3D Reconstruction of Macromolecules on Multiprocessors

M. Ujaldon
R. Asenjo
J.C. Cabaleiro
J.M. Carazo
E.L. Zapata

January 1994
Technical Report No: UMA-DAC-94/02

Published in:

2nd Euromicro Workshop on Parallel and Distributed Processing
Malaga, Spain, January 26-28, 1994, pp. 260-267

University of Malaga
Department of Computer Architecture
C. Tecnologico • PO Box 4114 • E-29080 Malaga • Spain

3D Reconstruction of Macromolecules on Multiprocessors

M. Ujaldón, R. Asenjo, E.L. Zapata
Dpto. Arquitectura de Computadores
Universidad de Málaga
29013 Málaga. Spain

J.C. Cabaleiro
Dpto. Arquitectura de Computadores
Universidad de La Coruña
La Coruña. Spain.

J.M. Carazo
Centro Nacional de Biotecnología
Canto Blanco.
Madrid. Spain.

Abstract

The filtered backprojection algorithm is a popular method for the reconstruction of n-dimensional signals from their (n-1) dimensional projections. In this work we will treat the particular problem of the three dimensional reconstruction of a 3D object from 2D images. The parallel algorithm we develop is general in the sense that it does not impose any restrictions on the size of the problem and is not dependent on the dimensions of the mesh. We have also compared this algorithm in multiprocessor architectures with hypercube and mesh arrangements, obtaining similar computation times. This last aspect indicates that the hypercube increases the hardware cost with no significant improvement in the efficiency of the algorithms.

1: Sequential algorithm.

The foundation on which all three dimensional reconstruction techniques are based was originally developed by Radon in 1917 [1]. He analyzed what is now known as the Randon Transform and its inverse.

Currently, three dimensional macromolecule reconstruction algorithms can be grouped into two large sets: On one hand we find the series expansion methods, among which ART (Algebraic Reconstruction Technique) and its many variants stand out. On the other we have reconstruction methods based on a discrete version of the inverse Radon transform.

This work was supported by the Ministry of Education and Science (CICYT) of Spain under project TIC-92-0942-C03.

In this work we will concentrate on this second approach, and, in particular, on the convoluted backprojection or the filtered backprojection algorithm.

Our filtered backprojection algorithm will use a random conic geometry. This way, each image will be taken with an azimuthal angle Φ to the Z axis and an inclination angle Θ with respect to the Y axis.

The rotation matrix corresponding to this geometry will give us the relationship between the (X, Y, Z) coordinates of the object with respect to its geometric center and the corresponding coordinates (x_p,y_p) in each of the projections. This transformation is described by the following equation:

$$(x_p,y_p) = [A] * (X, Y, Z)^T \tag{1}$$

where $[A] = D_\Theta * D_\Phi$, being D_Θ and D_Φ the rotation matrices of image *p* with respect to the Y and Z axis as a function of the angles Θ and Φ at which the image is taken:

$$[A] = \begin{bmatrix} -\sin(\theta) & \cos(\theta)*\sin(\phi) & \cos(\theta)*\cos(\phi) \\ 0 & \cos(\phi) & -\sin(\phi) \end{bmatrix}$$

The direct backprojection of each of the images is obtained by means of the application of matrix [A] to each point of the three dimensional cube corresponding to the object as indicated by equation (1), obtaining this way the 2D coordinates in the image for each 3D point. After this we add its contribution to the 3D point under consideration. The process is repeated for each of the images.

As a result of (1), we can find two different situations. The first one happens when the coordinates generated

(x_p,y_p) are outside the image and consequently, the p-th coordinate does not contribute to the 3D point of the object. The second situation occurs when (x_p,y_p) belongs to the p-th image. In this case, as x_p and y_p are generally real numbers, the intensity associated with these coordinates is calculated by means of a bilinear interpolation process of the type of:

$$a+(b-a)\delta_x+(c-a)\delta_y+((d-c)-(b-a))\delta_x\delta_y \quad \textbf{(2)}$$

where a, b, c, d are the intensities of the image in pixels (x_p,y_p),(x_p+1,y_p),(x_p,y_p+1) and (x_p+1,y_p+1) respectively and δ_x and δ_y the fractional parts resulting from eq. (1).

