Shaded molecular surface graphics on a highly parallel computer

R E Hubbard and D Fincham*

Department of Chemistry and Computing Service, University of York, Heslington, York YO1 5DD, UK
* DAP Support Unit, Queen Mary College, University of London, Mile End Road, London E1 4NS, UK

A highly parallel computer can be used to generate shaded surface molecular pictures with full hidden surface removal, and the paper discusses how this was done on the ICL Distributed Array Processor. The algorithm is characterized as a blocked depth-buffer. The picture is built out of precomputed primitives. Features of the algorithm may be generally applicable; on the DAP it gives a performance of 0.8 ms/atom.

Keywords: computer graphics, molecular structure, computer applications, shaded surface graphics

received 2 January 1984 revised 2 October 1984

Over the past few years a number of techniques have been developed for the representation of molecular surfaces by computer graphics. These can be divided into two groups: calligraphic and raster.

On calligraphic (or vector) equipment, dot surfaces[1,2] have proved extremely informative in allowing a description of molecular surfaces to be manipulated in real time. They provide an artificial representation of the molecule by defining a 'boundary' but have the advantage of revealing the underlying structure (see, for example, the diagrams in Reference 2).

Raster devices allow a more realistic approximation of CPK shaded sphere models to be displayed[3,4]. However, these diagrams require considerable calculation on conventional computer equipment (PDP11, VAX) and require special algorithms and machine code programming to achieve reasonable performance[4]. Also included in this category is the matrix printer representation of Tickle *et al*[5].

In the present paper, the use of the ICL Distributed Array Processor (DAP) to produce raster-graphic images of molecular models is described. An algorithm is presented for calculating a space-filling CPK model of a molecule. This algorithm represents the first attempt to draw molecules on this type of processor.

The architecture of the ICL DAP attached to the ICL 2980 at Queen Mary College, London, is described in detail elsewhere[6]. For the purposes of this paper, however, it is necessary to briefly describe the major features of the DAP and DAP FORTRAN, the language used to program it.

DAP

The essential feature of the DAP is an array of 64 × 64 single bit processors (4 096 in all). The DAP is a truly parallel computer in that the array of processors execute the same instruction simultaneously. To achieve maximum performance, therefore, calculations need to be constructed from truly parallel algorithms. These are coded in DAP FORTRAN, which is a superset of FORTRAN IV. DAP FORTRAN allows the constructs of matrix and vector variables and their manipulation. A matrix is an array of 64 × 64 elements; a vector an array of 64 elements. It is not necessary to process all elements of an array because individual elements may be masked by use of logical arrays. There is also a powerful library of matrix and vector functions and subroutines. As an example of DAP FORTRAN, consider a matrix Z that contains 4 096 elements (64 × 64). To set all negative elements of Z to zero requires just one line of code:

```
Z(Z.LT.0) = 0
```

Here a logical matrix expression is used, rather than the subscripts of conventional FORTRAN, to select particular elements where processing is to be carried out.

Since the DAP processors are single-bit, all arithmetic is provided in software. This enables maximum computational performance to be achieved by use of arithmetic of no more than the required precision. The DAP FORTRAN language itself provides integer arithmetic in byte increments from 1 to 8 bytes, and real arithmetic from 3 to 8 bytes. Examples of the use of INTEGER*1, INTEGER*2 and REAL*3 arithmetic are shown below.

ALGORITHM

The technique used to calculate shaded surface pictures on the DAP can be categorized as a blocked depth-buffer algorithm. A depth buffer is a record, for each pixel, of the current position of the front surface of the picture. This is continually updated as the composite molecular picture is built up from its component atoms. The calculation of the picture is carried out in blocks, each 64 × 64 pixels square. In the following detailed description of the algorithm, examples are included of some of the DAP FORTRAN code used in order to demonstrate how straightforward the algorithm and its implementation are.

