Visualization of Icosahedral Virus Structures from Reconstructed Volumetric Maps

Zeyun Yu [*] and Chandrajit Bajaj [†]
Department of Computer Science, University of Texas at Austin, Austin, TX 78712, USA
April 09, 2004

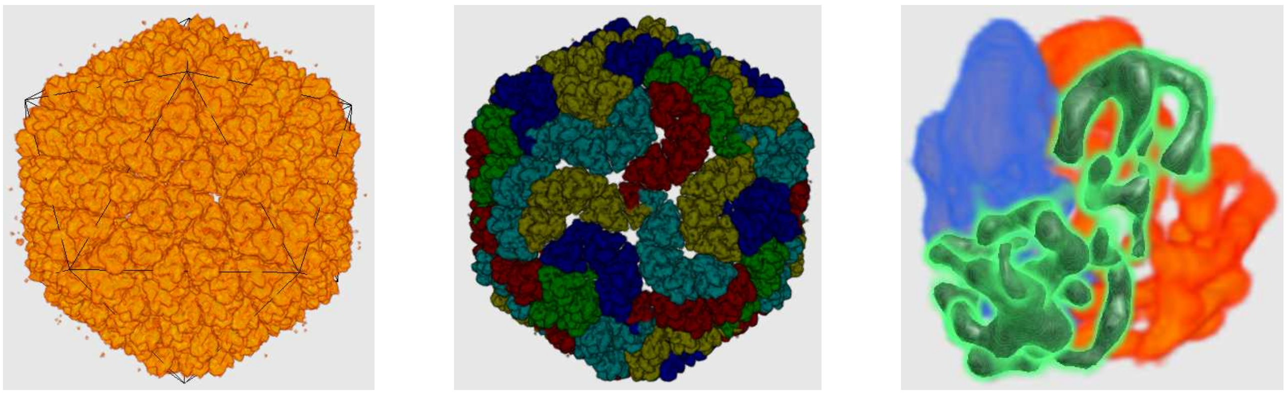

(a) original icosahedral map (b) segmented asymmetric subunits (c) segmented monomer

Figure 1: An example of icosahedral virus structure and its asymmetric subunits

Abstract

In this paper we present an automatic algorithm to segment the asymmetric subunits of an icosahedral density map. This approach is readily applicable to the structural analysis of a broad range of macromolecular structures that are reconstructed using the cryo-electron microscopy (cryoEM) technique. Our algorithm includes three major steps: the detection of critical points, the detection of icosahedral symmetry axes, and the segmentation of asymmetric subunits. The experiments on the real molecular density maps reconstructed by cryoEM technique as well as the synthetic maps calculated from the Protein Data Bank (PDB) demonstrate the high-quality performance of our approach.

CR Categories: I.4.6 [Image Processing and Computer Vision]: Segmentation—Region growing, partitioning; J.3 [Life and Medical Sciences]: Biology and genetics—Protein structure analysis;

Keywords: Volumetric feature detection, Image segmentation, Fast marching method, Symmetry detection, Protein structure analysis

[*]e-mail: zeyun@cs.utexas.edu
[†]e-mail: bajaj@cs.utexas.edu

1 INTRODUCTION

The three-dimensional (3D) structures of molecules are very important for us to study the functions of molecules in many applications. Although the structures of most existing proteins are solved by x-ray crystallography or NMR spectroscopy, they do not give the full picture of a functional biological complex. The study of large macromolecular complexes, such as viruses, ion channels, the ribosome and so on, offers a more complete structural and functional description of the protein machinery. However, it becomes much more difficult to crystallize such large macromolecular complexes. Therefore a non-crystallography technique using cryo-electron microscopy (cryoEM), commonly known as *single particle reconstruction*, provides a powerful tool in revealing the structures of large complexes at sub-nanometer resolutions (5 - 10$\AA$) [1, 3, 16]. By this technique, the 3D structural maps, representing the electron density of molecules, are computationally reconstructed from 2D projection images collected by transmission electron microscopes. The major difficulty of this technique, however, is the unknown orientations of the particles that ”float” in the vitreous ice. Hence, many of the existing macromolecules solved by this technique rely much on the symmetry of the structures being studied. In the present paper, we focus on 3D maps of icosahedral symmetric virus structures.

The reconstruction of the 3D density map of a macromolecule is certainly not the ultimate goal in a biological view of point. We actually have to explore or interpret the detailed structure inside the whole density map. To this end, the segmentation of each individual subunit plays a crucial role in the structure interpretation of a map. In particular, the segmentation of an icosahedral map includes two major tasks: the detection of icosahedral symmetry axes, and the segmentation of each asymmetric subunit. As an example, Figure 1(a) shows the outer capsid layer that is segmented from the whole density map of the rice dwarf virus. A mesh is incorporated showing the icosahedral symmetry. Figure 1(b) illustrates the segmentation of a total of 60 asymmetric subunits, each of which consists of $4\frac{1}{3}$ trimers. Each trimer can be further segmented into three identical monomers. Each monomer, as shown by iso-surfacing in Figure 1(c), is known as the fundamental protein that constitutes the whole virus [23]. Such a hierarchical decomposition provides us a very clear picture about the whole virus structure. The segmented monomer can be further analyzed to reveal the type and functions of the protein but this is outside the scope of the present paper.

Current efforts on the decomposition of an icosahedral map largely rely on manual work with an assistance of a graphical user interface [23, 4]. To accomplish an automatic segmentation, we propose in the following a variant of the traditional fast marching method [11]. we also present an efficient way to detect the symmetry axes of an icosahedral map. An automatic method for seed detection is discussed as well, based on an anisotropic vector diffusion.

The rest of this paper is organized as follows. We begin in Section 2 with an automatic way to detect the critical points of a scalar map. Then we propose a fast method in Section 3 to detect the icosahedral symmetry. Section 4 details the algorithms that segment the asymmetric subunits from the whole icosahedral maps. Various experimental results on both real and synthetic data will be shown in Section 5. Finally we conclude this paper in Section 6.

