3D Shape Reconstruction of Loop Objects in X-Ray Protein Crystallography

Tilo Strutz

Abstract—Knowledge of the shape of crystals can benefit data collection in X-ray crystallography. A preliminary step is the determination of the loop object, i.e., the shape of the loop holding the crystal. Based on the standard set-up of experimental X-ray stations for protein crystallography, the paper reviews a reconstruction method merely requiring 2D object contours and presents a dedicated novel algorithm. Properties of the object surface (e.g., texture) and depth information do not have to be considered. The complexity of the reconstruction task is significantly reduced by slicing the 3D object into parallel 2D cross-sections. The shape of each cross-section is determined using support lines forming polygons. The slicing technique allows the reconstruction of concave surfaces perpendicular to the direction of projection. In spite of the low computational complexity, the reconstruction method is resilient to noisy object projections caused by imperfections in the image-processing system extracting the contours. The algorithm developed here has been successfully applied to the reconstruction of shapes of loop objects in X-ray crystallography.

Index Terms—X-ray protein crystallography, machine vision, object recognition, 3D shape reconstruction, occluding contours, shape from contours, shape from silhouettes.

1 INTRODUCTION

The original reason for using optical cameras at experimental X-ray stations for protein crystallography was to observe the tiny protein crystal for its optimum centering in the X-ray beam. Therefore, the cameras have to be properly aligned with respect to the axis of rotation of the loop holding the crystal. Traditionally, this centering has been performed manually. In order to automate protein-structure determination [1], semiautomated and fully automated centering methods have been developed, e.g., in [2], [3]. The observation of the crystal, however, can also be used for more advanced tasks assisting or improving the recording of the diffraction pattern leading to faster or more accurate structural elucidation, e.g., [4], [5]. One preliminary step toward optimizing the X-ray experiment is to determine the three-dimensional shape of the loop object, which also reveals certain information about the size and shape of the crystal, for example. This paper proposes a new algorithm for 3D shape determination based on recordings of 2D images of the loop object.

The problem of reconstructing 3D shapes of objects from 2D views has been addressed for many years [6]. Aside from methods using active light, e.g., laser scanners, which are not considered here, two main types of approach can be observed, depending on whether the objects under investigation have a textured surface or not. Where there is distinct texture, it is possible to find corresponding object-surface points in pairs of images if the positions of the (moving) camera do not differ too much from each other. The array of all point correspondences forms a so-called disparity map, from which depth information, i.e., 3D coordinates, can be derived. If only two images are involved, the problem relates to binocular stereo vision. More recent approaches in computer graphics applications also attempt to map textures taken from the 2D views onto the reconstructed 3D object or scene model [7], [8], [9].

In the case of nontextured surfaces, these corresponding points cannot easily be found and the reconstruction is performed based on occluding contours [6], [10], [11], [12], [13], [14], [15]. These are the apparent boundaries between pixels belonging to the object and background pixels. They vary depending on the direction of view and are also called profiles. The surface model derived from all available contours via binary back-projection is called the visual hull. In fact, 3D object models generated from the visual hull may be ambiguous if the object surface has concave parts [16] (Fig. 1a). Only convex and saddle-shaped parts of the surface can be correctly identified (Figs. 1b and 1c). Existing approaches using occluding contours differ mainly in terms of the constraints put on the camera motion or its position and viewing directions. In many scenarios, the camera is not aligned with the object of interest, i.e., its orientation and motion relative to the object are unknown. The group led by Robert Cipolla, among others, has developed several methods to derive these parameters from different views of the object [8], [13], [14], [15]. Fortunately, the cameras at current experimental X-ray stations for protein crystallography are aligned and we only have to consider circular motion. This paper, therefore, focuses on low-complexity and robust 3D shape determination and presents a detailed description of a corresponding algorithm.

As an alternative to occluding contours, the shading, i.e., the variation in brightness, can give an indication of the curvature of nontextured object surfaces [17].

In real-life applications, features derived from images are subject to noise and robust approaches are essential. In [18],
for instance, a Kalman filter predicts contours in successive images and, hence, smoothes the surface. The combination of texture-based and contour-based methods typically increases the accuracy and correctness of the derived surface model [8], [9], [18], [19].

