Finding α-helices in skeletons

Nina Amenta Sunghee Choi Maria E. Jump
Ravi Krishna Kolluri
Thomas Wahl

Technical Report TR-02-27
Computer Sciences Department
University of Texas at Austin [†]

Keywords: Crystallography, Computational Geometry, Object Recognition

October 16, 2002

Abstract

We consider a problem which is part of the process of determining the three-dimensional structure of a protein molecule using X-ray crystallography: given an estimated map of the electron density of the molecule as a function on three-dimensional space, we identify regions which are likely to belong to α-helices. Our approach is to compute a new kind of skeleton - the *power shape* - and then identify the helical substructures within the power shape with a variant of geometric hashing.

1 Introduction

X-ray crystallography is one of the main techniques for determining three-dimensional protein structure. Experimental diffraction data provides the amplitudes of some of the Fourier coefficients of a three-dimensional map of electron density in a crystal of the protein. The phases of the Fourier coefficients are estimated using a variety of experimental and computational techniques. When there is high-resolution diffraction data and the phases are well-estimated, individual atoms are visible in the electron density map and determining the three-dimensional structure is easy. Often, however, only a noisy low-resolution map is available.

[†] Computer Sciences Dept., Austin, TX 78712, USA. Fax: 512-471-8885. Supported by NSF/CCR-9731977 and an Alfred P. Sloan Foundation Research Fellowship. `amenta@cs.utexas.edu`

At this point, a chemist will spend days or weeks at a computer graphics terminal, manually aligning a stick-figure molecular model containing thousands of atoms to the density map. Most of the really time consuming, difficult but decipherable, maps are at between 3 and 4 Åresolution. Finding secondary structures, especially the α-helices, is one of the first steps a human expert takes when aligning the model with the map, and hence it is one of the first steps we should attempt to automate.

Our work: Given a density map represented by a three-dimensional grid of function values as input, we compute an isosurface. We then compute a skeletal representation of the solid bounded by the isosurface, known as the *power shape*, composed of triangles. For each triangle in the power shape, we examine a set S' of nearby power shape vertices and find the helix that best agrees with S' by geometric hashing. If there is sufficient agreement, we report the points as part of a helix. Since there is a direct mapping between the power shape and the isosurface, this corresponds to labeling a section of isosurface as belonging to a helix, as in Figure 1.

We have tested the method successfully on two density maps, one at 3.0 Angstroms and the other at 3.5 Angstroms. At these resolutions α–helices are visible as twisted shapes in the isosurface. See Figure 1.

Importance of the problem: There are at least three ways in which automatically locating α-helices can be useful. First, it can be used as a domain-specific visualization tool. Highlighting helical portions of the isosurface can make things easier for the chemist during manual model building. Second, finding helices is used as part of a density map refinement algorithm. Information about the three-dimensional structure of the molecule is used to improve the estimated phases, thus improving the quality of the map itself. Often reconstruction is an iterative process in which model building alternates with phase improvement. This would be most useful for noisier, lower resolutions maps than those we have considered so far, but our technique might be applicable. Finally, it might be possible to combine automatic geometric interpretation of the density map with AI methods for predicting secondary structure from sequence data to automatically form tentative matches of portions of sequence data to the map.

2 Related work

There is an excellent existing tool for finding structural fragments such as α–helices in electron density maps. The most recent version of Kevin Cowtan's `fffear` program [19] can find helices in *very* low quality low resolution maps (6-8 Angstroms, larger than a single turn of a helix). It searches a discrete set of possible orientations of the fragment. For each orientation, it convolves the map with a filter resembling the fragment, by multiplication in the frequency domain. This is quite efficient, and independent of the fragment size. It takes advantage of the fact that the frequency domain representation is