2 Detection of Critical Points

In general, the critical points of a scalar map include three types: maximal, minimal, and saddle. However, only the maximal critical points are of interest to represent the structures, due to the special property of molecular density maps. This type of critical points can be easily computed from the local maxima of a given scalar map. Since noise is always present in the original maps, a pre-filtering process should be applied. A filter can be either linear or nonlinear. A linear filter (e.g., Gaussian filtering) may destroy some weak features and hence eliminate some critical points. A nonlinear filter [9, 17], however, tends to "flatten" a region, yielding many unwanted critical points. In this section we propose another type of filtering, based on the gradient vector diffusion, and extract the critical points from the diffused vector field.

In general, the gradient vector field generated directly from the original map suffers from noise and cannot be used to detect the critical points. In [18], the authors described a diffusion technique to smooth gradient vector fields. The gradient vectors are represented by Cartesian coordinates and a set of partial differential equations (PDEs) are separately applied to each component of the vectors. The equations are linear or isotropic, and therefore inherit the drawbacks of most linear systems. Another way to diffuse a gradient vector field is based on the polar-coordinate representation of the vectors [20, 21]. The drawback of this method is its computational burden due to the efforts that have to be made to deal with the periodicity of orientation. In the following we propose a set of anisotropic diffusion equations and attempt to address the afore-mentioned problems:

$$\begin{cases} \frac{du}{dt} = div(g(\alpha)\cdot\nabla u) \\ \\ \frac{dv}{dt} = div(g(\alpha)\cdot\nabla v) \\ \\ \frac{dw}{dt} = div(g(\alpha)\cdot\nabla w) \end{cases} \tag{1}$$

where (u, v, w) are initialized with the negative gradient vectors of the original maps. $g(\cdot)$ is a decreasing function and α is the angle between the central vector and its surrounding vectors. For instance, we can define $g(\alpha)$ as follows:

$$g(\vec{c},\vec{s}) = \begin{cases} e^{\kappa\cdot(\frac{\vec{c}\cdot\vec{s}}{\|\vec{c}\|\|\vec{s}\|}-1)} & if\ \vec{c}\neq 0\ and\ \vec{s}\neq 0 \\ \\ 0 & if\ \vec{c}=0\ or\ \vec{s}=0 \end{cases} \tag{2}$$

where κ is a positive constant; $\vec{c}$ and $\vec{s}$ stand for the central vector and one of the surrounding vectors, respectively.

Once the gradient vector field is generated and diffused, we can detect the *source* points, as defined below:

Definition: A point is called *source* if none of its neighbors points to it. In other words, a point A is a *source* if and only if, for any neighbor B of A:

$$\vec{v}_B\cdot\vec{BA}\leq 0 \tag{3}$$

where $\vec{v}_B$ is the diffused gradient vector at B and $\vec{BA}$ is the vector from B to A. All the *source* points are regarded as the critical points, which will be frequently used in the following.

3 DETECTION OF ICOSAHEDRAL SYMMETRY

The symmetry of a shape or object provides fundamental information for shape recognition. The problem of symmetry detection is hence critical in many applications, and has been extensively studied for decades in computer vision. The symmetry detection may be defined differently from one application to the other. Given an object, for example, one may ask: (1) Does this object exhibit certain symmetry? (2) If it does, what type of symmetry is it (reflectional, rotational, radial, etc)? (3) If the symmetry is rotational, what is the folding number and where is the symmetry axis? and so on. A lot of work has been devoted to answering the above questions in the literature (see [15, 8, 19, 22, 2, 13, 5] for details). Most of the past work, however, focused on simple objects, e.g., points, curves, polygons. In case of our macromolecular maps, we presume that the maps exhibit an icosahedral symmetry. Therefore, the symmetry detection can be simplified so that we only need to locate the 5-fold rotational symmetry axes. In addition, we assume that the center of the given map is always identical to the center of its corresponding icosahedra. This requirement is always satisfied in the maps that are reconstructed by the cryo-EM technique.

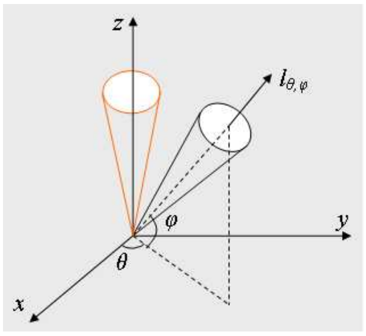


Figure 2: Rotation about an arbitrary axis

Given an axis $l_{\theta,\varphi}$ passing through the origin, where θ and φ are defined in a classical way (see Figure 2) such that $\theta \in [-\pi,\pi]$ and $\varphi \in [-\pi/2,\pi/2]$, a 3D scalar map $f(\vec{r})$ is said to have a 5-fold rotational symmetry about $l_{\theta,\varphi}$ if the following equation holds:

$$f(\vec{r}) = f(R_{(\theta,\varphi,2\pi/5)} \cdot \vec{r}), \text{ for } \forall \vec{r} \tag{4}$$

where the 3×3 matrix $R_{(\theta,\varphi,\alpha)}$ is defined as the coordinate transformation that rotates a point counterclockwise about an axis $l_{\theta,\varphi}$ by an angle of α. In particular, the matrix $R_{(\theta,\varphi,\alpha)}$ is composed of five fundamental coordinate transforms:

$$R_{(\theta,\varphi,\alpha)} = A^{-1} \cdot B^{-1} \cdot \begin{pmatrix} cos(\alpha) & -sin(\alpha) & 0 \\ sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot B \cdot A \tag{5}$$

where matrices A and B are defined as:

$$A = \begin{pmatrix} cos(\theta) & sin(\theta) & 0 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{6}$$

and

$$B = \begin{pmatrix} cos(\frac{\pi}{2} - \varphi) & 0 & -sin(\frac{\pi}{2} - \varphi) \\ 0 & 1 & 0 \\ sin(\frac{\pi}{2} - \varphi) & 0 & cos(\frac{\pi}{2} - \varphi) \end{pmatrix} \tag{7}$$

In order to detect all twelve symmetry axes, corresponding to the twelve vertices of an icosahedra, one can simply correlate the original map with its rotated map and search in the resulting correlation map for the peaks [8]. Obviously this method has a very high computational cost, as the time complexity is $O(NM)$, where N is the number of voxels and M is the number of angular bins used. In our application, N may be as big as 512^3 and M is about 46,000 (we take a uniform sampling on a sphere with a radius of 200-voxel size). Although a number of techniques can be employed to speed up the searching process by reducing the number of the angular bins (e.g., principal axis method [15] or hierarchical approach), it is still expensive if N is very big. In the following, we introduce another way for fast detection of rotational symmetries, given that the folding number is known. Basically the idea is to reduce N, the number of voxels to be tested, by restricting the correlation only to a subset of the critical points instead of the whole set of volume. Let us denote by $C_i, i = 1, 2, \cdots, p$ all the qualified critical points and by $B_j, j = 1, 2, \cdots, q$ all the angular bins. As detailed later, p could be very small by taking only a small subset of the critical points. The algorithm to detect the 5-fold rotational symmetry axes is described as follows:

1. **Compute the scoring function**

   For every $B_j, j = 1, 2, \cdots, q$, compute the corresponding θ_j, φ_j and do **(a)-(b)**.

