.

[118] A.K. Wong, M. You, and A.C. Chan. An algorithm for graph optimal monomorphism. *IEEE Trans. Systems, Man, and Cybernetics*, 20(3):628–636, 1990.

[119] Willy Wriggers and Pablo Chacón. Modeling tricks and fitting techniques for multiresolution structures. *Structure*, 9(9):779–788, September 2001.

[120] Willy Wriggers, Ronald A. Milligan, and J. Andrew McCammon. Situs: A package for docking crystal structures into low-resolution maps from electron microscopy. *Journal of Structural Biology*, 125(2-3):185–195, 1999.

[121] Willy Wriggers, Ronald A. Milligan, Klaus Schulten, and J. Andrew McCammon. Self-organizing neural networks bridge the biomolecular resolution gap. *Journal of Molecular Biology*, 284(5):1247–1254, 1998.

[122] Yinghao Wu, Mingzhi Chen, Mingyang Lu, Qinghua Wang, and Jianpeng Ma. Determining protein topology from skeletons of secondary structures. *Journal of Molecular Biology*, 350(3):571–586, 2005.

[123] Yuzhen Ye and Adam Godzik. Flexible structure alignment by chaining aligned fragment pairs allowing twists. *Bioinformatics*, 19(19):ii246–ii225, 2003.

[124] Zeyun Yu and Chandrajit Bajaj. A structure tensor approach for 3d image skeletonization: Applications in protein secondary structural analysis. In *Proceedings of IEEE International Conference on Image Processing (ICIP'06)*, pages 2513–2516, 2006.

[125] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. van Kaick, and A. Tagliasacchi. Deformation-driven shape correspondence. *Computer Graphics Forum*, 27(5):1431–1439, 2008.

[126] J. Zhang, M.L. Baker, G.F. Schröder, N.R. Douglas, S. Reissmann, J. Jakana, M. Dougherty, C.J. Fu, M. Levitt, S.J. Ludtke, J. Frydman, and W. Chiu. Mechanism of folding chamber closure in a group ii chaperonin. *Nature*, 463(7279):379–383, January 2010.

[127] Juan Zhang, Kaleem Siddiqi, Diego Macrini, Ali Shokoufandeh, and Sven J. Dickinson. Retrieving articulated 3-d models using medial surfaces and their graph spectra. In *EMMCVPR*, pages 285–300, 2005.

[128] Song Zhang, Çagatay Demiralp, and David H. Laidlaw. Visualizing diffusion tensor mr images using streamtubes and streamsurfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):454–462, 2003.

[129] Yefeng Zheng and David Doermann. Robust point matching for nonrigid shapes by preserving local neighborhood structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:643–649, 2006.

[130] Z. Hong Zhou. Towards atomic resolution structural determination by single-particle cryo-electron microscopy. *Curr Opin Struct Biol*, 18(2):218–228, 2008.

[131] Z. Hong Zhou, Matthew Dougherty, Joanita Jakana, Jing He, Frazer J. Rixon, and Wah Chiu. Seeing the herpesvirus capsid at 8.5å. *Science*, 288(5467):877–880, May 2000.

[132] Robert Zwanzig, Attila Szabo, and Biman Bagchi. Levinthal's paradox. *Proceedings of the National Academy of Sciences*, 89(1):20–22, January 1992.

Vita

Sasakthi Senanayaka Abeysinghe

Degrees

B.Sc. (Honors) Information Systems, September 2004
M.Sc. Computer Science, May 2007
Ph.D. Computer Science, May 2010

Professional Societies

Institute of Electrical and Electronics Engineers
Association for Computing Machinery

Journal Publications

Abeysinghe, S.S., Ju, T. (2009). Interactive skeletonization of intensity volumes, *The Visual Computer* **25**(5-7): 627–635.

Abeysinghe, S.S., Ju, T., Baker, M., Chiu, W. (2008). Shape modeling and matching in identifying 3D protein structures, *Computer Aided Design* **40**(6): 708–720.

Conference Publications

Abeysinghe, S.S., Ju, T. (2009). Interactive skeletonization of intensity volumes, *Proc. of CGI 2009* : 627–635.

Abeysinghe, S.S., Baker, M., Chiu, W., Ju, T. (2008). Segmentation-free skeletonization of grayscale volumes for shape understanding, *Proc. of SMI 2008* : 63–71.

Abeysinghe, S.S., Ju, T., Baker, M., Chiu, W. (2007). Shape modeling and matching in identifying protein structures from low-resolution images, *Proc. of SPM 2008* : 223-232.

May 2010

Protein structure from cryo-EM, Abeysinghe, Ph.D. 2010