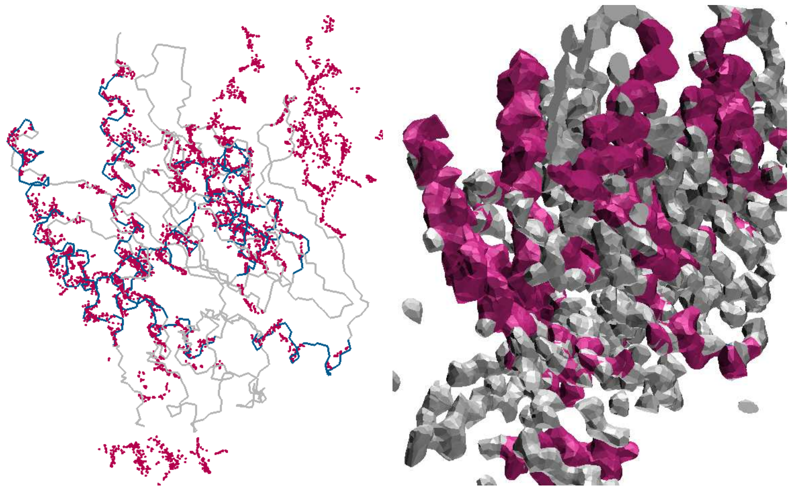

Figure 1: The output of our algorithm on an electron density map of the yMTD enzyme, courtesy of Prof. Jon Robertus (Chemistry, UT Austin). We succeed in labeling vertices of the power shape belonging to each of the the α-helices, with a few false positives. On the left, the power shape vertices which were labeled as helical (purple), with the molecular backbone as reconstructed by the chemists, helices highlighted in blue. Most of the purple points far from blue helices belong to helices in other copies of the molecule nearby in the crystal. On the right, parts of the isosurface corresponding to power shape vertices labeled as helical are purple. The map is at 3.0 Åresolution with an R-factor of .28 (The R-factor is a measure of the mismatch between the map and the constructed molecular model; in this case, when the model is presumed to be good, it can be considered a measure of noise in the map. An R-factor this low indicates a reasonably clean map.)

already given (the density map is constructed from its frequency domain representation). The spatial map of the filtered density is then computed (FFT) and scanned for peak filter responses. Cowtan's approach radically optimizes an earlier exhaustive search algorithm due to Kleywegt and Jones [20]. While still exhaustively searching all possible translations and orientations of the helix, it speeds things up by avoiding a convolution for each orientation-translation pair.

We approach the problem differently. Features (triangles) in the power shape determine positions and orientations which are checked for matches with the fragment. This cuts down the space of transformations examined, but requires comparing the data with the fragment under every transformation searched. In this paper we demonstrate that this approach can successfully locate α-helices at moderate resolutions. It remains to be seen if it can do so more efficiently than the frequency domain approach, in general or in some significant subclass of problems.

Computation of isosurfaces and skeletonization are usual steps in the manual map interpretation process. Existing programs, including `O` [6], `MapMan` [14] and `dm_skeletonization` find one-dimensional skeletons using a voxel-based thinning algorithm, proposed by Greer [5] in 1974. The skeleton is used to help identify the main chain of the protein during manual model building. A different skeletonization procedure was proposed by Leherte et al. [8, 9]. They construct a topological network on the set of critical points of the map, resulting in a sparser 1D skeleton (ours is a denser 2D skeleton). They have had some success in using this skeleton to identifying α-helices at 3 Åresolution (thus, comparable to this work), but it seems unlikely that it would extend to much lower resolutions because the number of critical points decreases with the map resolution.

There is some quite impressive work on the completely automatic determination of the entire 3D structure from diffraction data. The *wARP* system of Perrakis et al. [10] has been successful with density maps in the 1-2.5 Angstrom range. Their approach is based on the 'dummy atom' method of Lamzin and Wilson [7] for phase improvement. A different approach was taken recently by Wang [11], who employs a branch-and-bound algorithm in conformation space.

3 Skeletonization with the medial axis

We begin by describing the power shape construction. The power shape was devised as an approximation of the *medial axis* of a three-dimensional solid, a different kind of skeleton from that approximated by the one-dimensional skeletons used in current systems. Medial axes are somewhat more expressive, and might be of independent interest for visualization and other shape analysis tasks.

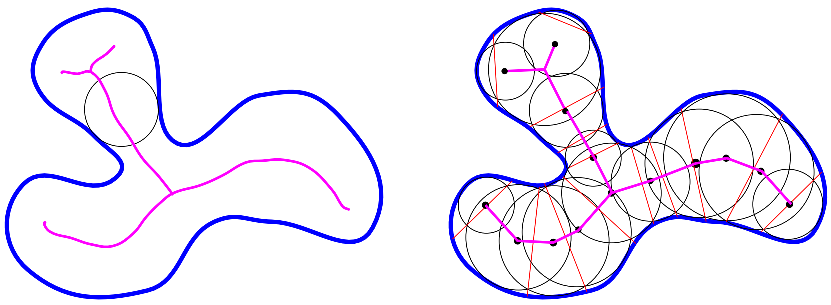

Figure 2: On the left, the medial axis of an object is formed by the centers of the maximal balls contained in the object. In three dimensions, the medial axis is two-dimensional. On the right, the power shape approximates the medial axis by the centers of a finite set of balls. In three dimensions the power shape is made up of triangles.

