1999

# An algorithm for parallel 3D Reconstruction of Asymmetric objects from Electron Micrographs

Robert E. Lynch
*Purdue University*, rel@cs.purdue.edu

Hong Lin

Dan C. Marinescu

Report Number:
99-029

# AN ALGORITHM FOR PARALLEL 3D RECONSTRUCTION OF ASYMMETRIC OBJECTS FROM ELECTRON MICROGRAPHS

Robert E. Lynch
Hong Lin
Dan C. Marinescu

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

CSD TR #99-029
September 1999
(Revised February 2000)

# AN ALGORITHM FOR PARALLEL 3D RECONSTRUCTION OF ASYMMETRIC OBJECTS FROM ELECTRON MICROGRAPHS *

ROBERT E. LYNCH, HONG LIN, AND DAN C. MARINESCU
DEPARTMENT OF COMPUTER SCIENCES
PURDUE UNIVERSITY
WEST LAFAYETTE, IN 47907

**Abstract.** Recently we proposed a 3D reconstruction algorithm based upon Fourier analysis using Cartesian coordinates. In this algorithm the computations to determine the values of the 3D Discrete Fourier Transforms of the density of an asymmetric object could be naturally distributed over the nodes of a parallel system. Now we present an improvement of this algorithm which, for reconstruction at points of an $N \times N \times N$ grid, uses $O(N^3)$ arithmetic operations instead of $O(N^5)$. We discuss the accuracy of the algorithm and present comparisons of values computed by a parallel program implementing the algorithm with known solutions. Our results are consistent with the ones produced by a sequential program used for many years for structural biology studies. We report the speedup and the load balancing results for processing cryo-EM data for several viruses. We use a cluster of inexpensive PCs interconnected by a switched Ethernet to carry out the 3D reconstruction. One iteration of the 3D reconstruction for the Bursalia Corella Virus that used to take about 4 hours using a sequential program was carried out in less than 3 minutes on 16 nodes using the program based upon our algorithm. The algorithm is general and can be used for 3D reconstruction of asymmetric objects for applications other than structural biology.

**Key words.** 3D reconstruction, parallel processing, Fourier Transforms, projection method, piecewise constant approximation.

**AMS subject classifications.** 65C20, 65D99, 65T99, 65-05, 92B99

**1. Introduction.** There are several practical methods for reconstructing a 3D object from a set of its 2D projections. These include use of Fourier Transforms, 'back projection', and numerical inversion of the Radon Transform. See [Gor74] for a review of these and other methods. Also, for descriptions of (sequential) methods for 3D reconstruction and related tasks, see [Dea93], [Fra96], and [Gra96], three of several books containing clear explanations and many references.

In [Lyn99] an outline is given of our first parallel algorithm for 3D reconstruction which was based on a Fourier Transform method proposed in [Cro70]. Here we present an improvement which, for reconstruction at points of an $N \times N \times N$ grid, uses $O(N^3)$ arithmetic operations instead of $O(N^5)$ as in [Lyn99].

**2. Informal Description of the Algorithm.** The atomic structure determination of macromolecules based upon electron microscopy is an important application of 3D reconstruction. The procedure for structure determination consists of the following steps:

**Step a** Extract individual particle projections from micrographs and identify the center of each projection.

**Step b** Determine the orientation of each projection.

**Step c** Carry out the 3D reconstruction of the biological macromolecule.

**Step d** Dock an atomic model into the 3D density map.

The development of parallel algorithms to carry out some of these computations is part of an ambitious effort to design an environment for 'real-time electron microscopy', where results can be obtained in hours or days rather than in weeks or months.

Algorithms for Step a, which include automatic identification of particle projections, the determination of the center and orientation of each virus particle projection are discussed elsewhere [Mar97], parallel algorithms to determine the orientation, Step b are presented in [Bak97]. We use the terms "projections" and "views" throughout this paper with essentially the same meaning, the first seems more suitable for the description of the algorithm, we associate a direction with one projection, the second one seems more meaningful in the context of the experiment, we have many views of a virus particle.

In this paper we are only concerned with Step c of the process outlined above, the 3D reconstruc-

*The research reported in this paper was partially supported by a grant from National Science Foundation, MCB 9527131.

tion of asymmetric objects. This process is carried out as follows:

**Step 1** Compute the 2D Discrete Fourier Transform of each projection.

**Step 2** Compute the 3D Discrete Fourier Transform of the electron density by interpolation knowing the set of 2D projections and their orientations.

**Step 3** Compute the inverse 3D transform to get the electron density.

The 2D Discrete Fourier Transform (DFT) of each projection is computed (Step 1). With the orientation information, the points of intersection of the plane of projection and 3D grid lines can be determined and then estimates of the 3D Fourier Coefficients of the density at non-integral grid points are determined. In the method outlined in [Lyn99], this step took $O(N^5)$ arithmetic operations; in this paper we show how this can be reduced to $O(N^3)$. An inverse 3D DFT is carried out to obtain estimates of the density at grid points (Step 3).

Some of these computations can be done independently from each other. For example in Step 1 each processor can be assigned a set of projections and carry out the 2D DFT concurrently. Data exchange among nodes is necessary to collect information for Step 2 and then each node calculates the Fourier Coefficients on its own set of 3D planes. Different portions of the 3D DFT of the electron density are stored on different nodes; where possible, we carry out 2D inverse transforms and then data are exchanged among nodes and so that the final set of 1D inverse transforms takes place to complete Step 3. Theoretical basis for this 3D reconstruction is given in Section 3.

To use efficiently a parallel computer or a cluster of workstations, we need parallel algorithms that partition the data and computations evenly among nodes to ensure load balance and, moreover, which minimize the communication among processors by maintaining a high level of locality of reference. Similar efforts have been reported in the past [Joh94], but the performance data available to us suggest that new algorithms have to be designed to reduce dramatically the computation time.

We have implemented a Data Parallel algorithm, the computation consists of several phases and at the end of some of these phases, a global exchange of data occurs. To minimize synchronization delays the load should be balanced among processors for every execution phase.

The algorithm for 3D reconstruction was designed to work well on workstation clusters. The program uses MPI (Message-Passing Interface) and consists of several phases including initialization, 2D Fourier analysis, estimating the 3D coefficients of the DFT, solving linear systems, and Fourier synthesis. In the first phase of the algorithm, pixel frames are distributed evenly among nodes and processed independently. The values of the 3D DFT are calculated by interpolation and collected for all three directions in a node. The systems are distributed among nodes so that a 2D Fourier synthesis for a plane can be done without an exchange. Then data must then be exchanged so that 1D syntheses can be done to obtain the electron density.

A more detailed partitioning of computational phases is presented in Section 9 where experimental results are given.

**3. 3D Reconstruction by Fourier Transforms.** In this section we summarize the Fourier Transform method (see [Cro70]), the basis of our original algorithm.

The experimental information is gathered from digitized images of many identical macromolecules obtained with a cryo electron microscope. Individual particle projections are identified and the orientation of each 2D image, with respect to a standard XYZ Cartesian System, and the location on the image of the projection of the centroid of the molecule's electron density are determined. See [Lyn99] for an outline and see [Bak97] and [Mar97] for more details.

Each pixel frame is a square $M \times M$ array of pixel values $p$ which are taken as the values of the projection of the electron density, $\rho$, normal to the $(r,s)$-plane of the pixel frame:

$$p_t(r,s) = \int_t \rho_t(r,s,t)\,dt, \tag{1}$$

where the subscript t indicates the unit vector normal to the $(r,s)$-plane of projection.

The electron density of the macromolecule is to be estimated at grid points of a 3D Cartesian coordinate system, we call the 'XYZ System'; the centroid of the density is at its origin. The Fourier

Transform of the electron density is taken as

$$F(\mathbf{h}) = \frac{1}{A^3}\int \rho(\mathbf{x})\, e^{-2\pi i\, \mathbf{h}^T\mathbf{x}/A}\, d\mathbf{x} \tag{2}$$

where $^T$ denotes transpose, $\mathbf{x} = (x, y, z)^T$, $\mathbf{h} = (h, k, \ell)^T$, and $A$ denotes the edge-length of a cube containing the electron density. As is customary in crystallography and electron microscopy, we call $\mathbf{x}$ a point in *real space* and $\mathbf{h}$ a point in *reciprocal space*.

*Projection Theorem.* We sketch a proof of the projection theorem which states that the 3D Fourier Transform of $\rho$ at points on a plane through the origin is equal to the 2D Fourier Transform of the projection of $\rho$ onto that same plane; (also see [Cro70], [Dea93], etc.).

A rotation about the origin of the XYZ System to an RST Cartesian System is accomplished by the transformation $\mathbf{r} = E\mathbf{x}$, where $E$ is a $3 \times 3$ orthogonal matrix; that is, $E$ is real, invertible, and its inverse, $E^{-1}$, is equal to its transpose, $E^T$. It is easy to verify that lengths are invariant: $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T\mathbf{x}} = \sqrt{\mathbf{r}^T\mathbf{r}}$; e.g., $\mathbf{x}^T\mathbf{x} = \mathbf{x}^T(E^TE)\mathbf{x} = (E\mathbf{x})^T(E\mathbf{x}) = \mathbf{r}^T\mathbf{r}$. Similarly, angles are invariant: if $\mathbf{u} = E\mathbf{h}$, then $\mathbf{h}^T\mathbf{x} = \mathbf{u}^T\mathbf{r} = \|\mathbf{x}\|\,\|\mathbf{h}\|\cos\theta$. Thus, $E$ defines a rotation of the XYZ System about its origin. Rotations of macromolecules are 'proper' so that the determinant of $E$ is equal to $+1$ and thus $d\mathbf{x} = d\mathbf{r}$ ('improper' rotations have $\det(E) = -1$). Set $\rho(\mathbf{x}) = \hat{\rho}(\mathbf{r}) = \rho(E^T\mathbf{r})$, then, by direct substitution into (2),

$$F(\mathbf{h}) = \frac{1}{A^3}\int \hat{\rho}(\mathbf{r})\, e^{-2\pi i\, \mathbf{u}^T\mathbf{r}/A}\, d\mathbf{r} = \hat{F}(\mathbf{u}) \tag{3}$$

In particular, if $\mathbf{u} = (u, v, w)^T$ with $w = 0$ and $(u, v, 0)^T = E\mathbf{h}'$, then

$$F(\mathbf{h}') = \frac{1}{A^2}\int\int \left\{\frac{1}{A}\int \hat{\rho}(r, s, t)\, dt\right\} e^{-2\pi i(ur+vs)/A}\, dr\, ds = \hat{F}(u, v, 0) = P(u, v) \tag{4}$$

This shows that the 2D Fourier Transform, $P$, of the 2D projection of the density onto the plane through the origin with normal parallel to the $t$-axis is equal to the 3D transform, $F$, at points $\mathbf{h}'$ on the reciprocal space image of the plane of projection.