Step 1. The initial stage of picture production is for atom types and coordinates to be read, and for each atom to be associated with a radius and a colour to scale the data. At present, pictures of 512 × 512 pixels resolution are generated. Data input from disc is performed by the 2980, and the atomic coordinates and atom types read into 64 × 64 matrices. All subsequent calculation is then on the DAP.

Step 2. After the scale of the picture has been calculated, shaded sphere pictures are generated in DAP matrices, one for each distinct atomic radius for the atoms in the molecule. These form the basic graphical objects which are used in picture generation. The calculation of these objects provides an interesting example of the power of DAP FORTRAN. The subroutine to calculate a shaded sphere is:

```
1 SUBROUTINE SPHEREGEN (IRAD,
    SPHEREZ, SPHEREI)
2 INTEGER*1 IRAD, SPHEREZ(,), SPHEREI(,)
```

This subroutine generates a 64 × 64 pixel shaded sphere of radius IRAD

```
3  REAL*3 Z(,)
4  INTEGER*2 SQUARES (,), INDEXVEC2(,),
     IRADSQ
5  INTEGER INDEXVEC ( )
6  LOGICAL SPHERE (,)
7  IRADSQ=IRAD**2
8  CALLXSHORTINDEX (INDEXVEC,-31)
9  INDEXVEC2=INDEXVEC**2
10 SQUARES=MATC(INDEXVEC2)
     +MATR(INDEXVEC2)
11 SPHERE=SQUARES.LT.IRADSQ
12 Z=0.0
13 Z(SPHERE)=SQRT(FLOAT(IRADSQ-
     SQUARES))
14 SPHEREI=0
15 SPHEREI(SPHERE)=(16.0*Z/IRAD)-0.5
16 SPHEREZ=Z+0.5
17 RETURN
18 END
```

The DAP library subroutine X05SHORTINDEX (line 8) generates a vector containing the sequence -31 to 32. This vector is then used to produce a matrix, SQUARES (line 10), the elements of which are the squares of the distance of that element from the centre of the matrix (which is at element (32,32)). This is achieved by MATC (vector), which produces a matrix in which all the columns are the same, and MATR, which produces a matrix in which all the rows are the same.

In line 11, a logical mask matrix SPHERE is produced which contains .TRUE. for every element of SQUARES that is less than IRADSQ. This produces the shape of the sphere, that is, a .TRUE. wherever the distance from the centre is less than the radius. This mask is then used in lines 13 and 15 to generate matrices containing the *Z* height and intensity of the pixels (matrix elements) that make up the sphere. The intensity for this simple sphere is calculated for front illumination as per Porter[7]. The *Z* height for the sphere is the height of each pixel element above the *XY* plane for a sphere with origin in the *XY* plane.

Having produced these graphical primitives, the picture is then assembled by shifting the appropriate object to the correct place and adding it to the picture. In the simplest implementation of the algorithm, the 512 × 512 picture is considered in 64 × 64 blocks (see Colour plate 1). For each block, the following stages are required.

Step 3. A block *Z* height and block intensity matrix are used to build up the picture. These are initialized to zero.

Step 4. A calculation is performed to determine which atom spheres will fall within this block.

Step 5. A calculation is performed to determine the shifts required to move the spheres representing each atom to the correct position relative to the block coordinates.

Steps 4 and 5 are performed in parallel on the data array to produce a mask identifying which atom appears in the block and two matrices containing the *x* and *y* shifts for each atom.

Step 6. Each atom is then added to the picture in turn. For each atom:

(i) A copy of the appropriate sphere *Z* height and intensity matrices is shifted to the correct position for the atom, to create atom *Z* height and intensity matrices. This is done using DAP shift functions, which are very fast.

(ii) The true atom centre *Z* height is added to the atom *Z* height matrix and the atom colour to the atom intensity matrix.