The medial axis: Given a closed surface F, we say a ball B is empty (with respect to F) if the interior of B contains no point of F. A *medial ball* is a maximal empty ball; that is, it is not completely contained in any other empty ball. The *medial axis* is defined as (the closure of) the set of the centers of the medial balls. In general, the medial axis of a three-dimensional solid is a two-dimensional surface.

Given the medial axis and the radius of the maximal empty ball for every point of the medial axis, the surface can be reconstructed perfectly. In this sense the medial axis contains more information than a one-dimensional skeleton could. For example, big side-chains like tryptophan usually show up as flattened blobs in the isosurface at 3 Å. The medial axis of such a blob is roughly a disk, while the medial axis of a tubular region is closer to a one-dimensional curve.

Power shape: Computing the exact medial axis of a three-dimensional object is difficult. We approximate the medial axis - an infinite union of balls - by a finite union of balls using the Voronoi diagram. To construct the finite union of balls approximating the infinite set of medial balls, we sample the surface and compute the Voronoi diagram of the sample set. We select a set of vertices of the Voronoi diagram far from the sample set as our approximate medial axis points. We discover the adjacencies of these points using the *power diagram*, an kind of weighted Voronoi diagram. The resulting polygonal structure is the *power shape*. A detailed description of the construction, and an analysis of its quality as an approximation of the medial axis, as a function of the quality of the sample set, can be found in our papers [1], [2].

To sample an isosurface from a density map, we extract a set of vertices using the marching cubes algorithm [15] as implemented in `VTK` [16]. We use the vertices of the isosurface to compute the power shape. When the density map is given on a sparse grid, marching cubes returns a sparse sample from the isosurface and the resulting power shape is very rough. Choosing more samples from the isosurface, using a smooth interpolant of the marching cubes vertices gives power shapes which do a much better job of approximating the medial axis; see Figure 3.

Simplification of the power shape: Unfortunately, the medial axis tends to be complicated-looking and unstable with respect to its input. Small perturbations on the surface introduce large "spikes" in the medial axis. On the other hand, *portions* of the medial axis induced by big shape features are quite stable and give a good approximate description of the shape.

To isolate the stable portions, we define a noise threshold ϵ, and define an unstable medial axis feature as one that might disappear if the surface were perturbed by ϵ; note that such a perturbation might induce topological changes in the object. The feature is in danger of disappearing if the points on the surface to which it corresponds are within distance ϵ of each other. To eliminate such features, we remove any ball which touches the surface at points that are within distance ϵ. The remaining balls may still be very redundant. We therefore remove balls which are almost completely covered by other balls, using a greedy algorithm, described in more detail elsewhere [1].

The net effect of this simplification process is to produce a two-dimensional skeletonization which reflects only large shape features. In Figure 3, we show the power shape of an isosurface for two different values of ϵ, one which merely removes quantization noise and another which eliminates many shape