*Application of the Projection Theorem.* The macromolecule has finite volume, it is sufficient to use the limits $-A/2$ to $A/2$ for each of the three integrations in (2). Then when $\mathbf{h} = (h, k, \ell)$ has integer components, $F(\mathbf{h})$ is a Fourier coefficient of the periodic extension of $\rho$, with period $A$ in each coordinate direction. Thus its density can be represented as a 3D Fourier Series:

$$\rho(\mathbf{x}) = \sum_{\mathbf{h}} F(\mathbf{h}) e^{2\pi i \mathbf{h}^T\mathbf{x}/A} \tag{5}$$

where $\mathbf{h}$ has integer components, $h$, $k$, and $\ell$. Since $ur + vs$ in (4) is equal to $\mathbf{u}^T\mathbf{r} = (\mathbf{h}')^T\mathbf{x}$, use of the series representation of $\rho$ yields

$$F(\mathbf{h}') = \frac{1}{A^3}\int \left\{\sum_{\mathbf{h}} F(\mathbf{h}) e^{2\pi i \mathbf{h}^T\mathbf{x}/A}\right\} e^{-2\pi i\, \mathbf{h}'\mathbf{x}/A}\, d\mathbf{x} = \sum_{\mathbf{h}} F(\mathbf{h}) \frac{1}{A^3}\int e^{2\pi i(\mathbf{h}-\mathbf{h}')^T\mathbf{x}/A}\, d\mathbf{x} \tag{6}$$

where the components of $\mathbf{h}'$ are, typically, not integers. Since

$$\frac{1}{A}\int_{-A/2}^{A/2} e^{2\pi i\, \ell\, t/A}\, dt = \mathrm{sinc}(\ell) = \left\{ \begin{array}{ll} \sin(\pi\ell)/\pi\ell & \text{if } \ell \neq 0 \\ 1 & \text{if } \ell = 0 \end{array} \right\}, \tag{7}$$

we have, as in [Cro70],

$$F(\mathbf{h}') = P(u, v) = \sum_{h,k,\ell} F(h, k, \ell)\, \mathrm{sinc}(h - h')\, \mathrm{sinc}(k - k')\, \mathrm{sinc}(\ell - \ell'). \tag{8}$$

*Evaluation of $F(h, k, \ell)$.* Similar to (5), the projection, $p$, of the electron density onto the plane of the pixel frame can be represented as a 2D Fourier Series with period $B$. The series and its

coefficients, $P$, are

$$(9)\quad p(r,s)=\sum_{u,v=-\infty}^{\infty} P(u,v)\,e^{2\pi i(ur+vs)/B},\quad P(u,v)=\frac{1}{B^2}\int_{-B/2}^{B/2} p(r,s)\,e^{-2\pi i(ur+vs)/B}\,dr\,ds,$$

and $P(u,v)$, with $u$, $v$ integers, can be used in (8).

One limits each of the indices to a finite number, say $N$; then there are $N^3$ unknown values $F(h,k,\ell)$. There are $M^2$ known values of $P$ for each projection and suppose there are $\nu M$ projections, with $\nu > (N/M)^3$. For $N = M$, [Ros98] gives a lower bound on $\nu$ such that the number of experimental data is equal to the number of unknowns.

One can solve the resulting linear least squares problem using standard mathematical software, such as the LAPACK routine SGELSS [And92], which uses the singular value decomposition. The order of magnitude of the number of arithmetic operations required to solve this system is the product of the number of equations and the square of the number of unknowns (see [Gol96], p. 263), so when $M = N$, $O(N^9)$ arithmetic operations are required.

*Reduction in Arithmetic.* In (8), $u$, $v$, $h$, $k$, and $\ell$, are integers and $P$ is evaluated at grid points of the transformed pixel frame. [Cro70] points out that the computation can be significantly reduced if $h'$, $k'$ in (8) are the integers $h$, $k$. But then the point at which $P$ is evaluated is no longer a grid point. Instead, (8) reduces to

$$(10)\qquad P(u',v')=\sum_{\ell} F(h,k,\ell)\,\mathrm{sinc}(\ell-\ell')$$

where $u'$ and $v'$ are not integers. Because $P$ is not evaluated at a grid point, some kind of interpolation must be used to estimate $P(u',v')$ in terms of values of $P(u,v)$ at grid points.

In (10), $h$ and $k$ are fixed and one has a single system of equations with $N$ unknowns; for $\nu N$ pixel frames, and the complexity of the algorithm to solve the system is $O(N^3)$. There are $N^2$ such systems, one for each integer pair $(h,k)$. So there are $O(N^5)$ operations required to estimate all the coefficients $F(h,k,\ell)$, instead of the $O(N^9)$ operations needed to solve (8).

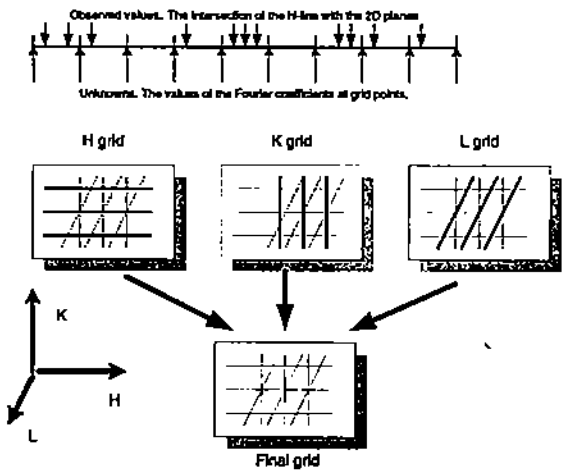


Figure 1. One-dimensional interpolation on a line parallel to the axes of the 3D grid in the Fourier domain. The $N$ values of the function at grid points (arrows pointing upwards) are unknown, the ones corresponding to arrows pointing downwards are the $I$ known/observed values. We can determine by interpolation the values at grid points for all H-lines and construct the H-grid. Similarly we obtain the K-grid and the L-grid. Then we can average the three independent values of the function, for every grid point and obtain a final grid in the Fourier domain.

The process can be repeated for each integer pair $(h,l)$ and $(k,l)$ as indicated in Figure 1. As a result we obtain three estimates for the value of each coefficient $F(h,k,\ell)$ and by averaging the

three estimates the accuracy of the algorithm can be improved. The three sets of values of the coefficients $F(h,k,\ell)$ corresponding to the pairs $(h,k)$, $(h,l)$ and $(k,l)$ are called the $L, K$ and $H$ grid respectively in Figure 1.

For each grid, the $N^2$ systems are independent from each other and they can be solved concurrently. This is the basis for the first method we described in [Lyn99].

**4. Projections onto Slabs.** We carried out numerical experiments with our program when (10) was reduced to a diagonal system. The quantity $F(h,k,\ell)\,\mathrm{sinc}(\ell-\ell')$ was included on the right side only if $|\ell-\ell'|<1/2$. For each $\ell'_j$ and the corresponding $P(u'_j,v'_j)$, (10) reduces to

$$P(u'_j,v'_j)=F(h,k,\ell)\,\mathrm{sinc}(\ell-\ell'_j)+G(h,k,\ell'_j) \tag{11}$$

such that all the other $F$'s are discarded and $G(h,k,\ell'_j)$ is the error. The least square solution of the set of equations (11) chooses $F(h,k,\ell)$ to minimize $E(h,k,\ell)=\sum_j G(h,k,\ell'_j)^2$ and since

$$\begin{aligned} dE(h,k,\ell)/dF(h,k,l) &= d\left[\textstyle\sum_j \left(P(u'_j,v'_j)-F(h,k,\ell)\,\mathrm{sinc}(\ell-\ell'_j)\right)^2\right]/dF(h,k,l) \\ &= 2\textstyle\sum_j \left\{\mathrm{sinc}(\ell-\ell'_j)\left[P(u'_j,v'_j)-F(h,k,\ell)\,\mathrm{sinc}(\ell-\ell'_j)\right]\right\} \end{aligned}$$

the least square error is minimized

$$F(h,k,\ell)=\left\{\sum_j \mathrm{sinc}(\ell-\ell'_j)P(u'_j,v'_j)\right\}\Big/\sum_j \mathrm{sinc}(\ell-\ell'_j)^2 \tag{12}$$

Solving (12) instead (10), reduces the amount of arithmetic by $O(N^2)$.

In this method, values of $P$ are given large weights when $\ell'$ is very close to $\ell$ because $\mathrm{sinc}(x)$ is an oscillating function decaying at the rate of $1/x$ and has its global maximum at $x=0$.

The resulting electron density values at grid points obtained by a Fourier Synthesis were remarkably close to those obtained, with the same input, by the method described in [Bak88]; the program described there has been producing electron density values for several years which have been accepted as accurate. Based on these experimental results, we constructed a model of the approximations which leads to equations like (10), but having only one term on the right side.

*Piecewise Constant Model.* Variation of the value inside a pixel square cannot be measured, and thus we take $p$ to be a piecewise constant function on the pixel frame:

$$p_t(r,s)=p_t(i\Delta r,j\Delta s)\quad \text{for } |r-i\Delta r|<\Delta r/2,\quad |s-j\Delta s|<\Delta s/2,\ \text{with } \Delta r=\Delta s, \tag{13}$$

where $i$ and $j$ are integers, the subscript t denotes a unit vector normal to the plane of the pixel frame, and $(i\Delta r, j\Delta s)$ denotes grid points at the center of pixel squares.

If we regard this function to be defined on a *plane*, then we are lead to the system in (10) because, except at the origin, it is unlikely that any of the grid points of the XYZ system would be in the plane of the projection of a randomly oriented molecule and similarly for its transforms in reciprocal space.

We now regard the projection (1) as a function defined on a *slab* of thickness $\Delta r$, rather than on a *plane*. This not only gives a formulation which is consistent in the three coordinate directions, but also leads to an algorithm using fewer arithmetic operations, namely, $O(N^3)$ instead of $O(N^5)$, and produces accurate results.

We take the slab to have thickness $\Delta r$, use $t$ to denote a point on the axis normal to the projection, and for each point $(r,s)$, we take the value of the projection to be independent of $t$ so that the left side of (1) and (13) becomes

$$p_t(r,s,t)=p_t(r,s,0),\qquad |t|<\Delta r/2. \tag{14}$$

This function is piecewise constant (because of (13) and (14)) in the three coordinate directions.