   (a) For every $C_i, i = 1, 2, \cdots, p$, do **(i)-(ii)**.

      i. compute the rotated points of C_i:

      $$\vec{r}_k(C_i, B_j) = R_{(\theta_j, \varphi_j, 2k\pi/5)} \cdot \vec{C}_i \tag{8}$$

      where $k = 0, 1, 2, 3, 4$, and R is defined in Eq. 5.

      ii. Calculate the deviation of $\{f(\vec{r}_k), k = 0, \cdots, 4\}$ by:

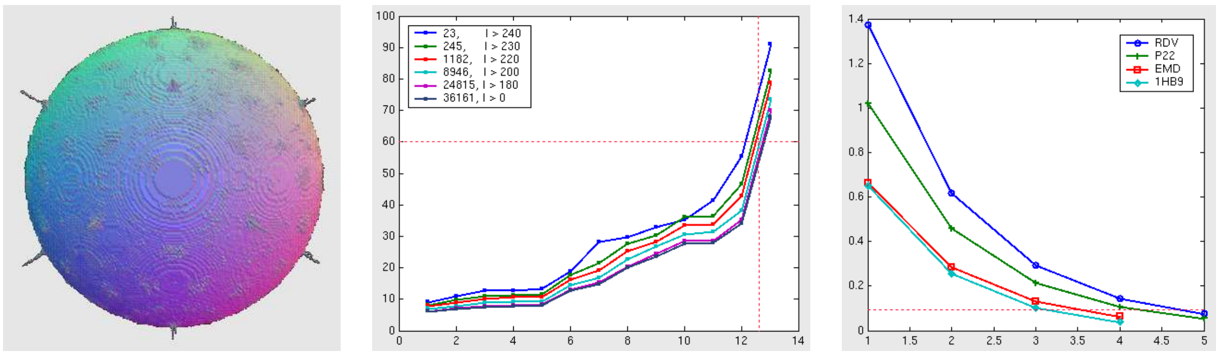
      $$Dev(C_i, B_j) = \frac{1}{5} \sum_{k=0}^{4} |f(\vec{r}_k(C_i, B_j)) - \overline{f}| \tag{9}$$

      where $\overline{f}$ is the average of $\{f(\vec{r}_k), k = 0, \cdots, 4\}$.

   (b) Compute the scoring function for each bin B_j:

   $$SF(B_j) = \frac{1}{p} \sum_{i=1}^{p} Dev(C_i, B_j) \tag{10}$$

   Apparently a smaller score indicates a better chance for that bin to be selected as a 5-fold symmetry axis.



(a) scoring function (b) six sets of critical points (c) number of refinements

Figure 3: Detection of 5-fold Symmetry axes

2. **Locate the symmetry axes**

   Search in the scoring function map $SF(B_j), j = 1, 2, \cdots, q$ for the twelve angular bins with the smallest scores. However, these bins cannot be too close to each other. Otherwise, the one with larger score is discarded and the searching continues until all twelve bins are located.

3. **Refine the symmetry axes**

   The twelve bins obtained may not form a perfect icosahedra. To refine the symmetry axes, we consider each of these twelve bins and rotate the other eleven bins by a certain amount towards the bin under consideration. This amount should depend on the relative positions of the other eleven bins to the one being considered. For a perfect icosahedra, any vertex has five (equally) nearest neighbors, five more next-nearest vertices and one farthest vertex. Hence, the amounts of rotation would be 63.43, 116.57, and 180 in degrees, respectively. Once we rotate all the bins, we then take the average of the new coordinates of the rotated bins (including the bin being investigated) as the new position of the bin being considered. By this way we can find the new positions of all twelve bins. This constitutes one round of the refinement step. It should be repeated until no significant changes happen for all twelve bins (e.g., the total displacement of all twelve bins on the sampling sphere is less than 0.1 between two neighboring refinements).

As an example, we show in Figure 3(a) the (inverted and normalized) scoring function of the outer capsid layer of the rice dwarf virus (RDV) map [23]. We can clearly see the ”peaks” with high contrast. The corresponding symmetry axes can be seen in Figure 1(a). To demonstrate the performance of our algorithm, we test on the same density map of RDV but with different numbers of critical points. The total number of critical points on this map is 36,161. We can choose only a subset of the critical points by requiring that the density on the critical points should be larger than a threshold (e.g., $I > t_0$). In Figure 3(b), six sets of critical points are tested. For each set, we can detect the twelve best symmetry axes by comparing their scores. In Figure 3(b), the scores of the selected symmetry axes are shown in the lower-left window, while the maximal score over all the angular bins for each of the six tests is shown in the upper-right window. The legend gives the number of critical points and the corresponding threshold for each of the six tests, provided that the original density map is normalized to [0, 255]. From this figure, we can see that the number of critical points does not significantly affect the resulting scoring functions. Interestingly the detected symmetry axes (data are not shown here) agree perfectly on all six tests, meaning that the total displacement of all twelve axes on the sampling sphere is less than 0.001 voxel between any two of the tests. Figure 3(c) shows that the typical number of refinements, as described in the third step of our algorithm, is no more than 5. The four datasets we test here will be revisited with more details in Section 5.

4 SEGMENTATION OF ASYMMETRIC SUBUNITS

Before we describe the algorithm for segmenting the asymmetric subunits of an icosahedral map, we briefly review the well-known fast marching method and introduce a variant of this method.