(iii) The atom *Z* height matrix is compared to the block *Z* height matrix. Where the *Z* height is greater than the block *Z* height, the latter is updated and the equivalent element of the block intensity matrix is updated with the atom intensity matrix. This is performed in DAP FORTRAN by the following code:

```
MASK = ATOM_Z.GT.BLOCK_Z
BLOCK_Z(MASK) = ATOM_Z
BLOCK_INTENSITY(MASK) = ATOM_
  INTENSITY
```

where MASK is a logical matrix and all other variables are integer matrices.

Steps 6 (i) to (iii) are repeated for each atom in a block. Steps 3 to 6 are repeated for each block to produce a complete picture.

Colour plate 1 shows one stage in the calculation of a complete figure. On the left can be seen the partly completed figure with the blocks still to be calculated outlined. The three small pictures on the right show the partial generation of the next 64 × 64 block. The top square contains the shaded sphere that is to be added to the picture, the middle square the sphere shifted to the correct position, and the lower square the partially complete block to which this shifted sphere has been added.

RESULTS AND DISCUSSION

Colour plates 2 and 3 show completed figures giving views of the haem groups and other selected atoms in the tetramer of haemoglobin Iowa[8] and the insulin dimer[9]. The algorithm produces shaded space filling pictures of good quality with correct hidden line removal and intersection between bonded atoms.

The calculational time is typically (7 + 25 + 0.8n) ms for n atoms. The first constant, 7 ms, is the time for initial setting up and precalculation of the shaded spheres. These calculations would not need to be repeated to rotate an existing picture. The 25 ms constant includes 4 ms to perform a general transformation on the coordinates, for up to 4 096 atoms. It should be emphasised that these extremely fast times (a 1 000 atom picture in less than 1 s) were obtained at the first attempt using a high level language. The authors believe that they convincingly demonstrate the power of the DAP architecture for three-dimensional raster graphics.

There are certainly possibilities for further performance improvement. The depth buffering uses INTEGER*2 (.e. 16-bit) arithmetic. Since Z-values are stored with the same precision as x and y, with values between 0 and 511, only nine of these bits are required. It would be possible to write appropriate 9-bit arithmetic routines in DAP FORTRAN, but this is a situation in which it might be more effective to use assembler level programming. It would be expected that the time could be halved as a result.

The algorithm has a number of interesting features. In this case, a sphere with front illumination and diffuse reflection was chosen as the basic graphical object, but other representations using different shading algorithms could be implemented by replacing the subroutine SPHEREGEN. The generation of the shaded spheres is a once-only calculation for the whole picture, so quite sophisticated shading and texturing models could be used without seriously affecting the speed at which pictures are generated.

It should be noted that X and Y clipping are performed automatically; if a sphere lies outside the picture or lies across the boundary of the picture, then it is treated correctly. Z clipping, that is removing parts of the picture that are above or below certain Z values, is also straightforward. The back Z plane or yon clipping is controlled by the value used to initialize the block Z height matrices. Hither or front plane Z clipping would require extra calculation.

The algorithm described above is particularly suitable and easy to implement for the DAP. The obvious connection between a processor array architecture and the pixel array of a raster screen leads one to expect that powerful graphics processors based on similar principles will appear on the market. However, there are also some general lessons to be learned from the authors' work that are independent of the type of computer architecture used.

First, it should be noticed that, typically, large molecules consist of a limited number of atom types, and so shaded sphere pictures can be built up from a limited number of primitives. Furthermore, the spherical symmetry means that these spheres can be shifted into position without rotation. Many modern graphics processors have hardware support for such an operation, often called Raster Op, and so this technique might be more widely used.

Second, it should be emphasized that the depth-buffer technique can be used for building up surface pictures from solid components. All intersections and hidden surface removal are carried out automatically, and the execution time is only linear in the number of components. The technique has had limited popularity because of the large storage required to hold a depth buffer for every pixel in the picture. A blocked form of the depth buffer was adopted because it was particularly easy to implement on the DAP, but it also has the advantage of requiring less storage and could perhaps be more generally used. In fact, it has been implemented successfully on a Perq minicomputer.