**5. Grid Points in Slabs.** Recall that the HKL System is the reciprocal space of the XYZ System of the macromolecule. The UV System is the reciprocal space of the pixel frame and the W-axis is orthogonal to the UV-plane. In Appendix A2 the 3-by-3 matrix $G = \sigma E$ is constructed to relate points in the UVW System and the HKL System:

$$(u, v, w)^T = G(h, k, \ell)^T \tag{15}$$

We are only concerned with points $(u, v, w)^T$ in the reciprocal space of the slab and points $(h, k, \ell)^T$ in the reciprocal space of the molecule that satisfy

$$|u|, |v| \leq M/2, \quad |w| \leq 1/2, \qquad |h|, |k|, |\ell| \leq N/2. \tag{16}$$

We call the sets of such points satisfying (15) and (16) the $\mathcal{HKL}$ and $\mathcal{UVW}$ systems, respectively. We want to find sets of integers, $h, k, \ell$, in $\mathcal{HKL}$ and the corresponding $u, v, w$ in $\mathcal{UVW}$.

Let $G_{i,j}$ denote the entry that is in the $i$-th row and the $j$-th column of $G$. Then the equation in (15) for $w$ is

$$w = G_{3,1}h + G_{3,2}k + G_{3,3}\ell. \tag{17}$$

We choose a pair of integers, $h$, $k$, which satisfy (16), and because of the restrictions on $w$ in (16), we need $\ell$ which satisfies

$$-1/2 - G_{3,1}h - G_{3,2}k \leq G_{3,3}\ell \leq 1/2 - G_{3,1}h - G_{3,2}k. \tag{18}$$

If this condition cannot be satisfied for any integer $\ell$ with $|\ell| \leq N/2$, then we reject the pair of integers $h, k$ and choose another pair.

Suppose that the pair of integers $h, k$ are such that (18) is satisfied for integers $\ell_i$ with $-N/2 \leq \ell_1 < \ldots < \ell_j \leq N/2$. Then for such an integer, $\ell_i$, we have

$$u_i = G_{1,1}h + G_{1,2}k + G_{1,3}\ell_i. \qquad v_i = G_{2,1}h + G_{2,2}k + G_{2,3}\ell_i. \tag{19}$$

If $|u_i| > M/2$ or $|v_i| > M/2$, then we reject the integers $h$, $k$, and $\ell_i$ and try another value of $\ell$.

When we find integers $h$, $k$, $\ell_i$ and corresponding $u_i$, $v_i$, such that (16) is satisfied, then we have found a grid point in the $\mathcal{HKL}$ System which corresponds to a point $(u_i, v_i, w_i)^T$ in the $\mathcal{UVW}$ System. In our model, which uses piecewise constant functions, we set the value of $P(u, v, 0)$ equal to $p_t(u, v, t)$, where $u$ and $v$ are the integers nearest $u_i$ and $v_i$, respectively.

Because $h$, $k$, $\ell_i$ are integers, the equation (10) with $N$ terms on the right side reduces to

$$F(h, k, \ell_i) = p_t(u, v, t), \tag{20}$$

Here the right side is a known value of the transform of the pixel values.

As described above, one exams each of the $N^2$ integer pairs $h, k$; however this $O(N^2)$ work can be reduced to $O(N)$ because we know, by use of simple linear algebra, that acceptable pairs lie in a specific slab of width $\Delta w$ containing the origin. Finding the set of acceptable integer pairs reduces to an integer programming problem but one for which we have very specific information about the location of its set of solutions.

**6. Estimate of the Electron Density.** Because we are dealing with finite sets of points, Discrete Fourier Transforms (DFT) are used instead of Fourier Transforms.

Each pixel frame produces estimates of the $F(h, k, \ell)$ for a set of grid points, $(h, k, \ell)$, near a plane in the $\mathcal{HKL}$ System. These are obtained from (20) in terms of values of the DFT of the of the pixel frame. For a fixed grid point, $(h, k, \ell)$, suppose that there are estimates of $F(h, k, \ell)$ obtained from $S$ different pixel frames; denote these values by $F_s(h, k, \ell)$, $s = 1, \ldots, S$. The final estimate of $F(h, k, \ell)$ is obtained by averaging:

$$F(h, k, \ell) = \frac{1}{S} \sum_s F_s(h, k, \ell) \tag{21}$$

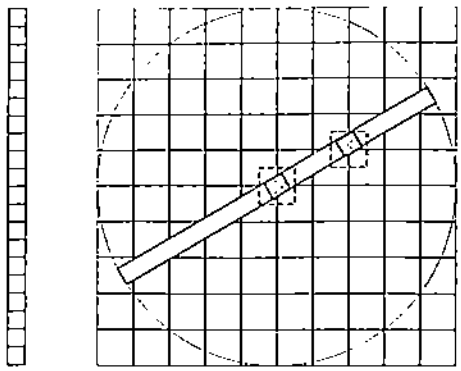

Figure 2. Slab projection simplified to the 2D case. A 1D 'pixel frame' is indicated on the left. The figure on the right shows this frame in its correct orientation on the 2D domain of the Discrete Fourier Transform. When a 2D grid point is in the pixel frame, as indicated by the shaded square, that value from the pixel frame is taken as $F$ at the grid point.

If there are no estimates of $F(h,k,\ell)$, then $F(h,k,\ell)$ is set equal to zero.

We accept these as the DFT of the the electron density and invert the transform to obtain the estimate of the electron density.

**7. The Effect of Zero-Fill.** We can put an $M_p \times M_p$ array of pixel values into a larger $M \times M$ array; the extra array entries are set equal to zero. We call this "zero-fill" and the amount of fill is called the "aspect ratio $k$", defined by

$$\text{zero-fill aspect ratio } \; k = M/M_p$$

After the zero-fill, 2D DFT of the larger array is determined. Here we show the relationship between the grid spacings of the DFT of the pixel values and the spacing when the domain of the DFT is expanded to a larger domain with zero-fill (see [Bri95], p. 90 ff).

To be able to display informative figures, we simplify the discussion to the case of a 1D set of pixel values at $M_p$ grid points and a 2D density. As we now show, zero-fill results in interpolation (and scaling) of the Discrete Fourier Coefficients in reciprocal space; although the domain in real space increases by the zero-fill factor, the spacing in reciprocal space is reduced by this factor, so the length of the domain is not changed. An example with $k = 2$ is shown in Figure 3.

The spacing in real space is $\Delta 4 = A/M_p$. For even $M_p$, the DFT of $f$ can be written as

$$\hat{f}_u = \frac{1}{\sqrt{M_p}} \sum_{p=-M_p/2+1}^{M_p/2} f_p \exp(-2\pi i\, up/M_p), \tag{22}$$

and its inverse as

$$f_p = \frac{1}{\sqrt{M_p}} \sum_{p=-M_p/2+1}^{M_p/2} \hat{f}_u \exp(2\pi i\, up/M_p). \tag{23}$$

The transformed values are complex. Because $f$ is real valued, the complex valued transformed function, $\hat{f}$, is conjugate symmetric so that its real part is an even function and its imaginary part is an odd function. The real and the imaginary parts of the transform $\hat{f}$ are shown in Figure 3(1b) as piecewise linear interpolants to the values.

The arguments of the exponential can be written in terms of the spacings $\Delta r$ and $\Delta u$ of real space and reciprocal space, respectively, as

$$2\pi i\, up/M_p = 2\pi i\, (u/A)(pA/M_p) = 2\pi i\, (u\Delta u)(p\Delta r)$$

so that the spacings in real and in reciprocal space are

$$\Delta r = A/M_p \qquad \text{and} \qquad \Delta u = 1/A,$$

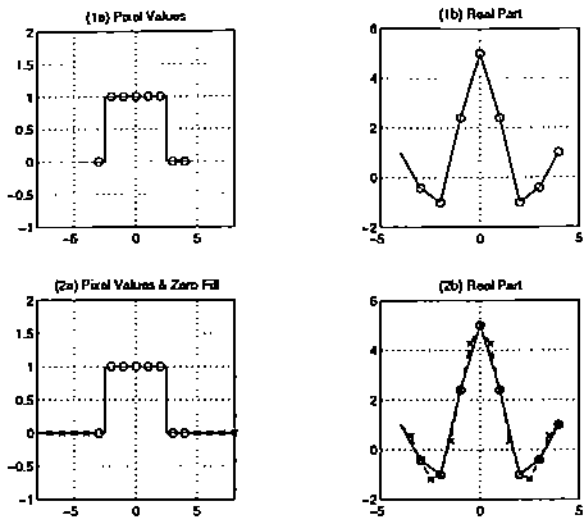


Figure 3. The effect of zero-fill. (1a): An 8-point step function; the circles represent the density inside and outside a uniform sphere. (1b): The real part of the 8-point DFT of (1a); the imaginary part is zero because the step function is symmetric. (2a): The function (2a) with zero-fill factor $k = 2$; (1a) is augmented by 8 zero values, indicated by crosses; the spacing is the same as in (1a). (2b): The real part of the 16-point DFT of (2c) multiplied by *sqrt*2. The spacing in (2b) is half that in (1b). The values, indicated by circles, at $0, \pm 1, \pm 2, \pm 3, 4$ in (2b) are the same as those in (1b); the 8 values indicate by $\times$'s are due to the zero-fill. This shows that zero-fill in real space results in *interpolation* (after scaling) in reciprocal space.

respectively. The units of $\Delta r$ are Ångstroms and those of $\Delta u$ are reciprocal Ångstroms.

In both spaces, there are $M_p$ points in a fundamental period. The length of the fundamental domain of $f$ in real space is $M_p \Delta r = A$, and the length of the fundamental domain of $\hat{f}$ in reciprocal space is $M_p \Delta u = M_p / A = \Omega$.

Next consider extending the domain of $f$ from $M_p$ grid points to $2M_p$ points with zero-fill (see Figure 3 for an example with $M_p = 8$). The transform is given by

$$F_u = \frac{1}{\sqrt{2M_p}} \sum_{p=-M_p+1}^{M_p} f_p \exp(-2\pi i\, up/2M_p).$$

Because of the zero-fill, $f_p = 0$ for $p = \pm M/2$, $\pm(M/2+1)$, ..., and we have

$$F_u = \frac{1}{\sqrt{2M_p}} \sum_{p=-M_p/2+1}^{M_p/2} f_p \exp(-2\pi i\, up/2M_p). \tag{24}$$

The argument of the exponential can be written in terms of $\Delta r$, $\Delta u$, and $\Delta U = \Delta u/2$ as

$$-2\pi i\, up/2M_p = -2\pi i\,(u/2A)(pA/M_p) = -2\pi i\,(u\Delta u/2)(p\Delta r) = -2\pi i\,(u\Delta U)(p\Delta r)$$

The grid spacing $\Delta r$ in real space does not change length when the length $A$ of the real domain is doubled to $2A$ and $f$ is set equal to zero in the extended domain. But, the grid spacing $\Delta U$ in reciprocal space, after the zero-fill, is half that of the spacing before the zero-fill. Consequently, the length of the fundamental domain in reciprocal space after the zero-fill is the *same* as without the zero-fill; specifically:

$$\Delta U = \Delta u/2, \qquad 2M_p \Delta U = 2M_p \Delta u/2 = M_p \Delta u = \Omega,$$

where $\Omega$ is the length of the original reciprocal space fundamental domain as well as the fundamental domain after zero-fill.