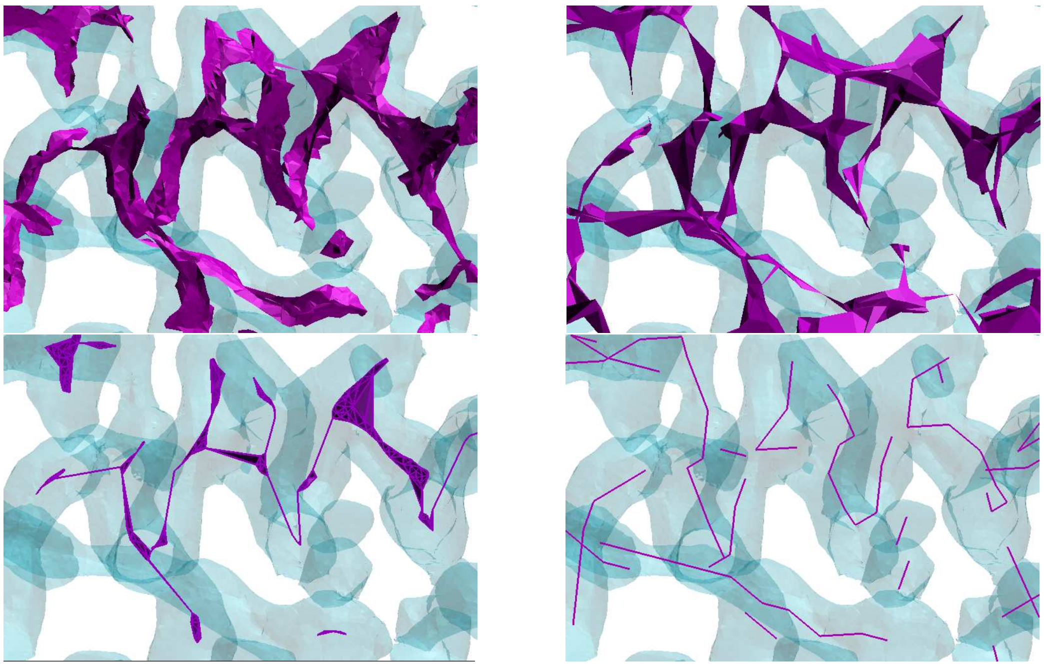

Figure 3: Upper left, the power shape calculated from a dense set of points on the smoothed isosurface (computed by applying two subdivision steps to the output of the marching cubes algorithm), and simplified to remove quantization noise. This power shape forms a good approximation of the medial axis of the (transparent blue) isosurface. Upper right, the power shape we actually use, calculated directly from the lower resolution marching cubes output. Lower left, removing parts of the power shape of the smoothed isosurface (ie. upper left) corresponding to small shape features leaves a skeleton very like the molecular backbone. Lower right, one-dimensional 'bones' skeleton, computed with `MapMan`.

features, leaving only the largest.

4 Search procedure

Our algorithm for recognizing α-helices in the power shape is a variant of *geometric hashing* [12]. Geometric hashing is a general technique for object recognition in computer vision. We will review geometric hashing then explain the speed-ups we can achieve in this special case by using the power shape.

Geometric hashing: The input to geometric hashing is a model M (eg. a helix) and a scene S (eg. the power shape), both represented as sets of points, and set of transformations T which might be applied to M. The output is a matching of portions of the scene with the model. The algorithm consists of a pre-processing phase, which only uses M, and a run-time query phase comparing S and M. Multiple models can be considered simultaneously, with little additional time required in the run-time query phase.

The crucial observation that makes geometric hashing work is that we can express the coordinates of the points in the model M in a basis B defined by a constant number of points of the model, so as to be invariant under the set T of transformations.

For instance, if T consists just of translations, it suffices to consider any point $p \in M$ as the origin of the set B of basis vectors. In that simple case, we match M to S as follows. Choose a point p in the model as the origin of coordinate system given by B. Use the coordinates of each point of M, with respect to B, as the index of an item in a hash table H. This completes the pre-processing phase. In the query phase, for each point q in S, use q as the origin for a basis B. If q is a point in the scene such that translating p to q matches M with a subset of S, corresponding points in the scene and the model will now have the same coordinates. We look up each point in S in the hash table H; if it finds a corresponding point of M, we count one 'vote' for this translation. If the number of votes is equal to the number of points in M, we conclude that is a match, and we output the translation.

To be useful in most settings, the requirements for a match have to be relaxed somewhat to accommodate error. First, the points of M and S might not match exactly. This is solved by rounding the coordinates used to index and look up items in the hash table. It might be that some points of M are not matched by points of S. In general this can be solved by requiring the number of votes to be at least a fixed percentage of the number of points in M. But if the point q corresponding to the correct origin is missing from S then the entire match is missed. This is solved by storing the coordinates of the model points in M with respect to *every* origin $p \in M$ into the hash table. Votes are counted separately for each choice of p; translations matching pairs $p \in M, q \in S$ which receive many votes are output. Note that although the hash table gets larger, the number of hash-table lookups in the run-time query phase remains the same.