It is easy to extend the techniques described to more complicated pictures. For example, the present authors have developed a shaded cylinder representation for bonds. Although these must be calculated individually, rather than precomputed, the calculation is still fast, about 10 ms/bond.

This work has used the DAP at Queen Mary College. There is at present no fast graphical display attached to this device, so it has not been possible to explore interactive graphics using the DAP. However, the experiments presented in this paper suggest that if computers of the DAP type were incorporated into the correct environment, such a device would provide a flexible and powerful raster graphics processor capable of real-time manipulation of realistic models of molecular structures. This, combined with the exceptional speed of the DAP as a calculating machine[10], could provide an exciting workstation for interactive model building systems based on high quality graphics representations linked to substantial real time calculations of molecular properties.

ACKNOWLEDGEMENT

The authors would like to thank the SERC, ICL and the Wellcome Trust for financial support.

REFERENCES

1 **Pearl, L H and Honegger, A M** 'Generation of molecular surfaces for graphic display' *J. Mol. Graph.* Vol 1 No 1 (March 1983) pp 9–12

2 **Langridge, R, Ferrin, T E, Kuntz, I D and Connolly, M L** 'Real-time color graphics in studies of molecular interactions' *Science* Vol 211 (1981) pp 661–666

3 **Porter, T K** 'The shaded surface display of large molecules' *Proc. Siggraph Conf.* (1979) pp 234–236

4 **Feldmann, R J, Bing, D H, Furie, B C and Furie, B** *Proc. Nat. Acad. Sci. USA* Vol 75 (1978) pp 5409–5412

5 **Tickle, I J, Borkakoti, N, Moss, D S and Palmer, R A** 'Colour stereo space-filling representations of ribonuclease-mononucleotide interactions' *J. Mol. Graph.* Vol 1 No 3 (September 1983) pp 68–70

6 **Parkinson, D** 'The Distributed Array Processor (DAP)' *Comp. Phys. Comm.* Vol 28 (1983) pp 325–336

7 **Porter, T K** 'Spherical shading' *Proc. Siggraph Conf.* (1978) pp 282–285

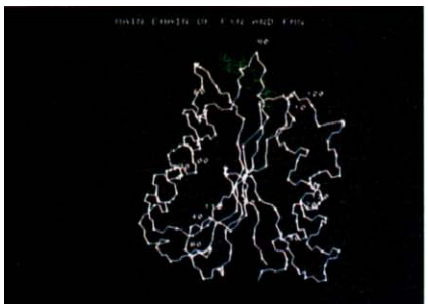
8 **Arnone, A** Personal communication

9 **Dodson, E J, Dodson, G G, Hodgkin, D C and Reynolds, C D** 'Structural relationships in the 2Zn insulin hexamer' *Can. J. Biochem.* Vol 57 (1979) pp 469–479

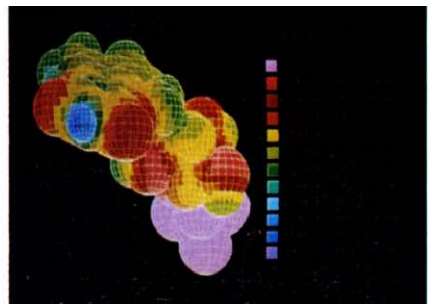
10 **Pawley, G S and Dove, M T** 'Molecular dynamics on a parallel computer' *Helv. Phys. Acta.* Vol 56 (1983) pp 583–592

H Nakamura, K Komatsu, S Nakagawa and H Umeyama

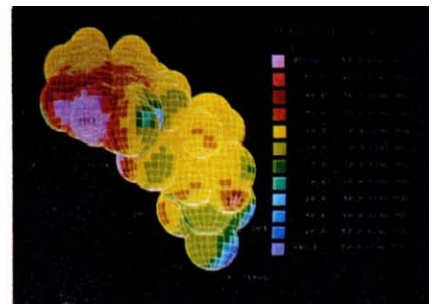
Visualization of electrostatic recognition by enzymes for their ligands and cofactors