Each image produces a backprojected body [2] which is a three dimensional cube that uniformly distributes each image in a direction perpendicular to its projection plane. In a simple backprojection algorithm, these backprojected bodies are rotated in space using the inclination and azimuthal angles of the projections and then they are added up [3]. The result is a three dimensional reconstruction that is an approximation to the density of the object.

Despite all of this, a simple backprojection reconstruction contains systematic errors. A star effect appears around the reconstructed points and this effect increases the density of the reconstructed points. the reconstruction can be improved using an extended point function or a transference function in the frequency domain that can reduce the density of the reconstructed points by means of an adequate convolution. This leads us to the ***filtered backprojection method***. We must find an expression for the extended point function we will denote as G. A detailed description of how to obtain it can be found in [4]. Its general expression is the following:

$$G(X^*,Y^*,Z^*)=\sum_p 2a\,sinc(2a\pi Z_p^*) \qquad (3)$$

where the asterisk * has been used in order to denote the coordinates of the object in the frequency domain.

The particular filtering functions for each geometry can be easily derived from expression (3). In particular, for a random conic geometry such as the one we are using, we must express the Z coordinate in the objects coordinate system as the 2 rotations carried out over it for each one of the p projections. This produces the following final expression of G for our case:

$$G(X^*,Y^*,Z^*) = \sum_p 2a\ \sin[2a\pi\,(X_p^*\sin(\theta_p)\cos(\phi_p) + Y_p^*\sin(\theta_p)\sin(\phi_p) + Z_p^*\cos(\theta_p))] \qquad (4)$$

In order to obtain the correct reconstruction, the Fourier transform of the backprojection must be divided by G, which is equivalent to applying a deconvolution of the object with the extended point function.

The division by G can be applied to the Fourier transform of the three dimensional object obtained after the backprojection or to the Fourier transforms of all the projections before the backprojection. According to [6], these two paths are mathematically equivalent. However, the computation time needed and the quality of the reconstruction can vary from one solution to the other depending on the number of data points we consider. If we use a small number of projections, then an application of the filter to the projections is more efficient, otherwise it is better to filter the backprojected object.

```
Compute the rot. matrix [A] for each image;
for p:= 1 to NumberImages do
    Backproject image p onto the object;
Compute 3D-FFT of the reconstructed object;
Filter the object with function G;
Filter the object with low-pass filter;
Compute inverse 3D-FFT.
```

Figure 1.- Complete algorithm for performing the filtering operation on the object obtained after backprojection.

As we generally have a large number of projections available, the best path for achieving a good reconstruction is three dimensional filtering (see figure 1). It will be the one we will use here. Nevertheless, our group developed in [5] a parallel implementation in ACLAN language for a more detailed study of the other alternative.

The approach chosen presents an additional advantage: It permits the implementation of a low pass filtering operation of the object at the same time it is divided by G, saving costly calculations of the direct and inverse 3D-FFT. This low pass filter is the final step in the three dimensional reconstruction process (see figure 1) and it attenuates the non significant high frequencies present in the calculated volume. In those applications in which the significant frequency range cannot be estimated a priori, this process can only be carried after the volume has been reconstructed by means of the statistical comparison of different 3D reconstructions [6]. In the applications we are concerned with, a simple low pass filter with a radially symmetric step its falling edge smoothed by a cosine function provides good results (even though in [7] we have developed more sophisticated filters that could be easily implemented).

2: The mesh topology.

A mesh is a two dimensional array of processing elements (PEs). In a generic mesh with X x Y PEs (X>0, Y>0), each one of them is connected by means of its N, S, E and W channels to the four neighboring PEs, except for the processors on the edges that only use 3 channels and on the 4 corners that only use 2. The index assigned to each PE is a function of its position (i,j) in the mesh ($0 \leq i < X$, $0 \leq j < Y$) and is given by the following formula: PE= j*X+i.

The method we have followed for the partition/ projection of algorithms onto the mesh is similar to the one developed by Zapata et al. [8][9] for hypercubes. It consists in the following steps: 1) Loop level analysis of the sequential algorithm, detecting the data and control dependencies. The maximum number of independent nested loops (do all) define the dimensions of the algorithmic space. 2) Projection onto the mesh of the two loops which maximize the parallelism, minimize communications and achieve a good load balance; the rest of the loops will be sequentiated. 3) Distribution of the variables participating in the algorithm among the PEs according to the indexing mode and type of distribution chosen. 4) Design of the parallel algorithm. 5) Optimization of the algorithm in steps 2 and 3.