All these reconstruction methods rely solely on the visual surface of the object under investigation. This is fundamentally different from tomography methods, which are based on the Radon transformation (strictly: the inverse Radon transformation) and are also able to reconstruct the interior of the object. As the reconstruction via the Radon transform makes the assumption that an infinite number of 2D projections of the 3D structure are available and that their projection directions are ideally uniformly distributed over all possible angles [20], the computational load is relatively high to compensate for missing projections. The Radon transform is the standard reconstruction technique in electron microscopy, for example [21].

This paper addresses the reconstruction of 3D surfaces of loop objects consisting of a nylon loop holding a protein crystal. The approach utilizes occluding contours determined from a sequence of images taken by a camera in a fixed position. Because of the standard set-up at experimental X-ray stations, the optical axis of the camera is perpendicular to the axis of object rotation. As in [7] and [22], the 3D problem is reduced to a 2D problem exploiting the known planar motion of the camera relative to the object. The proposed algorithm for 3D shape reconstruction is especially appealing as it requires—in contrast to the octree method [23] or space carving [9]—only elementary mathematics, has very low computational complexity, and is not sensitive to inaccurately measured contour points. In contrast to tomography-based approaches, as reported in [24], for example, the proposed method does not require special equipment and avoids any damage to the crystal, because the crystal is not exposed to additional radiation.

Next section describes the general set-up and explains the known reconstruction method based on silhouettes, as it is essential for the understanding of the proposed approach. Section 3 simplifies the stated problem utilizing given conditions in the camera’s motion. Section 4 explains the proposed reconstruction algorithm in detail. The last section presents two application examples and discusses the results.

2 General Set-up

The constraints used for the shape-reconstruction method that will be described have not been arbitrarily imposed, but have been given by the application enabling the drastic reduction of computational complexity compared to non-calibrated systems. The three-dimensional loop object rotates around a certain axis and is observed by a camera (see Fig. 2). The optical axis of the camera system is orthogonal to the axis of rotation. Furthermore, the rotation axis is parallel to the vertical axis of the images. The camera takes photos of the object at a certain number of angles of rotation. Based on these photos (or images), the shape (surface) of the object has to be reconstructed. The accuracy of the camera alignment can be assumed to be higher than the accuracy of the coordinates that are derived from these images via the applied image-processing methods.

In principle, a perspective image of the object is projected onto the camera sensor. Since, however, the loop objects are very tiny (diameter < 100 μm), the camera can be assumed to have a long focal length. Scaling and other effects of perspective projection can be, therefore, disregarded and the considerations will be restricted to orthographic projection in the following.

Each projection results in a silhouette of the object, from which the outer contour of the object can be determined (Fig. 3). The contours are generated based on a segmentation-from-motion method, which is able to separate moving objects from arbitrary static background using change detection [25], [26], [27], [28]. In general, these contours are neither smooth nor completely accurate, but imperfect, because of the limitations of the image-processing algorithm. The reconstruction algorithm has to deal with the implications caused by this type of noise.

Although the surface of the objects under investigation is generally somewhat textured, it can only be utilized with restrictions. The apparent texture varies heavily depending on the object position, owing to light reflections. Also, shadows may appear and disappear during rotation making the detection of corresponding points difficult.

Fig. 2. Loop object consisting of a nylon loop holding a tiny protein crystal.
Every point on the object surface is identified by its coordinates \((x, y, z)\) independent of the object position (Fig. 3). In the course of projection, \((x', y')\) coordinates describing the object boundary are measured, while the information about the \(z\) coordinate is lost. \(x'\) is aligned with the horizontal axis of the projection plane and \(y'\) with the vertical one. As the direction of projection is always perpendicular to the rotation axis and the \(y'\) and \(y\) axes are parallel, the recorded \(y'\) coordinate is equal to the \(y\) coordinate, while the relation between \(x'\) and \(x\) is dependent on the projection angle.