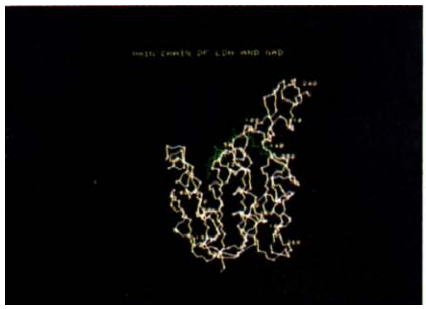
Colour plate 1a. Main chain of FXN (white) and FMN (green)



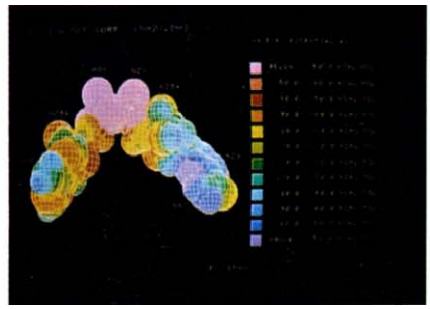
Colour plate 1b. G–on–G potential surface, where FMN is the guest molecule



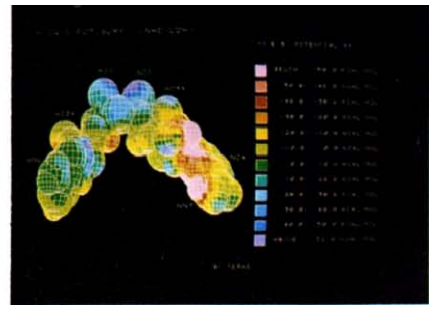
Colour plate 1c. H–on–G potential surface, where FXN is the host enzyme and FMN is the guest molecule



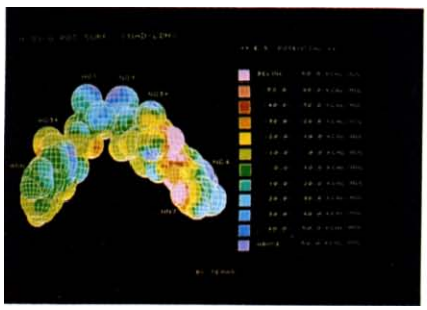
Colour plate 2a. Main chain of LDH-domain 1 (white) and NAD+ (green)



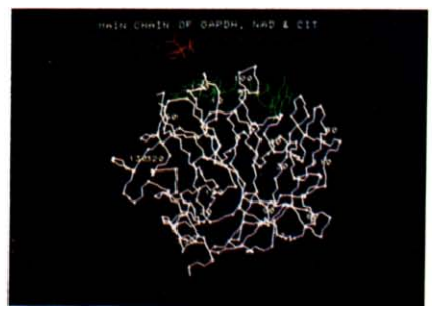
Colour plate 2b. G–on–G potential surface, where NAD+ is the guest molecule



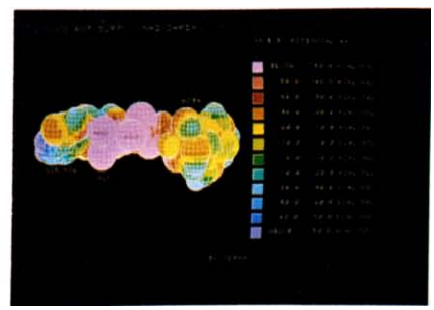
Colour plate 2c. H–on–G potential surface, where LDH with neutral His195 is the host enzyme and NAD+ is the guest molecule