3: Description of the parallel algorithm

3.1: Data distribution.

The sequential algorithm can be divided into three parts: Direct backprojection of each of the images, filtering of the three dimensional object (this computation includes the direct 3D FFT, the calculation of the filtering function and the inverse 3D FFT) and final low pass filtering.

The first part of the algorithm is given by equations (1) and (2). In this process there are three independent nested loops associated with the three dimensions of the object. By means of a 2 partition of the mesh, each PE analyzes a single subvolume of the 3D object. The X and Y dimensions of the object are divided into as many subsets as PEs in the array in that dimension of the mesh, and dimension Z is sequentiated. During this process we store the complete current image in each of the PEs, avoiding this way the costly interprocessor communications that are necessary for exchanging image data.

The calculation of the filtering function is carried out by means of equations (3) and (4). From the analysis of these equations we can deduce that there are four nested loops (the there dimensions of the object and the number of projections). In order to avoid the large number of communications associated with a partition change, we will use the same 2 partition we used in the backprojection, sequencing dimension Z. The fourth loop must also be sequentiated. This does not reduce the performance of the algorithm in a significant way because the innermost loop is a short loop and it allows us to use consecutive storage, idoneous for the calculation of the transform.

The sequencing of the Z dimension of the object permits the exploration of the symmetry of the filter in this dimension, running its associated loop only half of its length and thus generating a single weight for the two symmetric coordinates.

The third part of the algorithm is the final low pass filtering process. In this case there are there independent nested loops. As before, we parallelize over the mesh the first two dimensions of the object and we sequentiate the third.

According to the partition and the data distribution scheme chosen, the distribution of the object´s matrix (of dimensions N x N x N) is carried out as follows. Element (i,j,k) of the object is stored in position (i mod w_0, j mod w_1, k) of the local submatrix LOBJ(r_0,r_1,0) of the PEs whose indices r_0 and r_1 are $r_0 = i/w_0$ y $r_1 = j/w_1$, being $w_0 = N/X$ and $w_1 = N/Y$. This way, each PE stores a local submatrix LOBJ of dimensions w_0 x w_1 x N.

3.2: Parallel algorithm for filtered backprojection.

We will use the SCMD (Simple Code Multiple Data) programming model. In figure 2 we present the parallel algorithm for the backprojection executed by each processor. This process has 3 nested loops (lines L1, L2 and L3) which go through all the points of the object, and we parallelize the first two. (xl,yl,zl) and (X,Y,Z) are the local and global coordinates of the object in each PE, and (x_p,y_p) are the coordinates of the p-th projection.

```
void backprojection
{
L1 for (xl=0;xl<w0;xl++)
L2  for (yl=0;yl<w1;yl++)
L3   for (zl=0;zl<N;zl++)
      {
L4     Calculate  (X,Y,Z) from (xl,yl,zl)
L5     Calculate  (x_p,y_p) as a function of
        (X,Y,Z) applying eq.(1)
L6     Interpolate  the value of pixel (x_p,y_p)
        according to  eq.(2)
L7     Add the contribution of the pixel to
        point (xl,yl,zl)
      }
}
```

Figure 2.- Parallel backprojection algorithm.

The next step is to filter the object obtained after the backprojection. In figure 3 we present the algorithm that performs this operation. It has 4 loops (L1, L2, L3 and L7), and we parallelize the first two. In L7 we assign a lower boundary to those weights with values that are too low. Finally, in L8 we filter the coordinate and its symmetric one in the Z dimension (which, due to the data distribution scheme chosen will always be in the same PE).

In order to be able to carry out the filtering operation it is necessary to transform the data to the frequency domain, that is, calculate the 3D FFT over the data of the object obtained after the backprojection. This process is divided into three stages: Bit reversal, which performs a permutation of the data between the PEs, calculation of the internal butterflies with the local data of each processor and calculation of the external butterflies with interprocessor data. An implementation of these 3 stages is shown in [10].

```
void 3dfilter()
{
L1 for (xl=0;xl<w0;xl++)
L2  for (yl=0;yl<w1;yl++)
L3   for (zl=0;zl<N/2;zl++)
      {
L4     Calculate (X,Y,Z) from
        (xl,yl,zl)
L5     for (p=0;p<NumImag;p++)
L6      Calculate the contribution of
         image p and add it to function
         G(X,Y,Z) according to eq.(4)
L7     G(X,Y,Z)=min(alpha,G(X,Y,Z))
L8     Filter (xl,yl,zl) and
        (xl,yl,zl+N/2) with G(X,Y,Z)
      }
}
```

Figure 3.- Parallel algorithm for 3D filter.