4.1 The Fast Marching Method: A Variant

The fast marching method [6, 11, 12] is a simplified variant of the level set method [12] but it is much faster than the latter one. The basic idea of this method is that a contour is initialized from a pre-chosen seed point, and the contour is allowed to grow until a certain stopping condition is reached. Every voxel is assigned with a value called *time*, which is initially zero for seed points and infinite for all other voxels. Repeatedly, the voxel on the marching contour with minimal *time* value is deleted from the contour and the *time* values of its neighbors are updated according to the following equation:

$$||\nabla T(\vec{r})|| \cdot F(\vec{r}) = 1 \tag{11}$$

where $F(\vec{r})$ is called the *speed function* that is usually determined by the gradients of the input maps (e.g., $F(\vec{r}) = e^{-\alpha\|\nabla I\|}$, where $\alpha > 0$ and I is the original map). The updated neighbors, if they are updated for the first time, are then inserted into the contour.

The traditional fast marching method is designed for a single object segmentation. In order to segment multiple objects, such as the 60-component virus shells or 3-component molecular trimmers, one has to choose a seed for each of the objects. However, assigning only one seed to each object may cause a problem. As shown in Figure 4(a), we consider an image consisting of two components. Two seeds A and B are chosen as seen in the figure. If we let two contours, starting from A and B respectively, grow simultaneously and independently, we may see that the contour from B reaches C before the other contour from A does so. This apparently causes a wrong segmentation of the two components. To remedy this problem, we describe in the following an approach based on an idea of ”re-initialization”. Before we go into the details, we introduce a concept of *marching distance*.

Definition: Given a 3D scalar function $f(\vec{r})$, the *marching distance* between two points A and B in the function domain is defined by:

$$MD_f(A,B) = Min\{\int_{A \to B} e^{\|\nabla f(\vec{r})\|} ds\} \tag{12}$$

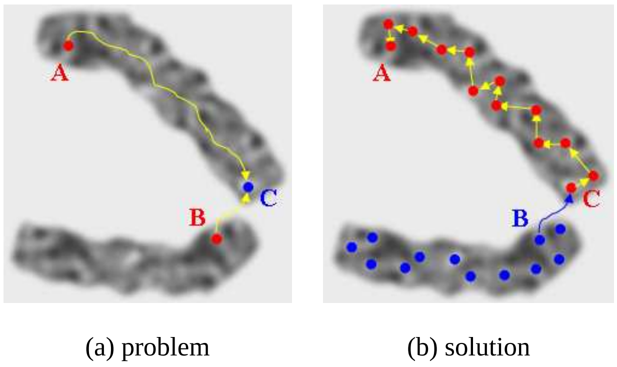
where $\int_{A\to B}$ is the integral along a path from A to B. The minimization is conducted over all the paths from A to B. In particular, if $f(\vec{r})$ is constant, the *marching distance* is degraded to *Euclidean distance* or *geodesic distance*, depending on whether the paths are constrained or not. It can be shown that the *marching distance* satisfies the properties of a classical metric:

- $d(x,y) \geq 0$, and $d(x,y) = 0 \Leftrightarrow x = y$.
- $d(x,y) = d(y,x)$.
- $d(x,y) + d(y,z) \geq d(x,z)$.

The discrete form of Equation 12 is as follows:

$$MD_f(A,B) = Min\{\sum_{\vec{r}=A}^{B} e^{\|\nabla f(\vec{r})\|} - \frac{e^{\|\nabla f(A)\|} + e^{\|\nabla f(B)\|}}{2}\} \tag{13}$$

The second term on the right-hand side of Equation 13 is used to guarantee the *discrete marching distance* satisfying the properties of a metric. Again, the minimization in Equation 13 is conducted over all the paths from A to B, where the paths may be defined on the basis of 6- or 26-voxel neighborhoods. Since the volumes under consideration are always digitized, we always refer to the *discrete marching distances* when we speak of the marching distance. By definition, the marching distance is analogous to the arrival time of a contour, starting from a seed, at an arbitrary point.



(a) problem (b) solution

Figure 4: A variant of the fast marching method

The problem seen in Figure 4(a) can be rewritten as: $MD(A,C) > MD(B,C)$. To remedy this problem, one can divide the marching distance $MD(A,C)$ into a number of small pieces by casting some critical points, denoted by $D_i, i = 1,2,\cdots,n$, orderly on the path from A to C (see Figure 4(b)). In terms of the fast marching method, the arrival time of the contour starting from A is re-initialized to 0 at the nearest critical point D_1 of A. Similarly, the arrival time of the contour starting from D_1 is re-initialized to 0 at the nearest critical point D_2 of D_1, and so on. By this way, the correct segmentation can be achieved as long as enough critical points are used such that the following condition holds (let $D_0 = A$ and $D_{n+1} = C$):

$$Max\{MD(D_i, D_{i+1}), \text{for all } i = 0,1,\cdots,n\} < MD(B,C) \tag{14}$$

The above ”re-initialization” idea can be simply implemented by regarding all the critical points as seeds. In other words, each component (object) will be assigned a number of seeds instead of just one seed. Every seed initiates one contour and all contours start to grow simultaneously and independently. Two contours corresponding to the same object should merge into one contour, while two contours corresponding to different objects should stop on their common boundaries. Unfortunately, we do not have any explicit clue about the correspondence of the critical points (seeds) to the objects. In case of the example seen in Figure 4(b), all we know for sure is that A and B belong to the upper and the lower objects, respectively. Hence we have to find a way to assign the membership of each of the other critical points to one of the objects that we want to segment. Given that at least one seed has been assigned to each object, we summarize in the following the overall algorithm of our proposed variant of the fast marching method:

1. **Detection of the critical points**

   This has already been discussed in Section 2.

2. **Classification of the critical points**

   (a) Let $C_i, i = 1, 2, \cdots, K$ denote all the critical points with unknown membership. If $K > 0$, we take all $C_i, i = 1, 2, \cdots, K$ as the seeds and run the traditional fast marching method until a point with known membership is visited. Denote this point by P.

   (b) The fast marching method produces a $time$ map, representing the arrival time on every voxel (*time* is infinite for unvisited voxels). We start from P and trace downhill by deepest gradient scheme to the valley of this map. We can prove that we must end up at a critical point $C_{i'}, i' \in \{1, 2, \cdots, K\}$, whose arrival time is zero. Apparently the arrival time at P gives the *marching distance* between P and $C_{i'}$. In other words, $C_{i'}$ is the nearest critical point of P, where P has a known membership while $C_{i'}$ does not. Based on our previous discussion, we can assign the membership of P to $C_{i'}$.