The projections differ only in one angle \(\varphi\), i.e., all directions lie in the same plane. This constraint is fulfilled by simply rotating the object around an axis which is parallel to the \(y\) axis. The rotation angle for each projection has to be recorded, as it is required for the reconstruction procedure later on. Please note that the object is typically not centered on the rotation axis.

The image of the silhouette of a certain projection is discretized yielding a contour consisting of a finite number of coordinates \([x', y'] = [x', y] \in \mathbb{Z}^2\). The coordinates are measured in pixel units. The true physical size of the objects under investigation is dependent on the magnification of the camera lenses.

3 SIMPLIFICATIONS

Since the rotation axis is parallel to the \(y\) axis (and perpendicular to the optical axis of the camera), all object points keep their \(y\) coordinate independent of the rotation angle. Therefore, all points assigned to the same value \(y\) can be regarded as a slice of the object. Accordingly, the problem is reduced to the reconstruction of two-dimensional slices from one-dimensional projections (Fig. 4).

The fundamental task in the process of reconstruction is now to determine the \((x_i, z_i)\) coordinates of a particular slice-boundary point from the pair of recorded values of \(x'_i\) and \(\varphi_i\).

4 RECONSTRUCTION OF COORDINATES

The reliable reconstruction of 2D coordinates of points lying on the borderline of each object slice requires several consecutive processing steps, which will be explained in the following. This includes the estimation of the position of the projected rotation axis \(x' = x_c\) that is required as reference for all other values measured along the \(x'\) axis.

The determination of the coordinates themselves utilizes support lines forming a polygon encompassing the slice. The separation of all support lines into useful and interfering ones is the final and important processing step increasing resilience to noise caused by inaccurate occluding contours.

The geometrical considerations are based on lines, which are tangential to the slice boundary at different projection angles. Fig. 5 shows a top view onto a single slice along the \(y\) axis. The \(x-z\) plane (thick coordinate system) corresponds to the absolute coordinate system as well as to the system of projection with rotation angle \(\varphi = 0\). The \(x'-z'\) planes correspond to projections with \(\varphi > 0\). The points at \(x_0 = x_c\), \(x'_1\) and \(x'_2\) are the measured \(x'_i\) coordinates of the object contour for this particular slice at rotational positions \(\varphi_0\), \(\varphi_1\), and \(\varphi_2\).

4.1 Determination of Center of Rotation

Initially, the center of rotation \((x_c, z_0)\) (with \(x_c = \text{const.}\) for all projections) is not known and must be estimated, because knowledge about \(x_c\) is essential for the reconstruction method described below.

One possible technique for obtaining information about \(x_c\) is to observe slices at two angles \(\varphi\) and \(\varphi + 180^\circ\). For both angles, the coordinates \(x'_j(\varphi)\), which corresponds to \(x'_j = x'_i(\varphi_i)\) in Fig. 5, and the counterpart \(x'_j(\varphi_j)\) from the left side of the slice are measured. The distance \(|x'_j(\varphi) - x'_j(\varphi)|\)
corresponds to the length of the projected line shown in Fig. 4. The center of rotation is computed by

\[ x_c = \frac{x'_R(\varphi) + x'_L(\varphi + 180^\circ)}{2}, \]  

since \( x'_R(\varphi) \) and \( x'_L(\varphi + 180^\circ) \) have opposite positions with respect to the center of rotation in the projection plane.

Without question, the value of \( x_c \) of a single slice depends heavily on the accuracy of the determined contour points. That is why information from all slices of the object under investigation and from all projection directions are averaged in order to get a reliable estimate. A similar technique exploiting the relation between rotational positions at \( \varphi \) and \( \varphi + 180^\circ \) has been proposed in [29] for the case of noncalibrated cameras.

The only disadvantage of this approach is that pairs of measured points are necessary, which have a rotational distance of 180 degrees. In case that this requirement cannot be fulfilled, the center of rotation could alternatively be determined by least-squares approximation of a circular-movement model as described in [2].

### 4.2 Support-Line Approach

The dashed lines in Fig. 5, tangential to the slice boundary, are so-called support lines. They are oriented along the corresponding projection directions and limit all together the perceptible range of the slice.