We will not calculate the FFT by means of its typical algorithm. We will use a real transform, the T transform, defined as follows:

$$T[f(x,y)](u,v) = T(u,v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y)\ cas(\frac{2\pi ux}{N})\ cas(\frac{2\pi vy}{N})$$

The calculation process of the T transform consists in the same 3 stages we have already mentioned for the FFT and it presents the same complexity, however, it has a double advantage: Significant memory savings, as it does not have to store the complex part of each of the voxels of the object (8Mb in the case of a 128x128x128 object) and a reduction of the computation time, as a complex addition needs two real operations and a product four. Once the T transform has been found the FFT is easily obtained by means of the following expressions:

$$\mathit{RealF}(u,v,w) = \frac{1}{4}[T(N-u,v,w)+T(u,N-v,w)+T(u,v,N-w)-T(u,v,$$
$$+T(N-u,N-v,w)+T(N-u,v,N-w)+T(u,N-v,N-w)-T(N-u,N-v,N-$$

$$\mathit{ImagF}(u,v,w) = \frac{1}{4}[-T(N-u,v,w)-T(u,N-v,w)-T(u,v,N-w)-T(u,v$$
$$+T(N-u,N-v,w)+T(N-u,v,N-w)+T(u,N-v,N-w)+T(N-u,N-v,N-$$

In the FFT calculation process we find, as before, three nested loops corresponding to each one of the dimensions of the object. Out of these we will parallelize the first two over the two dimensions of the mesh. This produces, for stages 1 and 3, communications in the mesh as illustrated in figure 4 for a mesh of X = 4 and Y = 4.

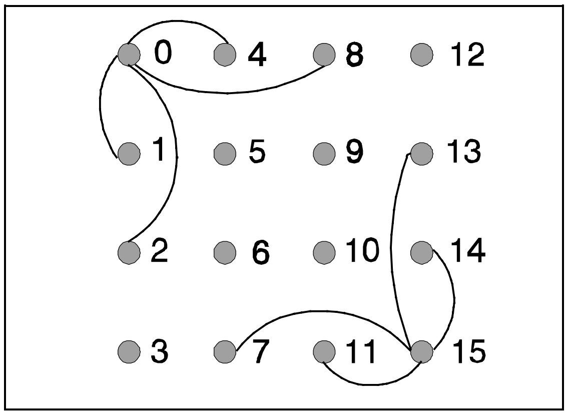

Figure 4.- The four different types of communications carried out in the 3D FFT algorithm on a 4 x 4 mesh for nodes 0 and 15.

The algorithm for the final low pass filtering operation is similar to the one shown in figure 3 and can thus be obtained in an analogous manner.

4: Evaluation.

The complexity of the complete parallel algorithm is a function of the number of projections (M), the size of the object to be reconstructed (N x N x N) and the number of PEs in each of the dimensions of the mesh (X and Y), as is shown in this expression:

O[1 + (M x B) + (2 x T) + F + P]

X	Y	PEs	Backpr. (B)	Filter using G (F)	Low Pass filter (P)
1	1	1	N^3	$N^3 * M$	N^3
N	1	N	N^2	$N^2 * M$	N^2
1	N	N	N^2	$N^2 * M$	N^2
N	N	N^2	N	N * M	N

Table I.- Complexity of the parallel filtered backprojection algorithm for different sizes of the mesh.

B is the complexity of the simple backprojection algorithm. As there are no interprocessor communications, the complexity of the backprojection algorithm only depends on the dimensions of the local volume of each PE:

$B = O[w_0 \text{ x } w_1 \text{ x } N]$

T is the complexity of the 3D FFT algorithm. The factor of 2 multiplying T appears as a consequence of having to perform both the direct and the inverse transform. this complexity is analyzed in detail in [10].