Finally, there are some obvious optimizations. Using the relaxations described in the previous paragraph, every transformation matching M to a subset of S will be found multiple times. Once a match is found, the corresponding points of S can be eliminated from further consideration. Also, if M is small (geometrically) with respect to S, only points of S near q need to be looked up in the hash table.

Our algorithm: In our case, we used 80 points distributed along a two-turn segment of an ideal α-helix backbone as the model M and the vertices of the power shape as the scene S, and the rigid motions (rotation and translation) as the set T of transformations. The preprocessing step involves building the hash table H. For this set of transformations we need three non-collinear points to define a reference frame. Given model points x_1, x_2, x_3, we let x_1 be the origin and define the three orthogonal basis axes by by the vectors

$$v_1 = n(\vec{x_2} - \vec{x_1})$$

$$v_2 = n(\vec{x_3} - \vec{x_1} - ((\vec{x_3} - \vec{x_1}) \cdot v_1)$$

$$v_3 = v_2 \times v_1$$

where $n()$ represents normalization. Each non-collinear triple of points in the first turn of our model helix is used to construct a basis and the coordinates of all the points in M are expressed in this basis and stored in the hash table. This preprocessing step needs to be done only once, for the particular helix model. Note that since order matters, every three points need to be taken in each of six permutations.

We use some observations about the structure of power shapes of helices in density maps to speed up the run-time query phase. First, helices tend to be dense, so that the parts of the power shape belonging to a helix are contained in a single connected component of the power shape. And second, because they are tightly wound, helices tend to contain rather large triangles spanning curves or even entire turns of the helix.

In generic geometric hashing, all triples of power shape vertices are tried as bases. But since only one triple from a particular helix in the scene has to be chosen for the helix to be found, and any helix in the scene has many triangles spanning three of its vertices, we limit our attention to triples of points which form triangles in the power shape. The vertices of each triangle only have to be considered in one order, since the basis according to each permuation was used to create entries in the hash table. For each triangle, we use the the connectivity of the power shape to select a subset S' of the power shape vertices in the neighborhood of the basis triangle. We perform a breadth first traversal of the graph formed by the power shape edges, starting at the basis triangle. We stop either when we have exhausted the connected component or at most a constant number c (we use $c = 200$) of vertices have been reached. We then look up only these c points in the hash table H.

Finally, when a match is found, we try to label as many power shape vertices as belonging to the helix as possible. When a basis is successful, meaning that at least one model-basis pair receives many votes, we expand the set of power shape vertices considered using another breadth-first search, and look those up in the hash table as well.

This results in the following algorithm:

```
For each triangle $t$ in the powershape do:
1:      Let $B$ be the orthonormal basis defined by the vertices of $t$
2:      Starting at the vertices of $t$ obtain the neighborhood
            of $t$ by breadth first traversal . This neighborhood forms $S' \subset S$.
3:      Compute the coordinates of all vertices in $S'$ with respect to basis $B$.
4:      Look up each vertex in $S'$ in the hash table
            and vote for (model,basis) pairs.
5:      If the number of votes for a given (model,basis) pair
            is above a threshold:
                a: Label all vertices that voted for this transformation as part of a helix.
                b: Expand the set $S'$ by breadth-first traversal and label as many new points a
```

5 Results

The input files for the run-time query phase were power shapes for the yMTD enzyme shown in Figure 1 and the barley chitinase shown in Figure 4, containing respectively 14,692 and 80,542 vertices and 144,420 and 203,774 triangles. We visually compared the sets of power shape vertices labeled as belonging to helices with the models of the molecular backbones constructed by the chemists, with the helicies on the backbone labeled by the DSSP algorithm [13]. We found that some of the power shape vertices around each of the helices in each molecule were labeled, except for a single-turn helix in the chitinase molecule (a twist in the chitinase backbone is erroneously labeled as helical by DSSP; our algorithm did not label any vertices in that area). We also labeled a few 'false positive' regions near short curves in the backbone. In the images, there appear to be many 'false positives'. This is because each density map contains parts of several molecules which are near the central molecule in the crystal. Most of the labeled vertices which do not appear to be near helices in the backbone are near helices in these other copies of the molecule. About 35% of the vertices of the yMTD power shape were labeled as belonging to helices and about 11% of the vertices in the chitinase power shape.