If all support lines taken from all different projection directions are combined, a polygon can be drawn around the slice (Fig. 6). The shorter the distances between polygon corners, the higher the curvature of the slice.

The entirety of polygons of all slices forms the visual hull of the object.

Fig. 6 also shows that only convex slices can be reconstructed. Concave parts are hidden in any direction of projection, due to occluding parts of the slice. This limitation, however, pertains to single slices only. The 2D shape may vary arbitrarily from slice to slice resulting in concave structures in the \( y \) direction of the object.

#### 4.2.1 Determination of Support-Line Intersections

The new reconstruction algorithm utilizes the fact that the corner points of the reconstructed polygon simply are the intersections of adjacent support lines if they are sorted by increasing rotation angle. We will concentrate on these intersections, because they are essential for the reconstruction method. The intersection points have to be determined as follows:

According to Fig. 5, let \( d_i \) be the distance between the observed boundary point \( x'_i \) of the projected slice and the center of rotation \( x_c \):

\[ d_i = x'_i - x_c. \]  

The determination of \( x_c \) was already explained in Section 4.1. Based on the parameterized form of the straight-line equation

\[ d = x \cdot \cos(\varphi) + \, z \cdot \sin(\varphi), \]

the coordinates \( (x_{ij}, z_{ij}) \) of the intersection of two support lines \( i \) and \( j \) can be computed as

\[ z_{ij} = \frac{d_j \cdot \cos(\varphi_j) - d_i \cdot \cos(\varphi_i)}{\sin(\varphi_j) \cdot \cos(\varphi_i) - \sin(\varphi_i) \cdot \cos(\varphi_j)}, \]

and

\[ x_{ij} = \begin{cases} \frac{d_i - z \cdot \sin(\varphi_i)}{\cos(\varphi_i)} + x_c, & \text{if } |\cos(\varphi_i)| > |\cos(\varphi_j)|, \\ \frac{d_j - z \cdot \sin(\varphi_j)}{\cos(\varphi_j)} + x_c, & \text{otherwise}, \end{cases} \]  

(see the appendix).

Although the technique of support lines seems to be simple and straightforward, it has to be handled with care. As soon as the coordinate defining the position of a particular support line is affected by inaccuracies, the intersection points of adjacent support lines can change their positions significantly leading to an invalid polygon. Moreover, this alteration gets higher, the smaller the rotational step between two projections is, as can be concluded from Fig. 7. In the example of Fig. 8, it can also be seen that the erroneous dot-and-dash support line becomes obsolete, and so do the two dotted lines, because the object is limited by the dashed support line at the right side. The relation between the closeness (with respect to the angle) of projection directions and noise sensitivity of line intersections has also been pointed out in [12].

The following sections propose a method compensating for these coordinate errors.

#### 4.2.2 Smallest Polygon

From Fig. 8, it can be concluded that in particular support lines belonging to overestimated values of \( x'_i \) become dispensable (dot-and-dash line), whereas support lines with underestimated \( x'_i \) still belong to the polygon enclosing
the slice (dashed line). That is, the polygon with the smallest possible length has to be found. This, however, yields reconstructions which are always smaller than the true polygon as soon as one of the \( x_i \)'s is underestimated. Since the intersection points lie typically outside the true slice area, this effect has even less influence.

The algorithm for finding the shortest traverse constructed by support lines works as follows:

- Let \( P \) be the number of projections.
- Determine all \( P \) support lines in terms of pairs \((x'_i, \varphi_i)\). The coordinates \( x'_L \) (see above) are not used in this process for simplicity.
- Determine all \( P \cdot (P - 1)/2 \) intersection points using (4) and (5).
- Exclude all points lying outside the smallest polygon (with respect to Fig. 8), i.e.,
  - exclude all points lying on the right side of the support line at \( \varphi = 0 \),
  - exclude all points lying above at least one of the support lines with angles \( 0 < \varphi < 180^\circ \),
  - exclude all points lying to the left of the support line assigned to \( \varphi = 180^\circ \), and
  - exclude all points lying below at least one of the support lines with angles \( 180 < \varphi < 360^\circ \).
- The remaining intersection points belong to the shortest traverse.