Comparing (22) and (24), one sees that $\hat{f}_u = \sqrt{2}F_{2u}$, $u = 0, \pm 1, \pm 2, \cdots$. Hence, $\sqrt{2}F$ *interpolates* to $\hat{f}$ at these points (see Figure 3).

In this example, we doubled the number of grid points and set $f$ equal to zero at the points outside the fundamental period. It is easy to verify that if we had used $M$ points, with $M > M_p$, and set $f$ equal to zero at the $M - M_p$ additional points, and kept the spacing $\Delta r$ in real space constant, then there is no change in the length of the domain in reciprocal space — it is $\Omega$ with or without zero-fill. The new grid length, $\Delta U$, in reciprocal space is a fraction of $\Delta u$, the original reciprocal grid length:

$$\Delta U = \frac{M_p}{M}\Delta u, \qquad M\Delta U = M\frac{M_p}{M}\Delta u = M_p\Delta u = \Omega.$$

Similar to the example above, which considered $M = 2M_p$, here a value $F_v$ interpolates to $\hat{f}_u$ at any point $v\Delta U$ which is a grid point $u\Delta u$ in the original domain.

Note that zero-fill does *not* extend the domain in *reciprocal* space — one does *not* obtain estimates of Fourier Transform values having longer wavelengths but, one does obtain estimates on a *finer* grid in reciprocal space.

These results extend to multi-dimensional transforms. Thus, when the $M_p \times M_p$ domain in real space of the 2D pixel frame is enlarged to $M \times M$ with zero-fill, then the grid spacing is reduced by the factor $M_p/M$. But the length of the sides of the domain in in *reciprocal space* is the *same* before as *after* the zero-fill.

*Use of subsets of the computed $F$'s* The use of zero fill increases the number of approximate Fourier coefficients (see Figure 3). The pixel frames are $P \times P$ and the arrays are $kP \times kP$. In order to reconstruct a density with $P \times P \times P$ (or fewer) density density values, one must select a portion of the $kP \times kP$ values. Figure 4 shows the effect of using values only close to the origin — clearly such reconstruction gives inaccurate representation. To obtain a reason reconstruction, one must use value extended across the whole domain. For example in the case illustrated in Figure 3), one could use values at x = -2, 0, 2, and 4, but certainly not at x = -1,-1/2,0,1/2, and 1 (see Figure 3).

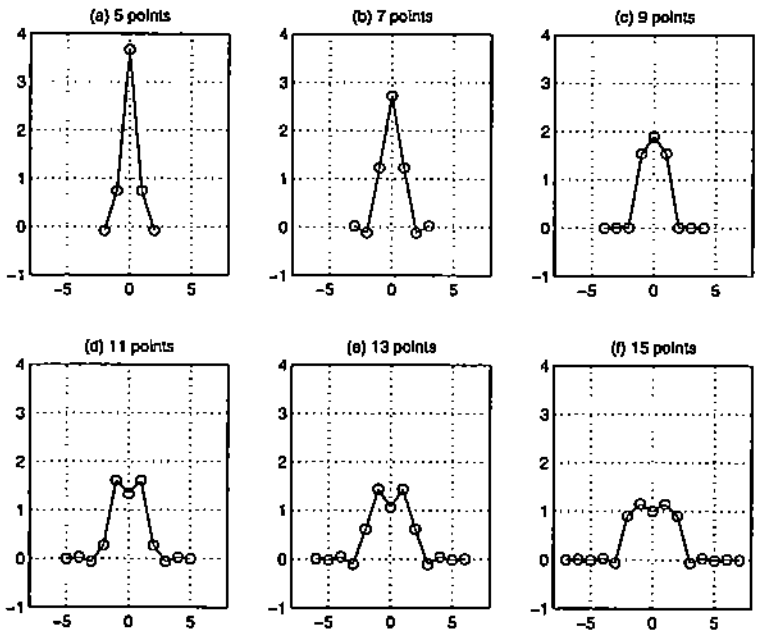


Figure 4. Use of subsets of $F$. (a),(b),...(f) show the 'reconstruction' obtained by synthesis of subsets of $F$. In each case we use a subset consisting of $F(0)$ and its nearest neighbors. This leads to inaccurate values of the density.

**8. Numerical Errors in 3D Reconstruction.** 3D reconstruction is subject to experimental as well as numerical errors. In this section we address only the problem of numerical errors. To deter-

| | |
|---|---|
| $P$ | The number of pixels on the edge of a $P \times P$ pixel frame |
| $D$ | The diameter of the sphere in pixel edges |
| $V$ | The number of pixel frames or 'views' |
| $k$ | Zero-fill aspect ratio; the pixel frame is put into a $kP \times kP$ array |

Table 1. The parameters for the test cases.

mine the accuracy of the reconstruction method we take a known object and generate projections for randomly selected orientations. Then we reconstruct the object from these projections and compare the original values with the computed ones. Our experiments are conducted using a uniform sphere with a constant density inside, equal to one, and zero outside, see Figure 5.

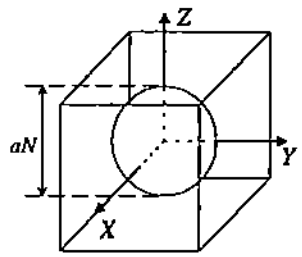


Figure 5. The original object: a uniform sphere with constant density inside and zero outside. The center of the sphere with diameter $aN$ with $a < 1$ coincides with the center of the cube with edge $N$.

We used the *piecewise constant* model to compute estimates of the values of 3D DFT coefficients. The parameters for a test case are: the size of a pixel frame, the diameter of the sphere, the number of pixel frames, the aspect ratio for zero-fill, and the size of 3D DFT. The size of output grid is determined by the size of pixel frame, it is $P \times P \times P$ for $P \times P$ frames. The parameters for a test care are listed in Table 1.

We calculate the estimate of the maximum and minimum pointwise error and also the mean square error as a function of radius to determine the effect of the zero-fill and the number of views. For the mean square error, we average of the square root of the sum of the squares of the errors in a set of annular regions inside the uniform sphere; we do not include grid points at the edge of the sphere where the jump discontinuity occurs.

We also report some of these same values as obtained with a sequential program widely used by the structural biology community, EM3DR.

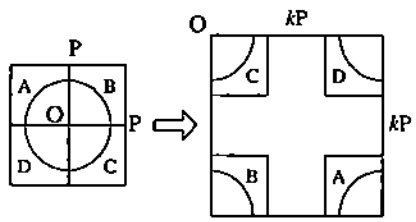


Figure 6. The transformation of a $P \times P$ pixel frame containing one projection consists of a translation of the origin and zero-fill. The frame is cut into 4 segments and rearranged in a larger frame of size of $kP \times kP$. $k$ is the aspect ratio of zero-fill (k = 1.5, in this example), and the centroid of the projection (point $O$) is moved to the upper left corner.

*The effect of zero-fill.* Before 2D DFT is carried out on pixel frames, the pixel values in the frame are rearranged and put into a larger frame with the extra array entries set equal to zero. Figure 6 illustrates the process of zero-fill. Errors as a function of the zero-fill aspect ratio $k$ are listed in Tables 2 and 3. The mean square errors listed in Table 2 decrease from $20-25\%$ for $k = 1$ to about 2% for $k = 4$. For a fixed $k$, the variation in the error is small as the size of the pixel frame changes with the ratio (diameter/[frame edge]) being kept nearly constant (about 0.79).

| Aspect ratio $k$ | Case 1<br>FrameSize 41 × 41<br>Diameter 32<br>NumberViews 20100 | Case 2<br>FrameSize 61 × 61<br>Diameter 48<br>NumberViews 45150 | Case 3<br>FrameSize 81 × 81<br>Diameter 64<br>NumberViews 80802 |
|---|---|---|---|
| 1 | 20.16 | 23.38 | 24.41 |
| 2 | 6.92 | 7.97 | 7.77 |
| 4 | 1.63 | 2.11 | |

Table 2. Mean square error (%) inside reconstructed uniform sphere. "FrameSize" denotes the number of pixels on each size of a pixel frame. "Diameter": denotes the diameter of the sphere in pixels. "NumberViews" denotes the number of projections used for reconstruction.

| Aspect ratio $k$ | Case 1<br>FrameSize 41 × 41<br>Diameter 32<br>NumberViews 20100 | Case 2<br>FrameSize 61 × 61<br>Diameter 48<br>NumberViews 45150 | Case 3<br>FrameSize 81 × 81<br>Diameter 64<br>NumberViews 80802 |
|---|---|---|---|
| 1 | 0.57/1.00 | 0.50/1.01 | 0.48/1.00 |
| 2 | 0.86/1.00 | 0.84/1.01 | 0.84/1.01 |
| 4 | 0.97/1.00 | 0.96/1.01 | |

Table 3. The effect of the zero-fill aspect ratio upon the minimum/maximum density values inside the sphere. *FrameSize* is size of the pixel frames in number of pixels and Diameter is the diameter of the sphere and em NumberViews is of projections used for reconstruction. The true values of the electron density are 1 inside the sphere and 0 outside.

The values in Table 3 list the minimum and maximum of the density inside the sphere. The variation of the density becomes smaller when the aspect ratio increases.

*The number of views.* The effect of the number of views on the percent errors and on the minimum and maximum errors are summarized in Tables 4 and 5 for the aspect ratio $k = 4$. For fixed pixel size and diameter, the variation in the percent error varies very little as the number of views increases. Similarly, there is little change in the range of computed values inside the sphere.

We expect that increasing the aspect ratio of the zero-fill allows us to use fewer projections for the 3D reconstruction. Table 6 and Figure 7 show the results when the size of pixel frames is $41 \times 41$ and the number of pixel frames varies from 25 to 200 for zero-fill aspect ratio is 1 and 4. For both zero-fill aspect ratios, the results converge when the number of pixel frames goes beyond a certain value (approximately 75 for this case). However, when zero-fill aspect ratio is 4, with the same number of pixel frames, the results get close to the correct values.

*Distribution of the errors as a function of radius.* Figure 8 provides conclusive evidence that the magnitude of errors increases with the radius. This increase is quite obvious in Figures 8.1b, 8.2b, 8.3b where we display the density function of the radius for $k = 1$. The oscillation of the graph is quite obvious in this case. For $k = 4$ figures 8 indicate again that the results are improved only slightly by increasing the number of views.