   (c) Repeat **(a)-(b)** until $K = 0$. This iterative process can be seen from Figure 4(b), where initially A has known membership. All the other (red) critical points are assigned memberships by marching back to A. We call this process the *fast anti-marching method*, in contrast to the traditional fast marching method. Note also that the obtained graph of the critical points for each object may not necessarily form a linear path, but instead form a forest of trees in general.

3. **Multi-seeded fast marching method**

   Once we classify all the critical points (seeds), we can follow the traditional fast marching method except the following modifications. First, each object may correspond to a number of seeds instead of just one. Secondly, since each seed initiates a marching contour, we must attach to each seed (and accordingly, the marching contour) with a membership index based on the classification of seeds. Once a voxel is conquered by a marching contour, it should be assigned with the same index of the marching contour. Thirdly, two marching contours with the same index should merge into one when they meet, while two marching contours with different indexes should stop on their common boundaries. This

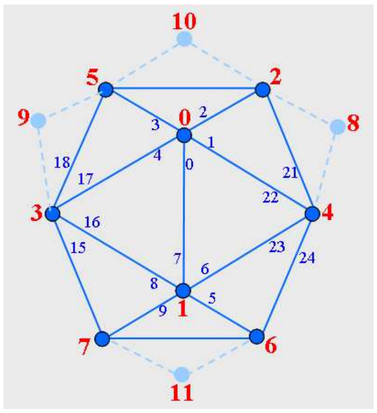


Figure 5: Index numbers of an icosahedral map

multi-label idea has been used elsewhere (e.g., [14]), but apparently the detection and classification of seeds are quite different in our approach, as discussed above.

The above approach can be readily applied to the segmentation of layers from the whole molecular density maps. We shall give examples in Section 5. In the following we describe how this approach is used to segment asymmetric subunits of an icosahedral map.

4.2 Asymmetric Subunits of Icosahedral Maps

The segmentation of asymmetric subunits of an icosahedral map follows the proposed variant of the fast marching method except that, in the second step (classification of critical points), we should take account of the icosahedral symmetry. In other words, once a critical point is assigned with a membership i ($i \in \{0, 1, \cdots, 59\}$), its corresponding 59 symmetric points (may or may not be a critical point) should be assigned with an index differently from $\{0, 1, \cdots, i-1, i+1, \cdots, 59\}$. We label the 12 symmetry axes, as detected in Section 3, as $\{S_j, j = 0, 1, \cdots, 11\}$ and sort them such that $\{S_1, S_2, S_3, S_4, S_5\}$ are the nearest neighbors of S_0, $\{S_6, S_7, S_8, S_9, S_{10}\}$ are the next nearest neighbors of S_0, and S_{11} is opposite to S_0, as seen in Figure 5 (numbers in red). Initially, all the critical points have unknown membership (indexes). We assign 0 to the critical point, denoted by P, which has the highest density value among all the critical points. Then we find all the other 59 symmetric points of P and assign them with indexes orderly and differently from 1 to 59, based on the following pseudo-C code (see the blue numbers in Figure

5 for part of these indexes):

$$\left\{\begin{array}{l}
index = 1; \\
for\,(i=1; i<5; i++)\,\{ \\
\quad Q = R^{0}_{2i\pi/5}(P); \\
\quad \text{assign } index \text{ to } Q; \\
\quad index++;\} \\
\\
for\,(j=1; j<11; j++)\,\{ \\
\quad P' = R_{0\to j}(P); \\
\quad for\,(i=0; i<5; i++)\,\{ \\
\quad\quad Q = R^{j}_{(2i+1)\pi/5}(P'); \\
\quad\quad \text{assign } index \text{ to } Q; \\
\quad\quad index++;\}\} \\
\\
P' = R_{0\to 11}(P); \\
for\,(i=0; i<5; i++)\,\{ \\
\quad Q = R^{11}_{2i\pi/5}(P'); \\
\quad \text{assign } index \text{ to } Q; \\
\quad index++;\}
\end{array}\right. \tag{15}$$

where $R^j_\alpha(P)$ is the rotation of P about the axis S_j by an amount of α, and $R_{j_1\to j_2}(P)$ is the rotation of P from S_{j_1} to S_{j_2}. Note that the total number of critical points with unknown indexes should be deducted by one every time when Q is a critical point.

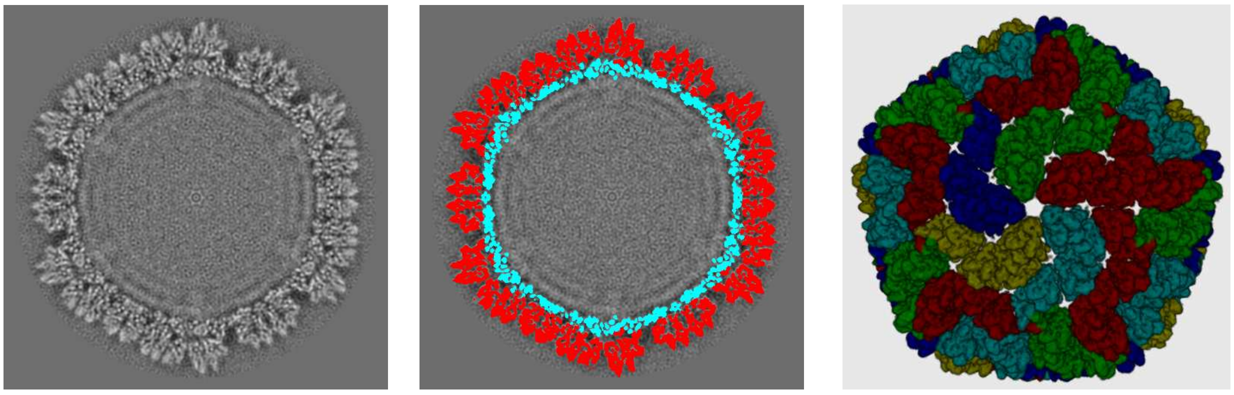
Equation 15 gives the first round of assignment, which guarantees that each of the 60 asymmetric subunits should have at least one critical point with membership assigned. Then we count all the critical points with unknown membership and if the number is positive, we should regard all such points as seeds and run the traditional fast marching method until one of the points with known membership is visited. Denote this point by A. Then, similar to **step (2.b)** of the algorithm discussed in Section 4.1, we should start from A and trace back to the nearest critical point of A that has unknown membership. Denote this critical point by B. Apparently B should be assigned the same index of A. Now the question is how to assign the indexes to the other 59 symmetric points of B? If the index of B is 0, we can simply apply Equation 15 to B. Otherwise, we have to "map" B back to its symmetric point B_0 that corresponds to index 0 (Note that mapping B_0 to B is exactly what we did in Equation 15). Let k be the index of B, $j = k/5$, and $i = k - 5 * j$. We can see that j is the index of the symmetry axis that B corresponds to, while i indicates how much we should rotate about the specified axis S_j in order to map B_0 to B. The following algorithm finds B_0, given B:

$$\begin{cases} if\,(j==0) \\ \quad B_0 = R^0_{-2i\pi/5}(B); \\ \\ else\,if\,(j<11)\,\{ \\ \quad B' = R^j_{-(2i+1)\pi/5}(B); \\ \quad B_0 = R_{j\to 0}(B');\,\} \\ \\ else\,if\,(j==11)\,\{ \\ \quad B' = R^{11}_{-2i\pi/5}(B); \\ \quad B_0 = R_{11\to 0}(B');\,\} \end{cases} \tag{16}$$

where $R^j_\alpha(P)$ and $R_{j_1\to j_2}(P)$ are defined the same as those in Equation 15. Once we find B_0, we can assign 0 to B_0 and run Equation 15 by replacing P with B_0. The above steps are repeated until all critical points have been assigned their corresponding indexes. Finally we run the multi-seeded fast marching method, as discussed in Section 4.1, to segment the 60 asymmetric subunits.

5 RESULTS

In this section we shall test our approach on four macromolecular structures, all of which demonstrate icosahedral symmetries. The first test is on the rice dwarf virus (RDV) [23], which has double capsid layers (or shells). Figure 6(a) shows a cross-section of the original density map. We can see that much noise is present in the map and the boundaries between different layers are unclear. We apply the algorithm described in Section 4.1 to the whole map and segment the outer layer (red color) and the inner layer (light blue color) as shown in Figure 6(b). However, the algorithm in Section 4.1 requires that, before the algorithm gets started, at least one point (may or may not be critical point) should have been correctly assigned to each component that we want to segment. This can be done manually with little effort based on the visual inspection on the biological structure. Once we segment the layers from the whole map, we can focus on the layers trying to segment the asymmetric subunits from each layer using the algorithm described in Section 4.2. Here we only demonstrate the segmentation of asymmetric subunits from the outer layer (see Figure 1(a) for original map of this layer). Figure 6(c) and Figure 1(b) show the segmented asymmetric subunits, viewed from the 5- and 3-fold symmetry axes respectively. It is worthwhile pointing out that we use five colors to distinguish between all 60 subunits such that any five subunits surrounding any 5-fold symmetry axis should have different colors (see Figure 6(c)). In other directions, however, one may see two adjacent subunits having the same color although technically they have different memberships. One can certainly find a more sophisticated coloring scheme to assure any two adjacent subunits having different colors, similar to the well-known world-map coloring problem. Each asymmetric subunit of the outer capsid layer of RDV consists of $4\frac{1}{3}$ trimers and each trimer is composed of three identical monomers. The algorithm



(a) original map (one slice) (b) segmented layers (one slice) (c) asymmetric subunits (5-fold view)

Figure 6: Rice Dwarf Virus (RDV) (at resolution 6.8 $\AA$).

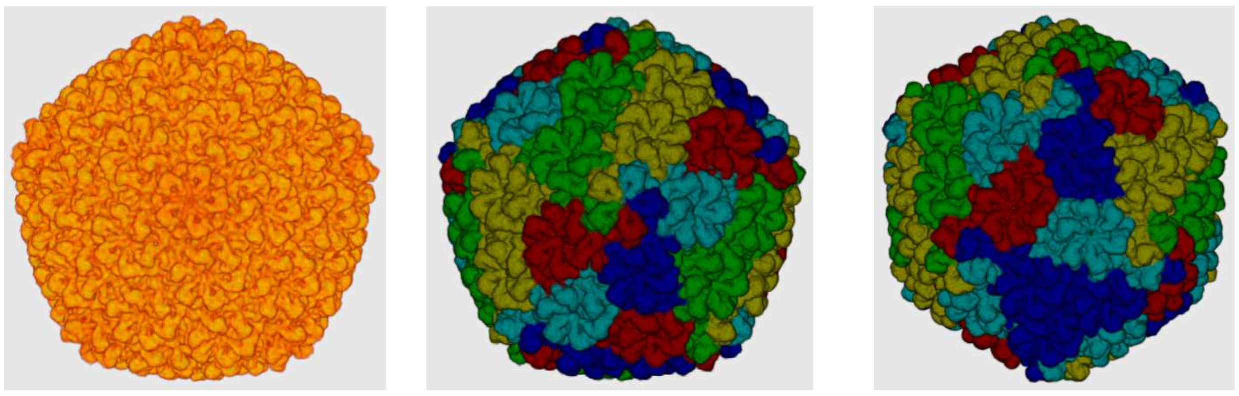
discussed in Section 4.2 can be easily simplified and applied to the segmentation of monomers from each trimer. Figure 1(c) shows the result with three monomers colored differently. In particular, one of the monomers is rendered with its iso-surface. This structure is the fundamental protein, known as $P8$, which constitutes the whole virus structure [23].

Figure 7(a) and Figure 8(a) show the capsid shells of two other icosahedral structures, the bacteriophage P22 [4] and the semliki forest virus [7], where only one capsid shell is present in both structures. Again, we first apply the algorithm in Section 4.1 to segment the shells from the whole density maps and then utilize the algorithm in Section 4.2 to find the asymmetric subunits of the shells. Fig.7(b-c) and Fig.8(b-c) demonstrate the results, viewed from 5-fold and 3-fold symmetry axes.