The consideration of the \( x'_L \) would increase the number of points per polygon or, if the coordinate \( x'_L(\varphi_i) \) has always a counterpart at \( \varphi_i + 180^\circ \), the corresponding coordinates could be averaged to possibly improve their accuracy.

### 4.2.3 Comparison of Intersection Points with Support Lines

In order to find the smallest polygon circumscribing the slice, all intersection points must be tested, whether they belong to the polygon or whether they are outside the slice area. This check requires a comparison of the intersection points with all support lines.

Unfortunately, the information about these points and lines are available for coordinate systems, which are rotated against each other. Intersection points are defined by \((x_j, z_j)\) pairs in the absolute system \((\varphi = 0)\) so far, whereas the support lines are described by the rotation angle \( 0 \leq \varphi_i < 360^\circ \) and the distance \( x'_i \) to the origin in the rotated system (Fig. 9). \( i \) enumerates all intersection points, and \( i \) enumerates all support lines. That is, the coordinates have to be converted for comparison.

Let us start with the most simple case that all intersection points \((x_j, z_j)\) have to be compared to the support line corresponding to \( \varphi_i = 0 \), i.e., the dashed line in Fig. 10a. For simplicity, a box-shaped slice is chosen.

If the \( x_j \) coordinate of the intersection point is larger than \( x'_i \) of the support line (which is equal to the intersection of support line and \( x \) axis for \( \varphi_i = 0 \)), then the intersection point is obviously outside the slice area, because the support line marks already the outmost right coordinate. Thus, the first rule is

\[
\text{if } \varphi_i = 0 \text{ AND } x_j > x'_i, \quad \text{then ignore intersection point } (x_j, z_j).
\]

The complementary rule can be formulated for \( \varphi = 180^\circ \) indicating the leftmost point of the slice (Fig. 10b). Please note that the distances between \( x_c \) and the origin of both, the \( x-z \) and the \( x'-z' \) coordinate system, are identical and
that $x'_i$ of the support line in Fig. 10b is measured from the origin of the $x'-z'$ system. For the sake of comparison, $x'_i$ must be converted into the $x-z$, i.e., the absolute coordinate system. The corresponding rule is

$$\text{if } \varphi_i = 180^\circ \text{ AND } x_j < 2 \cdot x_c - x'_i,$$

then ignore intersection point $(x_j, z_j)$.

The geometry becomes a little bit more difficult for arbitrary angles (Fig. 11a). Let us consider a particular support line and a corresponding parallel line crossing the intersection point (dot-and-dash line in Figs. 11a and 11b). If the distance $d_j$ of the parallel line to the center of rotation is higher than the distance $x'_i - x_c$ of the support line $i$, this intersection point $(x_j, z_j)$ has to be regarded as outside the slice area.

The desired distance $d_j$ is composed out of two parts (Fig. 11)

$$d_j = d_1 + d_2,$$

whereas

$$d_1 = (x_j - x_c) \cdot \cos(\varphi_i),$$

and

$$d_2 = z_j \cdot \sin(\varphi_i).$$

The rule for the exclusion of intersection points is as follows:

$$\text{if } d_j > x'_i - x_c, \text{ then ignore intersection point } (x_j, z_j).$$

Naturally, this includes both special cases discussed above, since $d_j = x_j - x_c$ for $\varphi_i = 0$ and $d_j = x_c - x_j$ for $\varphi_i = 180^\circ$, but also all cases with $180 < \varphi_i < 360^\circ$.