*The effect of noise.* Random numbers uniformly distributed between -0.1 and +0.1 were added to the constant value, $\rho = 1$, of the density inside the sphere to simulate the effect of the noise on the reconstruction process. The results shown in Figure 10 indicate that the noise leads to noticeable distortion of the sphere. The reconstruction algorithm behaves reasonably well, compare for example Figure 10(1a) with Figure 7(1f) and Figure 10(2a) with Figure 7(2f). The results support our expectation that the reconstruction is more accurate when we use a larger number of views and a larger aspect ratio. For the same number of views the effect of increasing aspect ratio is noticeable. Increasing the number of views has a more pronounced effect for the noisy data than for the noiseless case.

*Comparison of results of parallel 3D reconstruction program with the results produced by the sequential program* EM3DR. Table 7 lists the mean difference ($\overline{e}$), the variance ($\sigma$) and coefficient of variation ($\delta$) of the difference between the density computed by the parallel program based the algorithm described in this paper with the density computed by the sequential program, EM3DR.

| NumberViews | Case 1<br>FrameSize 21 × 21<br>Diameter 16<br>Aspect ratio 4 | Case 2<br>FrameSize 41 × 41<br>Diameter 32<br>Aspect ratio 4 | Case 3<br>FrameSize 61 × 61<br>Diameter 48<br>Aspect ratio 4 |
|---|---|---|---|
| 300 | 1.68 | 2.01 | 2.21 |
| 1250 | 1.41 | 1.77 | 1.88 |
| 2775 | 1.39 | 1.78 | 1.90 |
| 5000 | 1.34 | 1.72 | 1.88 |
| 11250 | | 1.67 | 1.89 |
| 20100 | | 1.63 | 1.95 |
| 45150 | | | 2.11 |

Table 4. The effect of the number of projections upon the least square errors (%) in 3D reconstruction. *FrameSize* gives the dimensions of the pixel frames in number of pixels, *Diameter* is the diameter of the sphere in number of pixels, and *NumberViews* is the number of projections used for reconstruction.

| NumberViews | Case 1<br>FrameSize 21 × 21<br>Diameter 16<br>Aspect ratio 4 | Case 2<br>FrameSize 41 × 41<br>Diameter 32<br>Aspect ratio 4 | Case 3<br>FrameSize 61 × 61<br>Diameter 48<br>Aspect ratio 4 |
|---|---|---|---|
| 300 | 0.97/1.00 | 0.96/1.00 | 0.96/1.00 |
| 1250 | 0.98/1.00 | 0.97/1.00 | 0.96/1.00 |
| 2775 | 0.97/1.00 | 0.97/1.00 | 0.96/1.00 |
| 5000 | 0.98/1.00 | 0.97/1.00 | 0.96/1.00 |
| 11250 | | 0.97/1.00 | 0.96/1.00 |
| 20100 | | 0.97/1.00 | 0.96/1.00 |
| 45150 | | | 0.96/1.01 |

Table 5. The effect of the number of projections upon the minimum/maximum density values inside the sphere. *FrameSize* gives the dimensions of the pixel frames in number of pixels, *Diameter* is the diameter of the sphere in number of pixels, and *NumberViews* represents the number of projections used for reconstruction.

Both versions of parallel 3D reconstruction program produced results close to those of EM3DR, a sequential program based upon the method described in [Cro70] and widely used for structural studies using cryo-EM data. The improved algorithm leads to slightly better agreement with the results produced by the sequential program.

Table 8 indicates that the least square errors are slightly larger than the ones reported above for zero-fill aspect ratio of 1. Table 9 indicates that the range of density values inside the sphere is slightly wider than the ones reported above even for $k = 1$. Figure 11 shows the distribution of the density for for several cases.

**9. The performance of a parallel program implementing the 3D reconstruction algorithm.** A program implementing the algorithm presented in this paper was written and tested using internal data as described in the previous section as well as several experimental data sets. The program is written in Fortran, uses the MPI library for communication, and was designed to run efficiently on a cluster of inexpensive PCs. We use a cluster of 16, 400 MHz Pentium II processors running SunOS5.6. Each processor has 256 MB of main memory and a 8 GB disk. The connectivity is provided by a 100 MBps Ethernet switch. The total cost of the system is about $40K. The actual performance of this system is comparable for this problem with the performance of a 16 processor SGI Origin 2000.

The program is based upon a data parallel execution model, all nodes perform essentially the same computation but on different data. The coordinator node reads the input files containing the set of projections and the orientation of each projection and then distributes the projections evenly among the set of available nodes. Then, each node processes the individual frames assigned to it; first it transforms each frame and expands it, if the zero-fill aspect-ratio $k > 1$ and then carries

| $k\backslash V$ | 25 | 50 | 75 | 100 | 150 | 200 |
|---|---|---|---|---|---|---|
| 1 | 21.67 | 20.69 | 20.39 | 20.81 | 20.68 | 20.62 |
| 4 | 8.84 | 4.47 | 2.08 | 1.68 | 1.85 | 1.97 |

Table 6. The effect of the zero-fill aspect ratio, $k$, and the number of projections, $V$, upon the errors (%) inside the sphere. The true values are 1 inside the sphere and 0 outside.

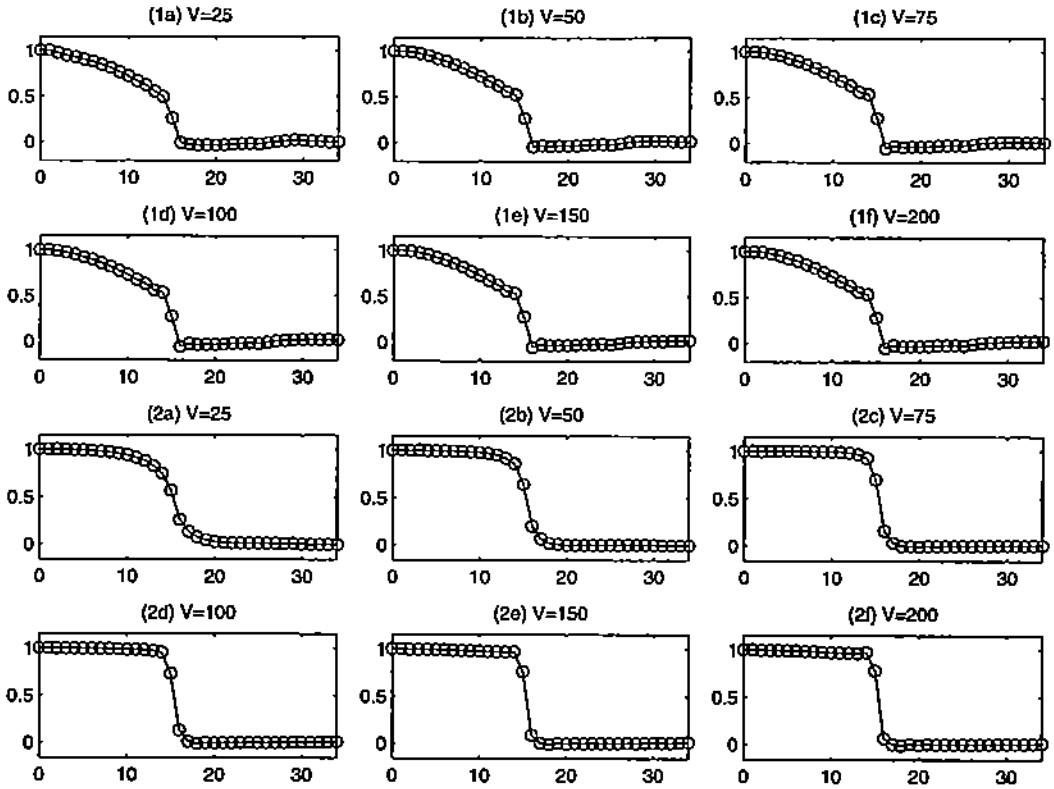


Figure 7. The effect of the number of views, $V$, upon the distribution of the density function of the radius of the sphere. For Figures (1a)-(1f), the aspect ratio of zero-fill $k = 1$; for Figures (2a)-(2f), $k = 4$. The size of pixel frames is $41 \times 41$, and the diameter of the sphere is 32.

out a 2D DFT. A data exchange stage occurs at the end of the Fourier analysis phase, each node is assigned a set of linear equations. After solving the linear systems the nodes carry out a 2D DFT then a global exchange takes place and a 1D FFT completes the Fourier synthesis phase. Finally, the coordinator node gets individual sections of the 3D map from the other nodes and writes the electron density map out. We have opted for this solution because we do not have a parallel file system and several nodes reading the input data concurrently and then writing the output density maps concurrently would lead to an unacceptable performance degradation due to I/O contention.

We are primarily interested in the load balancing properties of the algorithm and in the speedup of the implementation. While the tests conducted with the internal data presented in Section 7 gave us enough confidence in the correctness of the algorithm and its implementation, we used actual data collected in cryo-EM experiments as shown in Table 10, to further test the correctness of our program. We used only data for symmetric objects because our objective was to compare our results with the results produced by a sequential program widely used by the structural biology community. Figure 12 shows the picture of Paramecium Bursaria Chlorella Virus, type 1 (or PBCV_1), reconstructed in Cases E and I in Table 10.

A first objective of our analysis is to profile the program and determine the time used for each execution phase. Table 11 shows that interpolation is the most intensive phase, followed by the 2D Fourier analysis, while solving the linear systems requires a relatively low amount of arithmetic

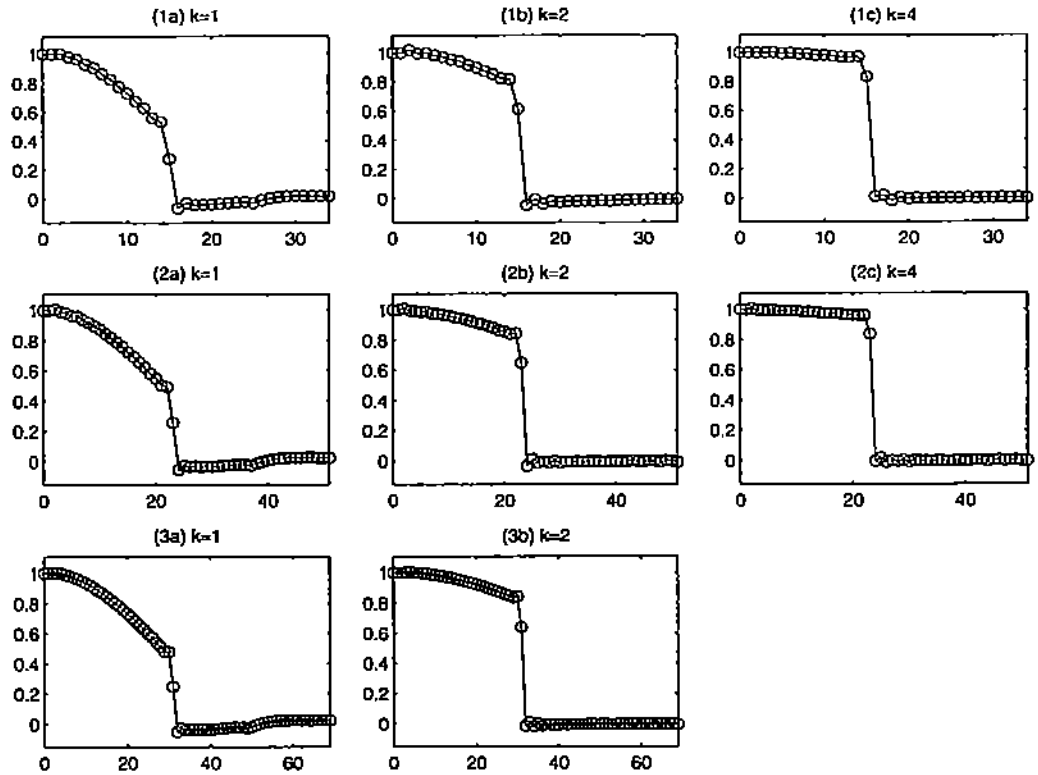


Figure 8. The effect of the zero-fill upon the distribution of the density inside and outside the sphere function of the radius. The computed values for aspect ratio of zero-fill 1, 2 and 4. The size of pixel frames, the number of pixel frames,, and diameter of the sphere differ: for (1a)-(1d) $41 \times 41$, 20100, and 32, for (2a)-(2d), $61 \times 61$, 45150, and 48, for (3a)-(3d), $81 \times 81$, 80802, and 64.