F is the complexity of the filtering process using function G. As in the case of the backprojection, the filtering algorithm using G does not include communications between processors and consequently its complexity is only a function of the local dimensions of the object andthe number of projections:

$F = O\ [M \text{ x } (w_0 \text{ x } w_1 \text{ x } N)]$

N / PE	4 x 4	4 x 2	2 x 2	2 x 1	1 x 1
8	1.643	2.867	5.287	10.067	19.563
16	11.819	21.919	41.413	80.554	158.124
32	91.557	174.104	331.738	649.482	1279.749
64	730.047	1401.952	2639.149	5202.854	10287.19
128	5842.210	11404.85	21099.86	41884.01	83098.69

Table II.- Execution times (in seconds) of the filtered backprojection algorithm on a Transputer network with an object size of N x N x N over different X x Y meshes using 128 projections.

Finally, P is the complexity of the final low pass filtering process. This process is similar to the filter using G, but here the images do not participate and the complexity only depends on the size of the volume.

$$P = O\,[w_0 \times w_1 \times N]$$

Table I presents the complexities of the previous stages for different dimensions of the mesh and a generic size of the problem. We have chose partitions that optimize interprocessor communication times obtaining a behaviour that is very similar to the optimal one. If the size of the mesh is the same as the first two dimensions of the object, the complexity of the algorithm only depends on the number of images and on the third dimension of the object, as they are the only loops we do not parallelize. On the other hand, if the mesh has a single PE, the complexity of the algorithm is the same as for the case of a single processor system. In other words, with the algorithm partition/projection methodology we have used, the sequential execution is just a particular case of the parallel algorithm.

It is interesting to point out that in the general algorithm, the 3 loops which go through the dimensions of the object have the same range of values. From this property we can extract the first conclusions on the performance of the parallel algorithm . This conclusions have been experimentally verified:

- The result obtained does not depend on the two dimensions that are parallelized (X and Y, Y and Z or X and Z).
- If we perform the implementation on a single mesh with X <> Y, the dimension of the object we parallelize over the larger dimension of the mesh is not relevant.

We will now analyze the performance of our algorithm in the mesh topology. In table II we present the execution times (in seconds) for the filtered backprojection algorithm developed in this work. As is shown, multiple executions have been carried out as a function of the size of the object to be reconstructed and the dimensions of the X x Y mesh. A fixed value of 128 has been taken for the number of projections used in the reconstruction process.

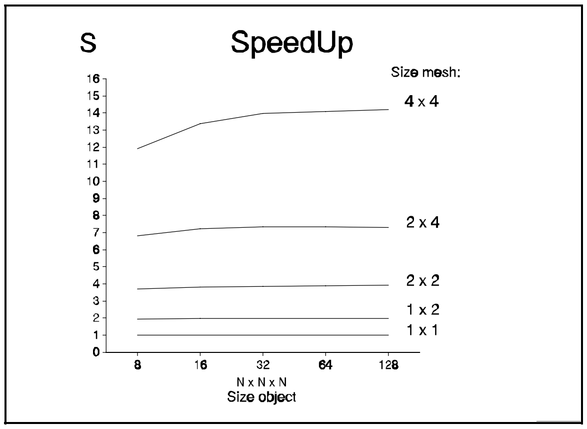

Figure 5.- Gain (S) with respect to the size of the object´s matrix (N) for different mesh sizes.

In figures 5 and 6 we present the gain and the efficiency corresponding to the times in the table. It can be observed how both parameters increase when the size of the problem increases as compared to the dimensions of the mesh (X x Y). This is because the number of communication operations grows with size at a slower rate than the number of local computations, thus reducing the penalizing factor due to communications. This way, we can see how the gain approaches its optimal value (X*Y, equal to the number of PEs). The same happens for the efficiency (whose optimal value is 1), when N >> X*Y, this is, when the local operations predominate over communications.

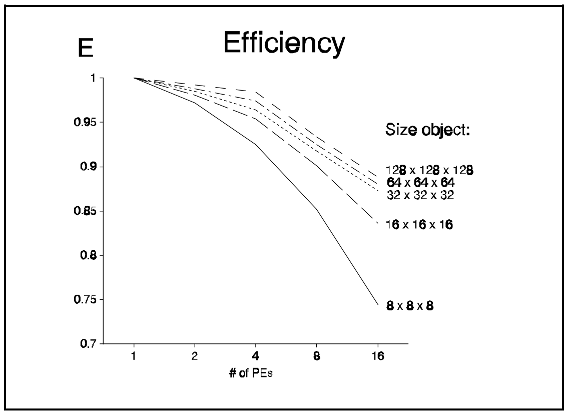

Figure 6.- Efficiency (E) with respect to the number of PEs for different sizes of the matrix.