Our last experiment is conducted on a synthetic map, which is reconstructed by applying 60 rotation matrices (available on VIPER [10]: http://mmtsb.scripps.edu/viper/) to a crystal structure chosen from Protein Data Bank (PDB). The whole crystal map is then blurred to certain resolution, as seen in Figure 9(a). Our segmentation algorithm described in Section 4.2 is applied and the results are shown in Figure 9(b). In contrast, the true segmentation of the asymmetric subunits is shown in Figure 9(c). We see very little difference between our results and the true ones. To further evaluate our approach, we show in Figure 9(d) the critical structure from which we construct the synthetic map. Its blurred map is shown in Figure 9(e) and one of the segmented subunits by our approach is shown in Figure 9(f). We can see that these two maps agree quite well with each other. Finally we point out that all pictures shown in this section, except the cross-section ones in Figure 6, are generated by our own volume-rendering based software.

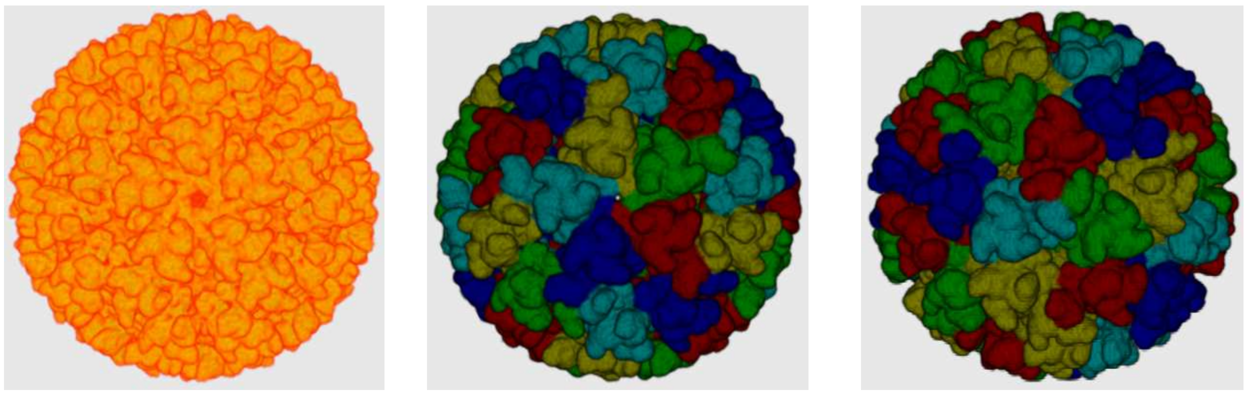
6 Conclusion

In this paper we present an automatic algorithm to detect the asymmetric subunits of an icosahedral density map. Our segmentation approach is a variant of the well-known fast marching method, which is further adapted to incorporate the icosahedral symmetry in order to segment the correct asymmetric subunits. To this end, we also propose an efficient way to detect the symmetry axes of an icosahedral map. An automatic method for seed detection is also discussed, based on an anisotropic vec-



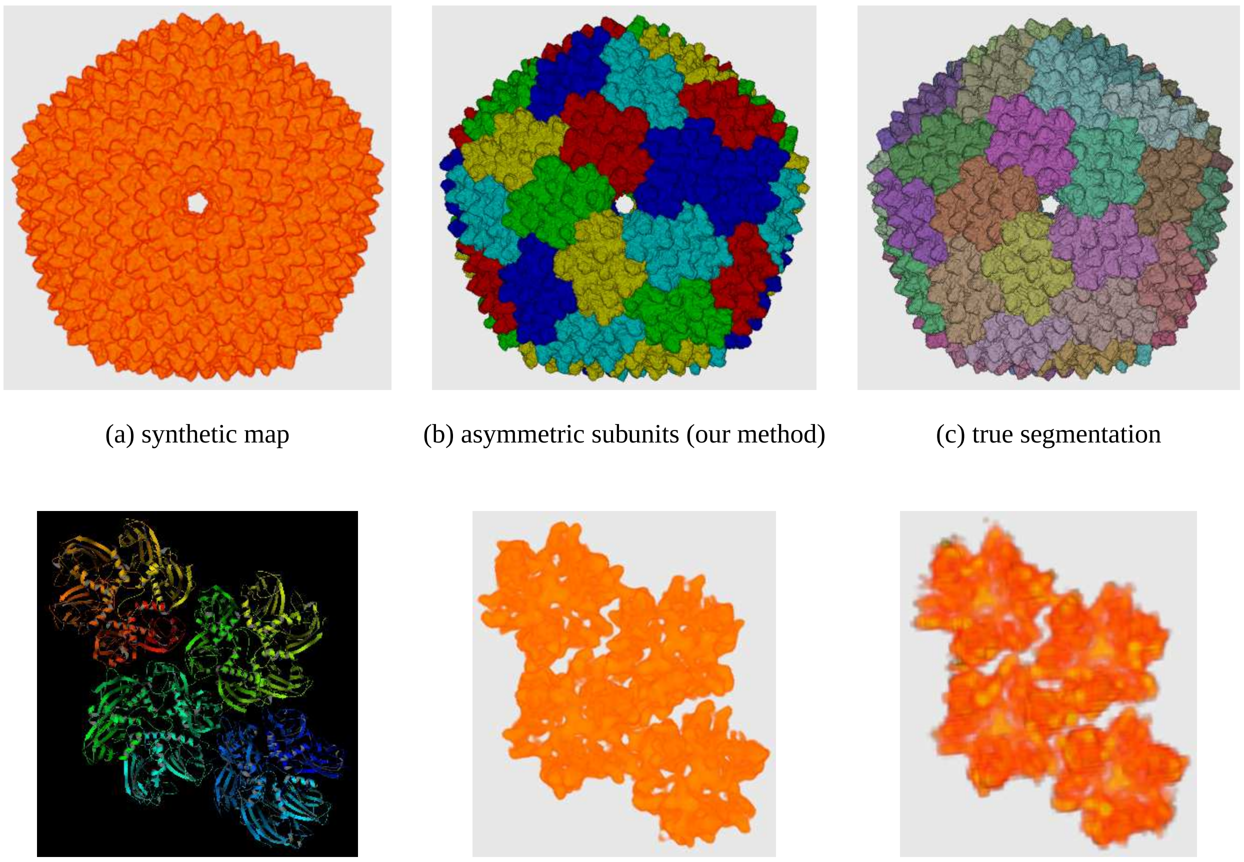
(a) original capsid layer (b) asymmetric subunits (5-fold view) (c) asymmetric subunits (3-fold view)

Figure 7: Bacteriophage P22 after the capsid maturation (at resolution 9.5 $\AA$).