| Virus | Polyomavirus | Papillomavirus |
|---|---|---|
| Number of views | $158 \times 60$ | $60 \times 60$ |
| Size of pixel frame | $69 \times 69$ | $99 \times 99$ |
| Size of output | $35 \times 35 \times 35$ | $49 \times 49 \times 49$ |
| Average density | 289.604 | 80.4715 |
| Mean difference ($\overline{e}$) | 12.5827 | 0.7743 |
| Variance ($\sigma$ ) | 36.1211 | 7.8799 |
| Coefficient of Variation ($\delta$ ) | 12.4726% | 9.7922% |

Table 7. The difference in the densities computed with a 3D parallel reconstruction program based upon the algorithm described in this paper and the ones computed with a sequential program.

operations. In [Lyn99] we reported that solving the linear systems was the most time-consuming phase of 3D reconstruction. The expected improvements of the algorithm described in this paper are confirmed by the measurements.

Our next objective is to study the load balancing properties of the algorithm and the speedup. Table 12 shows the time used in each node when the program solves one of the problems in multiple nodes. From the data in Table 12, we can see that the computation is evenly distributed among multiple nodes. We need to keep in mind that the coordinator is assigned extra duties in the initial and final phases of the algorithm.

Table 13 shows the seedups, for the nine problems presented above. The speedups are larger than 1.85 for two nodes, larger than 3.5 for four nodes, range from a low of 3.7 to a high of 6.9 in eight nodes and from 6.8 to 11.1 in 16 nodes. In case of problem H and I due to the problem size we were unable to run in one node and report only the speedups relative to the running time in two

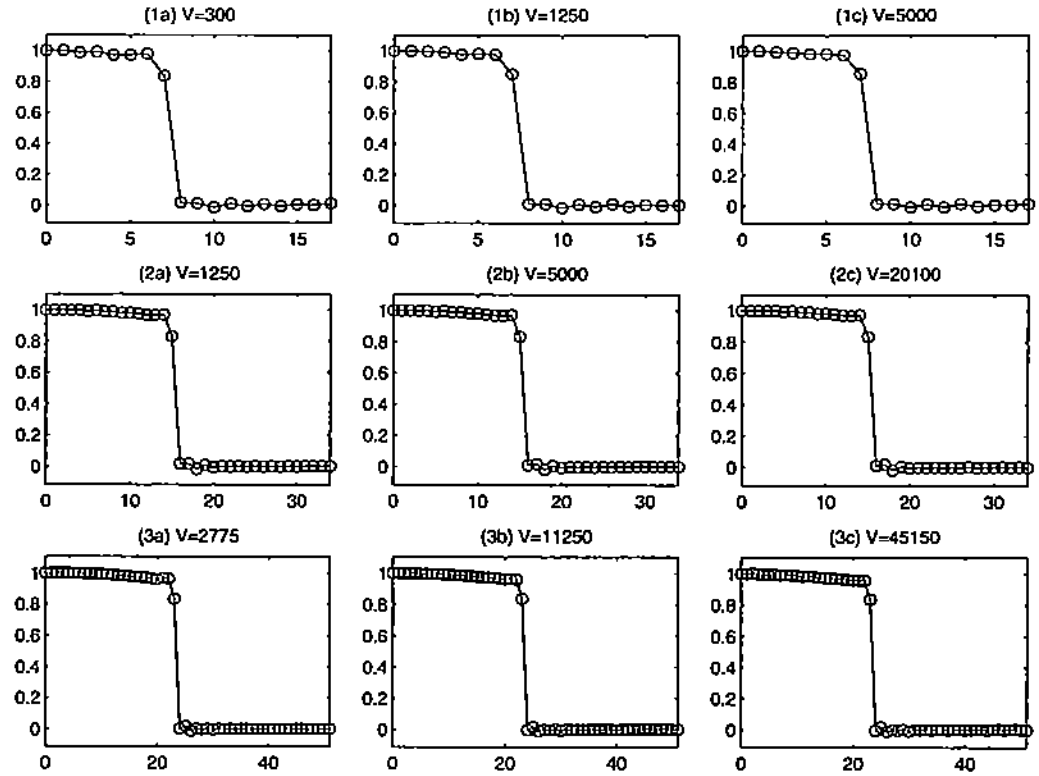


Figure 9. The effect of the number of views upon the distribution of density inside and outside the sphere, function at the radius. The computed values for different numbers of views when the aspect ratio of zero-fill is 4. The size of the pixel frames and the diameter of the sphere are: for (1a)-(1d) 21 × 21, and 16, for (2a)-(2d) 41 × 41, and 32, for (3a)-(3d), 61 × 61, and 48.

| FrameSize | NumberViews | InsideError (%) |
|---|---|---|
| 21 × 21 | 45 × 60 | 28.45 |
| 41 × 41 | 198 × 60 | 51.29 |
| 61 × 61 | 459 × 60 | 33.35 |
| 81 × 81 | 831 × 60 | 32.46 |

Table 8. The least square errors (%) for 3D reconstruction with a widely used sequential program that uses icosahedral symmetry. *FrameSize* and *NumberViews* are the size and number, respectively, of pixel frames. *InsideError* is least square error inside the sphere.

nodes.

**10. Exploiting symmetry .** The 3D reconstruction algorithm was specifically designed for asymmetric objects. However, when it is used for objects with known symmetry, such as dihedral, icosahedral, or other kind of symmetry, the symmetry can be used to decrease the execution time and only a portion of the object has to be reconstructed.

Such a symmetric object is composed of a number $A$ asymmetric units, e.g., an object with icosahedral symmetry consist of $A = 60$ asymmetric units, one with dihedral symmetry has $A = 10$. If a projection plane is normal to the orientation of one of the asymmetric units, then for each of the other asymmetric units there is an orientation normal to the same projection.

The symmetry of an object allows us to use fewer pixel frames for its 3D reconstruction. One particle projection can be used repeatedly. For example, in case of icosahedral symmetry, one projection has associated with it 60 orientations; in dihedral symmetry, one projection has 10 orientations. This allow us to carry out the Fourier Transform of a pixel frame only once and use the transform $A$ times in estimating the 3D DFT by interpolation. In case of a symmetric particle the number of

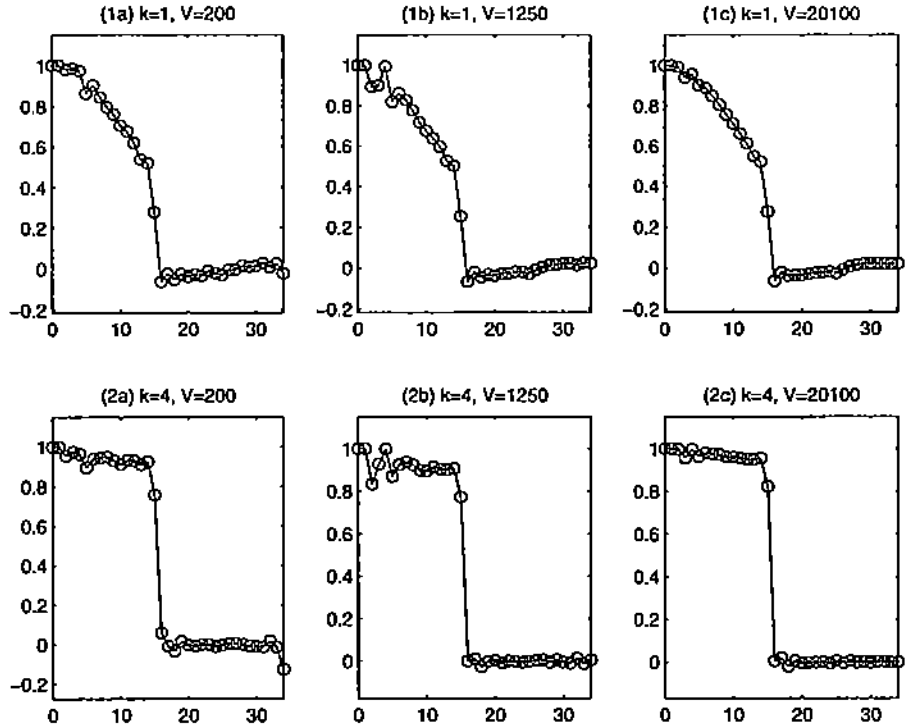


Figure 10. The effect of noise. Pixel values were computed for the 41 × 41 projections of the uniform sphere with diameter of 32. Then uniformly distributed random numbers equal to 5% of the maximum projected density were added to the pixel values.

| FrameSize | NumberViews | InMin | InMax | OutMin | OutMax |
|---|---|---|---|---|---|
| 21 × 21 | 45 × 60 | 0.4787 | 1.0000 | 0.0000 | 0.1615 |
| 41 × 41 | 198 × 60 | 0.9225 | 1.8677 | −0.0009 | 0.4451 |
| 61 × 61 | 459 × 60 | 0.3715 | 1.0000 | −0.0199 | 0.1870 |
| 81 × 81 | 831 × 60 | 0.6659 | 1.5402 | −0.0859 | 0.3422 |

Table 9. The minimum and maximum density for 3D reconstruction with a widely used sequential program that uses icosahedral symmetry. *FrameSize* and *NumberViews* are the size and number, respectively, of pixel frames. *InMin* and *InMax* are the minimum and the maximum of the density value inside the sphere, *OutMin* and *OutMax* are the values outside the sphere, respectively. The true values are 1 inside and 0 outside.

arithmetic operations increases, because the number of projections is multiplied by $A$. At the same time we have a corresponding reduction in the amount of I/O operations and the input data size simply because we reduce the number of projections by $A$.