The run-time query phase of the algorithm as described in the previous section required about 2 hours and 15 minutes on the chitinase and about 2 hours for the yMTD. This large running time is mostly spent searching large portions of the power shape that do not represent a helix.

Recall, however, that only one basis defining a match for the helix in the scene must be found, and many bases are generally found for each helix, so it should be possible to skip some candidate bases and still find every helix. A common optimization in geometric hashing is to randomly choose a small subset of bases to try. Here, we chose instead to heuristically eliminate triangles that are very small (assuming that the tightly-wound helices almost universally contain long triangles) and those that were numerically unsuitable as bases. Using about 30% of the triangles in the chitinase power shape reduced the running time to 40 minutes with no appreciable difference in the quality of the output, as shown in Figure 4.

We also considered a different heuristic, eliminating triangles from consideration if each of their vertices belonged to another triangle which had already formed a basis. This reduced the running time even more, but seemed harder to justify.

6 Discussion

There is a lot of potential here for further work. We have demonstrated that α-shapes can be found in density maps at moderate resolution by examining a skeletal representation. We are currently engaged in further experimentation with this implementation, including randomly choosing bases, experiments with noisier, poorly-phased maps, and a direct comparison with `fffear`.

In addition, there are interesting alternative implementations of both of the two basic building blocks of this approach, skeletonization and discrete shape matching. Geometric hashing is one of several techniques, including the possibly more efficient random sample with consensus [18] and alignment methods [17], for locating a model in a scene. 'Bones' skeletons might be used instead of power shapes, and the very simplified power shapes from smoothed isosurfaces (Figure 3, lower left) might be more sensitive than the rough power shape we are currently using. Medial axis approximations computed by voxel-based methods might also work well, and would probably be faster to compute in this context.

7 Acknowledgments

We thank Prof. Leonides Guibas (CS Stanford) for suggesting that we explore X-ray crystallography.

We are deeply grateful to Prof. Jon Robertus (UT Chemistry) for sharing yMTD and barley chitinase density maps and his view of the model construction process, and to Jeff Almrud (UT Chemistry) for sharing many references and insights into the state of the art in protein crystallography.

References

[1] N. Amenta, S. Choi and R. Kolluri. The power crust. *Manuscript*, 2001. Available at `http://www.cs.utexas.edu/users/amenta/powercrust`.

[2] N. Amenta, S. Choi and R. Kolluri. The power crust, unions of balls, and the medial axis transform. Submitted by invitation to *Computational Geometry: Theory and Applications*, special issue on surface reconstruction. http://www.cs.utexas.edu/users/amenta/pubs/

[3] C. Bajaj, V. Pascucci, D. Schikore. *Visualization of Scalar Topology for Structural Enhancement* Proceeding of the IEEE Visualization, (1998), pp. 51-58.

[4] Blum, H., A transformation for extracting new descriptors of shape, *Models for the Perception of Speech and Visual Form* (Walthen-Dunn, W., ed.) MIT Press, 1967.

[5] J. Greer. Three-dimensional pattern recognition" an approach to automated interpretation of electron density maps of proteins. *Journal of Molecular Biology* **82** (1974) pp. 279–301.

[6] T.A. Jones, J.-Y. Zou, S.W. Cowen, and M. Kjeldgaard. *Acta Crystellographica A***47**(1991)pp.110-119.

[7] V.S. Lamzin and K.S. Wilson. Acta Crystellographica D **49** (1993) pp. 129-147.

[8] L. Leherte, S. Fortier, J. Glasgow, and F. H. Allen. Molecular scene analysis: A topological approach to the automated interpretation of protein electron density maps. Acta Crystallographica, D50:155-166, 1994.

[9] L. Leherte, J. Glasgow, K. Baxter, E. Steeg, and S. Fortier. Analysis of three-dimensional protein images. *Journal of Artificial Intelligence research*, **7** (1997) pp. 125–159.

[10] A. Perrakis, R. Morris, and V.S. Lamzin. Automated protein model building combined with iterative structure refinement. *Nature Structural Biology*, **6**:5 (1999), pp. 458–463.

[11] C. Wang. Determining molecular conformation from distance or density data. PhD thesis, MIT, Feb, 2000.

[12] Y. Lamdan and H. Wolfson. Geometric hashing: a general and efficient model-based recognition scheme. In *Proceedings ICCV '88*, pages 238–249, 1988.