(a) original capsid layer (b) asymmetric subunits (5-fold view) (c) asymmetric subunits (3-fold view)

Figure 8: Semliki Forest Virus (at resolution 9.0 $\AA$).



(a) synthetic map (b) asymmetric subunits (our method) (c) true segmentation

(d) crystal structure (e) blurred map of (d) (f) one subunit of (b)

Figure 9: A synthetic map, Bacteriophage PRD1 Wt Virion, calculated from Protein Data Bank ($PDBID = 1HB9$).

tor diffusion. All experiments we demonstrate here unanimously show high agreements between our results and the manually segmented subunits, as seen in the corresponding references. Our approach can be readily applied to the structure analysis of large macromolecular complexes with icosahedral symmetry, which, however, is currently relying heavily on manual fashions in the existing literature.

ACKNOWLEDGEMENTS

This work was supported in part by NSF grants ACI-0220037, CCR-9988357, EIA-0325550 and a subcontract from UCSD 1018140 as part of the NSF-NPACI project, Interaction Environments Thrust. We are grateful to Dr. Wah Chiu's group, affiliated with Baylor College of Medicine (Houston, TX), for providing us the reconstructed density maps of both RDV and P22. The original map of Semliki Forest Virus is obtained from a public database founded by European Bioinformatics Institute: http://www.ebi.ac.uk/msd/Services.html. We are also thankful to Jun Li, one of our group members, for constructing the synthetic map and its true segmentation color map as seen in Figure 9(a)&(c).

REFERENCES

[1] T. S. Baker, N. H. Olson, and S. D. Fuller. Adding the third dimension to virus life cycles: three-dimensional reconstruction of icosahedral viruses from cryo-electron micrographs. *Microbiology and Molecular Biology Reviews*, 63(4):862–922, 1999.

[2] S. Derrode and F. Ghorbel. Shape analysis and symmetry detection in gray-level objects using the analytical fourier-mellin representation. *Signal Processing*, 84(1):25–39, 2004.

[3] J. Frank. *Three-dimensional Electron Microscope of Macromolecular Assemblies*. San Diego: Academic Press, 1996.

[4] W. Jiang, Z. Li, M. L. Baker, P. E. Prevelige, and W. Chiu. Coat protein fold and maturation transition of bacteriophage p22 seen at subnanometer resolution. *Nature Structural Biology*, 10(2):131–135, 2003.

[5] G. Loy and A. Zelinsky. Fast radial symmetry for detecting points of interest. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(8):959–973, 2003.

[6] R. Malladi and J. A. Sethian. A real-time algorithm for medical shape recovery. In *Proceedings of International Conference on Computer Vision*, pages 304–310, 1998.

[7] E. J. Mancini, M. Clarke, B. Gowen, T. Rutten, and S. D. Fuller. Cryo-electron microscopy reveals the functional organization of an enveloped virus, semliki forest virus. *Molecular Cell*, 5:255–266, 2000.

[8] T. Masuda, K. Yamamoto, and H. Yamada. Detection of partial symmetry using correlation with rotated-reflected images. *Pattern Recognition*, 26(8):1245–1253, 1993.

[9] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.

[10] V. Reddy, P. Natarajan, B. Okerberg, K. Li, K. Damodaran, R. Morton, C. III Brooks, and J. Johnson. Virus particle explorer (viper), a website for virus capsid structures and their computational analyses. *Journal of Virology*, 75:11943–11947, 2001.

[11] J. A. Sethian. A marching level set method for monotonically advancing fronts. *Proc. Natl. Acad. Sci.*, 93(4):1591–1595, 1996.

[12] J. A. Sethian. *Level Set Methods and Fast Marching Methods (2nd edition)*. Cambridge University Press, 1999.

[13] D. Shen, H. S. Ip, K. T. Cheung, and E. K. Teoh. Symmetry detection by generalized complex moments: a close form solution. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(5):466–476, 1999.

[14] E. Sifakis and G. Tziritas. Moving object localization using a multi-label fast marching algorithm. *Signal Processing: Image Communication*, 16(10):963–976, 2001.

[15] C. Sun and J. Sherrah. 3d symmetry detection using the extended gaussian image. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(2):164–168, 1997.

[16] M. van Heel, B. Gowen, R. Matadeen, E.V. Orlova, R. Finn, T. Pape, D. Cohen, H. Stark, R. Schmidt, M. Schatz, and A. Patwardhan. Single-particle electron cryo-microscopy: towards atomic resolution. *Quarterly Reviews Biophysics*, 33(4):307–369, 2000.

[17] J. Weickert. *Anisotropic Diffusion In Image Processing*. ECMI Series, Teubner, Stuttgart, ISBN 3-519-02606-6, 1998.

[18] C. Xu and J. L. Prince. Snakes, shapes, and gradient vector flow. *IEEE Trans. Image Processing*, 7(3):359–369, 1998.

[19] R. Yip, W. Lam, P. Tam, and D. Leung. A hough transform technique for the detection of rotational symmetry. *Pattern Recognition Letter*, 15:919–928, 1994.

[20] Z. Yu and C. Bajaj. Anisotropic vector diffusion in image smoothing. In *Proceedings of International Conference on Image Processing*, pages 828–831, 2002.

[21] Z. Yu and C. Bajaj. Image segmentation using gradient vector diffusion and region merging. In *Proceedings of International Conference on Pattern Recognition*, pages 941–944, 2002.

[22] K. Yuen and W. Chan. Two methods for detecting symmetries. *Pattern Recognition Letter*, 15:279–286, 1994.

[23] Z. H. Zhou, M. L. Baker, W. Jiang, M. Dougherty, J. Jakana, G. Dong, G. Lu, and W. Chiu. Electron cryomicroscopy and bioinformatics suggest protein fold models for rice dwarf virus. *Nature Structural Biology*, 8(10):868–873, 2001.