The symmetry in real space imposes symmetry in reciprocal space and this effect can be used to reduce the number of arithmetic operations required to compute the 3D DFT. For example to reconstruct an object with icosahedral symmetry, we only need to estimate the 3D DFT in one of the octants instead of four. This results in the reduction of the number of arithmetic operations for interpolation as well as in solving linear systems by a factor of four.

**11. Motivations for using parallel algorithms for 3D reconstruction of asymmetric objects.** The 3D reconstruction algorithm proposed by Crowther [Cro70] almost 30 years ago has been used extensively by the structural biology community ever since. The algorithm is based upon Fourier-Bessel Transforms and can be used for reconstruction of symmetric objects.

The protein shell of a spherical virus exhibits various degrees of symmetry, but the core of the virus consisting of genetic material does not. Structural studies of the virus core provide the motivation for 3D reconstruction algorithms of asymmetric objects.

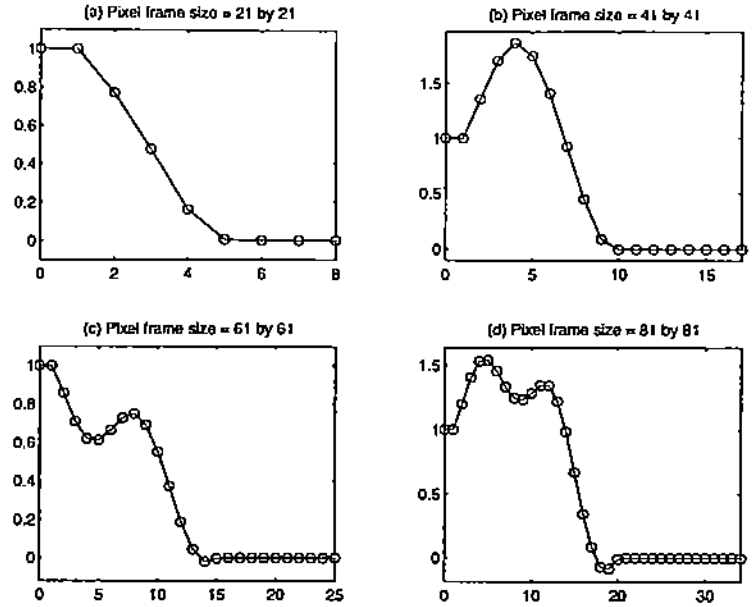


Figure 11. The density function at the radius for the sequential program. The size of pixel frames and the number of pixel frames are: (a) T 21 × 21 and 45 × 60, (b) 41 × 41 and 198 × 60, (c) 61 × 61 and 459 × 60, and (d) 81 × 81 and 831 × 60.

| Problem | Virus | Pixels | Views | Symmetry |
|---|---|---|---|---|
| A | Polyomavirus (Papovaviruses) | 69 × 69 | 158 × 60 | Icosahedral |
| B | Papillomavirus (Papovaviruses) | 99 × 99 | 60 × 60 | Icosahedral |
| C | Sindbis virus (Alphaviruses) | 221 × 221 | 389 × 60 | Icosahedral |
| D | Sindbis virus (Alphaviruses) | 221 × 221 | 643 × 60 | Icosahedral |
| E | Paramecium Bursaria Chlorella Virus, type 1 | 281 × 281 | 107 × 60 | Icosahedral |
| F | Ross River Virus (Alphaviruses) | 131 × 131 | 1777 × 10 | Dihedral |
| G | Bacteriophage Phi29 | 191 × 191 | 609 × 10 | Dihedral |
| H | Auravirus ( Alphaviruses) | 331 × 331 | 1940 × 60 | Icosahedral |
| I | Paramecium Bursaria Chlorella Virus, type 1 | 359 × 359 | 948 × 60 | Icosahedral |

Table 10. Data for 9 (nine) problems used to test the parallel 3D reconstruction program. The virus family is indicated in parenthesis. The number of pixels, the number of views/projections and the types of symmetry are indicated.

The amount of experimental data for the reconstruction of an asymmetric object is considerably larger than the one for a symmetric one as discussed in Section 10. While a typical reconstruction of an icosahedral virus at say 20 Å resolution may require a few hundreds projections, e.g. 300, the reconstruction of an asymmetric object with the same dimensions and at the same resolution would require 60 times more data, i.e. 18,000 projections.

X-ray crystallography is the only method to obtain high resolution ($2 - 2.5$ Å ) electron density maps for large macromolecules like viruses, while until recently electron microscopy was only able to provide low resolution (20 Å ) maps. Cryo-EM is appealing to structural biologists because crystallizing a virus is sometimes impossible and even when possible, it is technically more difficult than preparing samples for microscopy. Thus the desire to increase the resolution of cyo-EM methods to the 5 Å range. In the last years results in the $7 - 7.5$ Å range have been reported, [Böt97], [Con97]. But increasing the resolution of the 3D reconstruction process requires more experimental data. It is estimated that the number of views to obtain high resolution electron density maps from

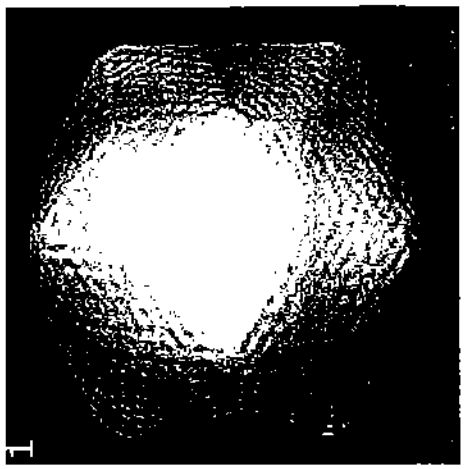

Figure 12. The Paramecium Bursaria Chlorella Virus, type 1, reconstructed using the algorithm and the program discussed in this paper. Result presents as cases E and I.

| Execution Phase | A | B | C |
|---|---|---|---|
| Initialization | 0.26 | 0.13 | 0.13 |
| 2D Fourier analysis | 1.75 | 1.2 | 75.9 |
| Interpolation | 11.49 | 8.34 | 270.2 |
| Data Exchange for solvesys | 0.0016 | 0.006 | 0.073 |
| Solvesys and combine | 0.019 | 0.053 | 0.64 |
| 2D Fourier synthesis | 0.067 | 0.23 | 4.49 |
| Data Exchange for 1D synthesis | 0.010 | 0.027 | 0.33 |
| 1D Fourier synthesis | 0.030 | 0.10 | 2.15 |
| Gather data | 0.0050 | 0.015 | 0.18 |
| Write density map | 0.2 | 0.45 | 3.11 |

Table 11. Time (in seconds) for each phase of the 3D reconstruction program for problems A,B and C running in one node.

cryo-EM micrographs should increase by two order of magnitude from the current hundreds to tens of thousands.

The amount of experimental data may further increase because structural studies of even larger virus-antibody complexes may be necessary. Last but not least, using larger zero-fill aspect ratios to improve the accuracy of 3D reconstruction will increase the number of arithmetic operations and the amount of space needed for reconstruction. Thus it is not unrealistic to expect an increase in the volume of experimental data for high resolution asymmetric objects by three to four orders of magnitude in the next future.

Even though nowadays faster processors and larger amounts of primary and secondary storage are available at a relatively low cost, the 3D reconstruction of asymmetric objects at high resolution requires computing resources, CPU cycles, primary and secondary storage, well beyond those provided by a single system. Thus the need for parallel algorithms.

**12. Conclusions.** Recently, we proposed a parallel 3D reconstruction algorithm for objects without symmetry based upon Fourier analysis using Cartesian coordinates [Lyn97], [Lyn99].

In this paper we discuss an improvement of our method which, for reconstruction at points of an $N \times N \times N$ grid, uses $O(N^3)$ arithmetic operations instead of $O(N^5)$ as in the original method. Though developed for structural biology studies the algorithm is general and can be used for any applications to reconstruct a 3D object from its 2D projections.

We report the results of an error analysis that shows that embedding the pixel frames into larger arrays, a technique we call "zero-fill", helps lower the numerical errors in the reconstruction process but increases the amount of space and the number of arithmetic operations. For example we can

| 1 | 2 | 4 | 8 | 16 |
|---|---|---|---|---|
| | | | | 60.63 |
| | | | | 57.56 |
| | | | | 57.55 |
| | | | | 57.55 |
| | | | 91.70 | 57.55 |
| | | | 88.68 | 57.55 |
| | | 161.99 | 88.70 | 57.55 |
| 567.22 | 322.45 | 158.79 | 88.69 | 57.55 |
| | 318.88 | 158.81 | 88.69 | 57.55 |
| | | 158.76 | 88.69 | 57.55 |
| | | | 88.69 | 57.55 |
| | | | 88.69 | 57.56 |
| | | | | 57.55 |
| | | | | 57.55 |
| | | | | 57.55 |
| | | | | 57.56 |

Table 12. Time (in seconds) used by each node for problem D, Execution with 1, 2, 4 , 8, and 16 nodes.

| Input \ Node Number | 2 | 4 | 8 | 16 |
|---|---|---|---|---|
| A | 1.82 | 2.79 | 1.16 | |
| B | 1.85 | 2.86 | 1.54 | |
| C | 1.82 | 3.57 | 6.18 | 8.59 |
| D | 1.76 | 3.50 | 6.19 | 9.35 |
| E | 1.73 | 3.25 | 4.31 | 5.98 |
| F | 1.94 | 3.63 | 6.32 | 7.73 |
| G | 1.92 | 3.21 | 5.77 | 7.00 |
| H | | 1.93 × 2 | 4.04 × 2 | 7.62 × 2 |
| I | | 1.87 × 2 | 2.77 × 2 | 4.84 × 2 |

Table 13. The speedups attained by the 3D reconstruction program for the nine sample problems. The last two problems could only run in two or more nodes.

reduce the error inside a uniform sphere from about 25% for a zero-fill aspect ratio of $k = 1$ to less than 4.5% for $k = 4$. For the sake of completeness we report on errors both inside and outside a uniform sphere though for practical purposes only the errors inside matter.

The magnitude of the least square errors of a 3D reconstruction program based upon the algorithm presented in this paper is slightly lower than the ones for a sequential program based upon the algorithm described in [Cro70] when the zero-fill aspect ratio is one and significantly lower when we increase $k$.