Finally, in figure 7 we include a comparison of the execution times we have obtained in a hypercube and in a mesh. In this comparison we have considered the following partitions ({(hypercube),mesh}): {(2,2),4x4}, {(2,1),4x2} y {(1,1),2x2}.

From the analysis of figure 7 we can conclude that the execution times are very similar in both topologies, with a slight improvement in the case of the hypercube. This happens because the type 2 and type 4 communications carried out in the FFT algorithm (see figure 4) are direct in the hypercube, whereas in the case of the mesh they

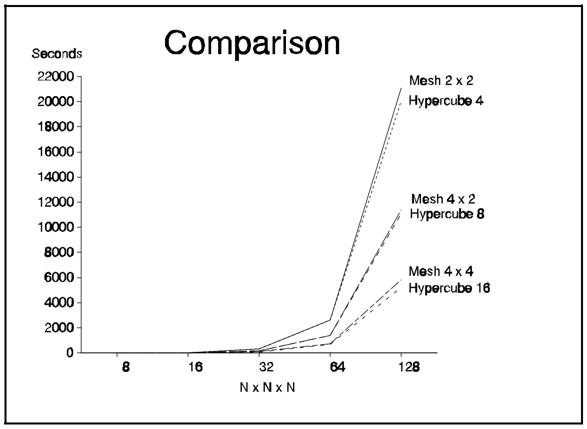

Figure 7.- Processing times in mesh and hypercube.

need two steps for their implementation. In large meshes, these communications will be slower as the PEs that need to communicate are farther away from each other. When this happens a toroidal distribution can be used so that this communications do not penalize the process too much. The rest of the algorithms (see figures 2 and 3), which are the ones that exhibit greater algorithmic complexity, do not require communications and present identical behaviors in both topologies. We can thus conclude that in our parallel algorithm, the mesh topology is better than the hypercube as it presents lower hardware cost and similar parallel performance.

References

[1] Radon, J., *"Uber die Bestimmung von Funktionen durch ihre Integralwerte langs gewisser Mannigfaltigkeiten"*, Ber. Verh. Konig. Sachs. Ges. Wiss. Leipzig, Math. Phys. Kl. 69, 262, 1917.

[2] Hoppe, W., Schramm, H.J., Sturm, M., Hunsmann, N., y Gabmann, J., "*Three dimensional electron microscopy of individual biological objects".* I. Methods, Z. Naturforsch. Teil A. 31. 645, 1976.

[3] Vainshtein, B.K., *"Finding the structure of objects from projections".* Sov. Phys. Crystallogr., 15, 781, 1980.

[4] Hader, D.P *"Image Analysis in Biology"*. CRC Press, 1992.

[5] Pena, T.F., *"Retroproyección filtrada de imágenes en computadores hipercubo"*. Memoria de Licenciatura. Fac. de Física. Univ. Santiago, 1990 (en castellano).

[6] Radermacher, M., Verschoor, A., Wagenknecht,T. and Frank, J. *"Three-dimensional reconstruction from a single exposure, ramdom conical tilt series applied to the 50S ribosomal subunit of escherichia coli"*. J.Microscopy, Vol.146.

[7] Frank, J., Verschoor, A., Wagenknecht, T. *"Computer processing of electron microscopy images of simple molecules"*, in: T.T. Wu, ed., New Methodologies in Studies of Protein Configurations, Van Nostrand Reinhold, New York, 1985, pp.36-89.

[8] Zapata,E.L., Rivera,F.F. and O.G.Plata, *"On the partition of algorithms into hypercubes"*, en: Advances of Parallel Computing, D.J.Evans, Ed. JAI Press, 1990, pp. 149-171.

[9] Rivera, F.F. *"Partición y proyección de algoritmos en computadores hipercubo: reconocimiento de formas"*, Tesis Doctoral. Facultad de Física. Univ. de Santiago de Compostela, España, 1990. (en castellano).

[10] Zapata, E.L., Rivera, F.F., Benavides, J.J., Carazo, J.M. and Peskin, R. *"Multidimensional fast Fourier transform into SIMD hypercubes"*. IEE Proceedings Part E: Computers and Digital Techniques, Vol.137, No.4, pp.253-260, 1990.