[13] W. Kabsch and C. Sander. *DSSP : Definition of secondary structure of proteins given a set of 3D coordinates*, Biopolymers 22 (1983) , pages 2577-2637

[14] G.J. Kleywegt and T.A. Jones. Halloween ... Masks and Bones. In *From First Map to Final Model*, edited by S. Bailey, R. Hubbard and D. Waller. SERC Daresbury Laboratory, Warrington, pp. 59-66.

[15] W.E. Lorensen and H.E. Cline. Marching Cubes: a high resolution 3D surface reconstruction algorithm, *Proc. of SIGGRAPH '87* (1987), pp 163-169.

[16] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit: An Object-Oriented Approach To 3D Graphics*, Prentice Hall, 1997.

[17] D.P. Huttenlocher and S. Ullman. Recognizing Solid Objects by Alignment with an Image, *Inter. Journ. Comp. Vision* 5(2):195–212 (1990).

[18] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. Assoc. Comp. Mach.*, 24(6):381–395, (1981).

[19] K. Cowtan. Modified phased translation functions and their application to molecular fragment location, *Acta Cryst. D***54**, (1998), pp. 750-756.

[20] G. J. Kleywegt and T.A. Jones. Template convolution to enhance or detect structural features in macromolecular electron-density maps, Acta Cryst., D53, (1997) 179-185.

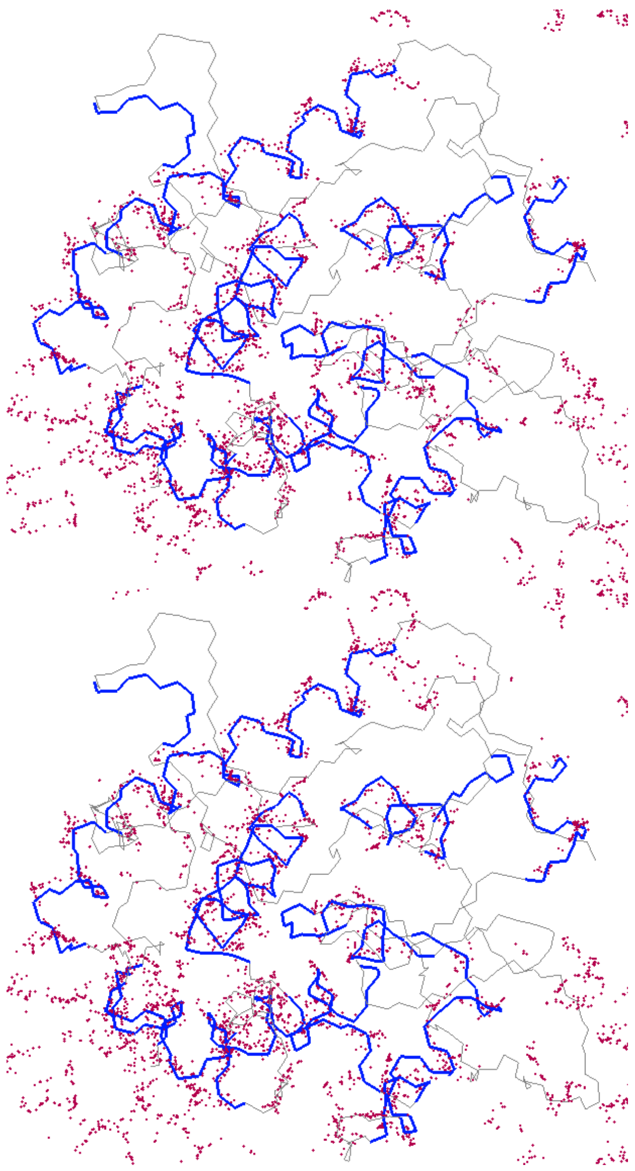

Figure 4: The output of the basic, and an optimized version of the algorithm, on a density map for the barley chitinase protein. The power shape vertices labeled as belonging to helices are shown, along with the molecular backbone as reconstructed by the chemists with the helices hilighted. We fail to locate any helical points near two short regions labeled as helical in the backbone; the one at the upper left was not considered helical by the chemists either. Again, there are many points distributed on helices on other copies of the molecule in the crystal, as well as some false-positives near non-helical regions on the backbone. The input density map was at 3.5 Åresolution.