In practice, the data collected in cryo-electron microscopy is subject to experimental errors due to various factors e.g. variations of the intensity of the beam, the non-uniform layer of ice, and other sources of noise. Additional errors occur when extracting the individual projections from the micrographs, determining the center of each projection, etc. The traditional wisdom is that using a number of projections much larger than the minimum number required for reconstruction, see [Ros98], the effect of errors can be overcome. Indeed many structures have been solved along the years yet the reconstruction was carried out at relatively low resolution.

Our results confirm our intuition that errors have a non-uniform distribution, the further we are from the center of the sphere the larger are the errors. We studied also the effect of the number of projections upon the magnitude of errors. Since we are performing a Monte Carlo calculation, we expect that the least square error should decrease as $1/\sqrt{number\ of\ views}$. Interestingly enough, the least square errors seem to decrease even slower than the rate above, e.g. in one case, the error

in the density inside a uniform sphere was only 4.59 % for 1250, 4.51 % for 5000, and 4.45 % for 20100 projections.

We profiled the program and report results regarding the parallel aspects of our algorithm, namely load balance and speedup. As expected, the most time consuming phases of the program execution are: (a) the interpolation, (b) the 2D Fourier analysis, and (c) the initialization phase where input files containing the projections and the orientation of each projection are read in. Recall that in our previous experiments [Lyn99] we reported that the most time consuming phase of the program was solving linear systems.

The load balancing results are very good. In most cases the execution times of all but the coordinator node are within 1 % of each other. The need to exchange data among nodes and to synchronize after each phase reduces somewhat the speedup on realistic problems. We report on the results of 3D reconstruction for 8 virus structures. The speedup in 4 nodes is about 3.5, ranges from a low of 3.7 to a high of 6.9 for 8 nodes and is in the range of 7 to 11 for 16 nodes. While the problem size may be too small in some of the cases we report on there is no doubt that further improvements in the implementation of the algorithm are needed.

Finally, we point out that the experiments reported in this paper were conducted on a low-cost parallel system consisting of a cluster of 16 Pentium II processors running at 400 MHz, each with 256 MB of memory and interconnected by a 100 Mbps Ethernet. With a faster interconnection network our results would be better.

**13. Acknowledgments.** The authors are grateful for many insightful discussions with Michael G. Rossmann and Timothy S. Baker. Robert Ashmore from Baker's lab provided assistance with the sequential reconstruction program. We especially thank Wei Zhang, a graduate student in Biology, for very helpful information about the EM3DR program and data.

## Appendix

**A1. Pixel Frame Orientation.** By computer processing, the orientation with respect to the Standard XYZ System of each pixel frame is determined [Bak97]. The orientation of a pixel frame with respect to the XYZ System is specified by three angles $\theta$, $\phi$, $\omega$.

These angles lead to the coordinate transformation from a point in the RST System of the pixels to a point in the XYZ System of the macromolecule. Except for the units of length along the axes, this transformation from the RST System to the XYZ System in real space is the same as that in reciprocal space from the UVW System to the HKL System; this is because both coordinate systems are right-handed Cartesian systems.

In terms of the coordinates of the XYZ System, the unit normal of the plane containing the pixel frame is the vector

$$(\sin(\theta)\cos(\phi),\ \cos(\theta)\cos(\phi),\ \cos(\theta)); \tag{25}$$

the orientation of this unit vector is in the direction from the frame towards the macromolecule (i.e., its direction is opposite to the direction of travel of the electrons). The angle $\omega$ specifies a clockwise rotation about this normal. Equivalently, this vector gives the coordinates in the HKL System of the unit normal to the UV-plane on which the DFT of the pixel values is obtained.

The angles $\theta$ (colatitude) and $\phi$ (longitude) are those of spherical coordinates for the HKL System. $\theta$ is the angle from the positive L-axis to the normal vector (25) and $\phi$ is the angle from the positive H-axis to the projection of the normal vector (25) onto the HK-plane. $\omega$ is the clockwise (left-handed) rotation, looking from the end of the normal to the origin, of the frame.

A right-handed (counterclockwise) rotation of the HK-plane about the $L = L_1$-axis through the angle $\phi$ moves the unit axes H and K to $H_1$ and $K_1$, respectively.

Then a right-handed rotation of the $H_1L_1$-plane about the $K_1 = K_2$-axis through the angle $\theta$ moves the unit vectors $H_1$ and $L_1$, to $H_2$ and $L_2$, respectively. The resulting unit $L_2$-axis has coordinates $(h_2, k_2, \ell_2)$ with respect to the HKL System as given in (25) above.

Finally, the $H_2K_2$-plane is rotated clockwise through the angle $\omega$ about the $L_2 = L_3$-axis. The unit vectors $H_2$ and $K_2$, are moved to $H_3$ and $K_3$, respectively.

**A2. Coordinate Transformation.** The result of the three rotations described in Appendix

A1 can be expressed in term of a matrix product. A column vector $(h,k,\ell)^T$ in the HKL System is rotated to the vector $(h_3,k_3,\ell_3)^T$:

$$(h_3,k_3,\ell_3)^T = D(\omega)C(\theta)B(\phi)(h,k,\ell)^T = E(\phi,\theta,\omega)(h,k,\ell)^T,$$

where

$$B(\phi) = \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad C(\theta) = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix},$$

$$D(\omega) = \begin{pmatrix} \cos\omega & -\sin\omega & 0 \\ \sin\omega & \cos\omega & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

and

$$E(\omega,\theta\,\phi) = \begin{pmatrix} \cos\omega\cos\theta\cos\phi + \sin\omega\sin\phi & \cos\omega\cos\theta\sin\phi - \sin\omega\cos\phi & -\cos\omega\sin\theta \\ \cos\omega\cos\theta\cos\phi - \cos\omega\sin\phi & \sin\omega\cos\theta\sin\phi + \cos\omega\cos\phi & -\sin\omega\sin\theta \\ \sin\theta\cos\phi & \sin\theta\cos\phi & \cos\theta \end{pmatrix}.$$

Each of these four matrices is an *orthogonal* matrix; that is, each is real and its inverse is its transpose; thus

$$(h,k,\ell)^T = B^T(\phi)C^T(\theta)D^T(\omega)(h_3,k_3,\ell_3)^T = E^T(\phi,\theta,\omega)(h_3,k_3,\ell_3)^T.$$

The matrix $E$ would map a point in the HKL System to a point in the UVW System provided that the units of length in the two systems were the same. Denote the conversion factor by $\sigma$:

$$\sigma = \frac{\text{unit of length in the UVW System}}{\text{unit of length in the HKL System}} = \frac{\Delta U}{\Delta H}.$$

The transformation of a point in the HKL System to one in the UVW System is then given by

$$(u,v,w)^T = \sigma E(\phi,\theta,\omega)(h,k,\ell)^T,$$

and the inverse is given by

$$(h,k,\ell)^T = (1/\sigma)E^T(\phi,\theta,\omega)(u,v,w)^T.$$

In particular, the third column of $E^T(\phi,\theta,\omega)$ is the column form of the unit normal (25) and and

$$(h,k,\ell)^T = (1/\sigma)E^T(\phi,\theta,\omega)(0,0,1)^T$$

gives that normal in the HKL System, but with its length having been adjusted for the difference in the units of length in the UVW System and HKL System.

The transformations given above require that each pixel be square. They must be modified if, for example, the pixels are (nonsquare) rectangles.

## REFERENCES

[And92] Anderson, E., Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A, McKenney, S. Ostrouchov, and D. Sorenson, LAPACK Users' Guide, SIAM Publications, 1992.

[Bak88] Baker, T. S., J. Drak, and M. Bina, "Reconstruction of the three-dimensional structure of simian virus 40 and visualization of chromatin core", Proc. Natl. Acad. Sci. USA, 85:422-426, 1988.

[Bak97] Baker, T. S., I. M. B. Martin, and D. C. Marinescu, "A parallel algorithm for determining orientations of biological macromolecules imaged by electron microscopy", CSD-TR 97-055, 1997.

[Böt97] Böttcher, B., S. A. Wynne, and R. A. Crowther, "Determination of the fold of the core protein of hepatitis B virus by electron cryomicroscopy", Nature (London) 386, 88–91, 1997.

[Bri95] Briggs, W. L., and V. E. Henson, *The* DDT, *An Owner's Manual for the Discrete Fourier Transform*, SIAM Publications, 1995.

[Con97] Conway, J. F., N. Cheng, A. Zlomick, P. T. Wingfield, S. J. Stahl, and A. C. Steven, "Visualization of a 4-helix bundle in the hepatitis B virus capsid by cryo-electron microscopy". Nature (London) 386, 91–94, 1997.

[Cro70] Crowther, R. A., D. J. DeRosier, and A. Klug, "The reconstruction of a three-dimensional structure from projections and its application to electron microscopy", Proc. Roy. Soc. Lond. A 317, 319–340, 1970.

[Dea93] Deans, S. R., *The Radon Transform and Some of Its Applications*', 2nd Edit., Krieger Publishing Company, 1993.

[Fra96] Frank, J., *Three-Dimensional Electron Microscopy of Macromolecular Assemblies*, Academic Press, 1996.

[Gor74] Gordon, R., "Three-dimensional reconstruction from projections: A review of algorithms", Intern. Rev. of Cytology 38, 111–151, 1974.

[Gra96] Grangeat, P., and J-L Amans, Eds., *Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine*, Kluwer Academic Publishers, 1996.

[Gol96] Golub, G. H., and C. F. Van Loan, *Matrix Computations* 3rd Edit., The Johns Hopkins University Press, 1996.

[Her79] Herman, G. T., Editor: *Image Reconstruction from Projections, Implementation and Applications*, Springer-Verlag, 1979

[Joh94] Johnson, C. A., N. I. Weisenfeld, B. L. Trus, J. F. Conway, R. L. Martino, and A. C. Steven, "Orientation determination in the 3D reconstruction of icosahedral viruses using a parallel computer", CS & E, 555–559, 1994.

[Lyn97] Lynch, R. E., and D. C. Marinescu, "Parallel 3D reconstruction of spherical virus particles from digitized images of entire electron micrographs using Cartesian coordinates and Fourier analysis", CSD-TR #97-042, Department of Computer Sciences, Purdue University, 1997.

[Lyn99] Lynch, R. E., D. C. Marinescu, H. Lin, and T. S. Baker, "Parallel algorithms for 3D reconstruction of asymmetric objects from electron micrographs," Proc. IPPS/SPDP (13th International Parallel Processing Symposium and 10th Symposium on Parallel and Distributed Processing), pp. 632–637, 1999.

[Mar97] Martin, I. M., D. C. Marinescu, T. S. Baker, and R. E. Lynch, "Identification of Spherical particles in digitized images of entire micrographs", J. of Structural Biology, 120, 146–157, 1997.

[Ros98] Rossmann, M. G., and Y. Tao, "Cryo-electron-microscopy reconstruction of partially symmetric objects", J. Structural Biology, 125, 196–208, 1999.