5 APPLICATION AND DISCUSSION

The method described here has been applied to the reconstruction of 3D shapes of loop objects recorded with optical cameras at different experimental X-ray stations for protein crystallography. Figs. 12 and 13 show pairwise the comparison between recorded 2D projections, from which the slice contours have been calculated, and the corresponding pictures containing the remaining intersection points (visualized using GnuPlot [30]). The shown pictures (756 x 508 pixels) were taken at the European Molecular Biology Laboratory (EMBL) in Hamburg, Germany. The loop objects consist of a nylon loop that is drilled at the loop stem, holding a tiny protein crystal in liquor. In total, 17 projections arbitrarily taken at rotational distances between 10 and 40 degrees have been used for the shape reconstruction of this example. The 2D processing leads to 527 slices. Once the 2D contour points have been determined, the 3D shape reconstruction based on determination and filtering of all intersection points takes only 49 milliseconds on an Intel Pentium @ 3.0 GHz. The number of projections was specified for the primary task of crystal centering [2]. The number should be as small as possible, since the process of rotation takes considerable time at most existing X-ray stations for crystallography.

The determined surface points allow three-dimensional modeling of the loop object. All coordinates were converted into the Virtual Reality Modeling Language (VRML) and were visualized using the Cortona3D viewer [31]. Fig. 14 shows the result after full-automated reconstruction without any interactive postprocessing.

The vertical texture reflects the slice-by-slice processing and is caused by the imperfections of the contour detection algorithm used, which was not optimized to produce continuous (smooth) contours. Please note that the segmentation algorithm determines each contour from a single projection, while the reconstruction of the outer boundary of an object slice must consider all projections. Therefore, the improvement of the contour detection would not completely avoid the problems arising from projection errors discussed in Section 4.2. Fig. 15 depicts the erroneous reconstruction without using the removal of false intersection points.

Fig. 16 shows a second example, where a relatively huge protein crystal is mounted on a LithoLoop. The 3D shape reconstruction is based on 36 views at a rotational distance of 10 degrees each. The processing of all intersection points (397 slices) takes 212 milliseconds. In this example, a higher number of projections (compared to the first example) were chosen. This allows the reconstruction of smoother contours along single slices. The number, however, should not set arbitrarily high, because smaller rotational steps can lead to higher degradations of the slice-boundary reconstruction as was addressed already in Section 4.2.1.

According to [10], for any direction of projection “a convexity of the contour corresponds to a convex patch of the surface and a concavity to a saddle-shaped patch” (compare Fig. 1). That is, in the vertical direction ($y$ axis) concave structures, which are part of a saddle-shaped surface, can also be reconstructed. Horizontal structures, however, must be convex to be visualized correctly. This drawback could be reduced by, for instance, introducing one or more axes of rotation, i.e., different angles for the slicing of the object enabling the reconstruction of any saddle-shaped part of the surface. In X-ray crystallography, this might be achieved using kappa goniometers for
rotation. Concave parts of the surface, however, remain hidden. Large distances between reconstructed surface points (intersection points of support lines) give an indication of possible concavities.

Further investigation could aim at deriving measures of the accuracy of the reconstruction method requiring test systems with well-defined physical dimensions. This will also give indication, whether orthographic projection yields sufficient accuracy or whether perspective projection has to be considered. The contour determination should be improved by exploiting the correlation between adjacent slices enabling the reconstruction of smoother surfaces. Estimating depth information based on matching corresponding surface points could enable the reconstruction of concave surface
regions. Appropriate illumination avoiding strong reflection and shadows would be an essential prerequisite.

Modeling crystals smaller than the loop based on occluding contours is very difficult or even impossible, because the crystal is not always as clearly visible as in the first presented example. Nevertheless, as long as the crystal is somewhat thicker than the loop, information of the crystal size can be concluded from the maximum horizontal width of the loop object, when inspecting the loop at proper perspective (Fig. 14 in the middle).

The technique presented here enables the rapid reconstruction of three-dimensional shapes based on silhouettes due to its low computational complexity and shows high robustness in the presence of noisy object contours.
APPENDIX

DERIVATION OF INTERSECTION POINTS

Based on the parameterized form of the straight-line equation

\[ d = x \cdot \cos(\varphi) + z \cdot \sin(\varphi), \]

the coordinates \((x_{ij}, z_{ij})\) of the intersection of two support lines, \(i\) and \(j\), can be computed as follows: let \(i = 1\) and \(j = 2\), then

\[
\begin{align*}
I : d_1 &= x \cdot \cos(\varphi_1) + z \cdot \sin(\varphi_1), \\
II : d_2 &= x \cdot \cos(\varphi_2) + z \cdot \sin(\varphi_2), \\
I : x_1 &= \frac{d_1 - z \cdot \sin(\varphi_1)}{\cos(\varphi_1)}, \\
II : x_2 &= \frac{d_2 - z \cdot \sin(\varphi_2)}{\cos(\varphi_2)}, \\
&= \frac{d_1 - z \cdot \sin(\varphi_1)}{\cos(\varphi_1)} = \frac{d_2 - z \cdot \sin(\varphi_2)}{\cos(\varphi_2)}, \\
[x_1 - z \cdot \sin(\varphi_1)] \cdot \cos(\varphi_2) &= [x_2 - z \cdot \sin(\varphi_2)] \cdot \cos(\varphi_1),
\end{align*}
\]

\[ d_1 \cdot \cos(\varphi_2) - z \cdot \sin(\varphi_1) \cdot \cos(\varphi_2) = d_2 \cdot \cos(\varphi_1) - z \cdot \sin(\varphi_2) \cdot \cos(\varphi_1), \]

\[ z \cdot \sin(\varphi_2) \cdot \cos(\varphi_1) - z \cdot \sin(\varphi_1) \cdot \cos(\varphi_2) = d_2 \cdot \cos(\varphi_1) - d_1 \cdot \cos(\varphi_2), \]

\[ z = \frac{d_2 \cdot \cos(\varphi_1) - d_1 \cdot \cos(\varphi_2)}{\sin(\varphi_2) \cdot \cos(\varphi_1) - \sin(\varphi_1) \cdot \cos(\varphi_2)}. \]

(If \(\varphi_i = 90\), then obviously \(d_i = z\); if \(\varphi_i = 270\), then \(z = -d_i\).)

The resulting coordinates of the intersection point are

\[
\begin{align*}
&z_{12} = \frac{d_2 \cdot \cos(\varphi_1) - d_1 \cdot \cos(\varphi_2)}{\sin(\varphi_2) \cdot \cos(\varphi_1) - \sin(\varphi_1) \cdot \cos(\varphi_2)}, \\
x_{12} = \begin{cases} \\
\frac{d_1 - z \cdot \sin(\varphi_1)}{\cos(\varphi_1)} + x_c, & \text{if } |\cos(\varphi_1)| > |\cos(\varphi_2)|, \\
\frac{d_2 - z \cdot \sin(\varphi_2)}{\cos(\varphi_2)} + x_c, & \text{otherwise.}
\end{cases}
\end{align*}
\]

Since the distances \(d_i\) are relative to the center of rotation, the \(x\) coordinate must be increased by \(x_c\).
ACKNOWLEDGMENTS

This research was mainly funded by the BIOXHIT project (EC FP6, contract number LSHG-CT-2003-503420) during the author's stay at the European Molecular Biology Laboratory (EMBL, Outstation Hamburg). The author thanks Victor Lamzin for general support, and Annette Faust for preparing and mounting the crystals as well as taking the photos presented in this paper (both from EMBL, Outstation Hamburg). The author also thanks the anonymous reviewers for their careful comments on an earlier version of this manuscript.

REFERENCES


Fig. 16. Example 2: (a) photos taken at projections angles of 70, 120, and 170 degrees, (b) three-dimensional model of the loop object based on all intersection points (total number of projections: 36), and (c) based on smallest polygons.
STRUTZ: 3D SHAPE RECONSTRUCTION OF LOOP OBJECTS IN X-RAY PROTEIN CRYSTALLOGRAPHY

Tilo Strutz received the Dipl.-Ing. degree in electrical engineering in 1994, the Dr.-Ing. degree in signal processing in 1997, and the Dr.-Ing. habil. degree in communications engineering in 2002 from the University of Rostock, Germany. He worked at the European Molecular Biology Laboratory (Outstation Hamburg) in the field of multidimensional signal processing and data analysis from 2003 to 2007. He is now a professor of information and coding theory at the Hochschule für Telekommunikation Leipzig (University of Applied Sciences). His research interests range from general signal processing to special problems of image processing, to data compression.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.