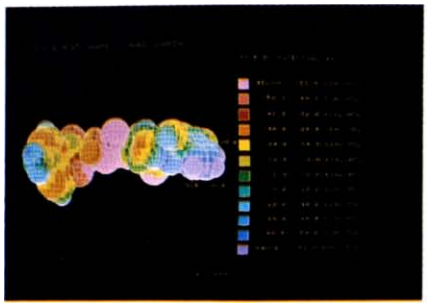
Colour plate 2d. H–on–G potential surface, where LDH with protonated His195 is the host enzyme



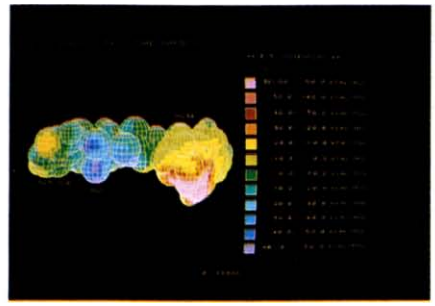
Colour plate 3a. Main chain of GAPDH-domain 1 (white), NAD+ (green) and citrate (red)



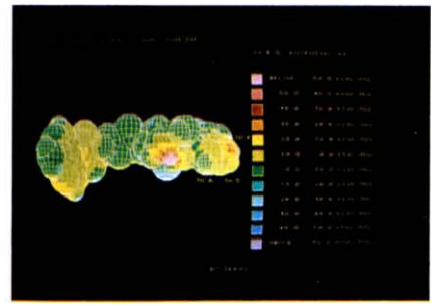
Colour plate 3b. G–on–G potential surface, where NAD+ is the guest molecule



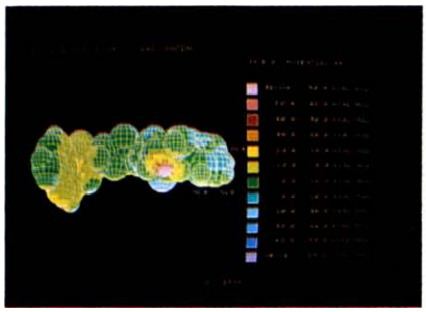
Colour plate 3c. Same G–on–G potential surface as 3b, viewed from the reverse side



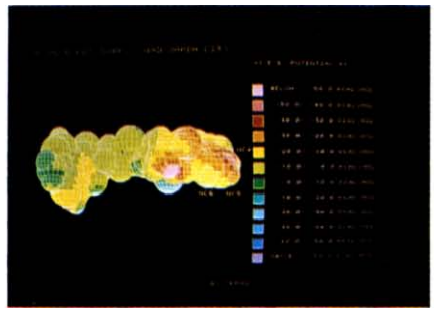
Colour plate 3d. H–on–G potential surface, where GAPDH without citrate is the host enzyme and NAD+ is the guest molecule, respectively, and both Cys149 and His176 are ionized



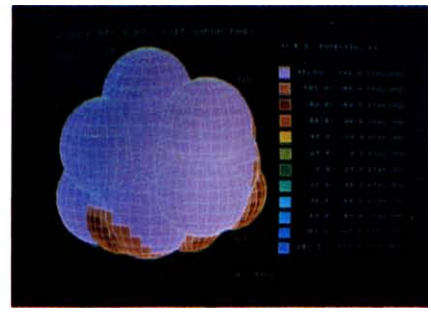
Colour plate 3e. Same H–on–G potential surface as 3d, viewed from the reverse side



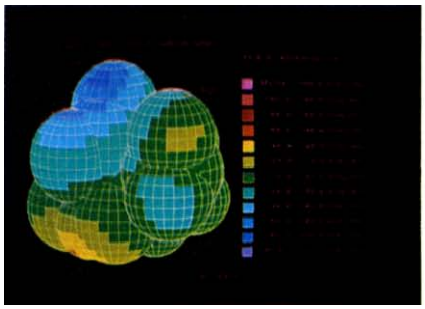
Colour plate 3f. H–on–G potential surface, where GAPDH with neutral Cys149 and His176 excluding citrate is the host enzyme, viewed from the same side of 3e



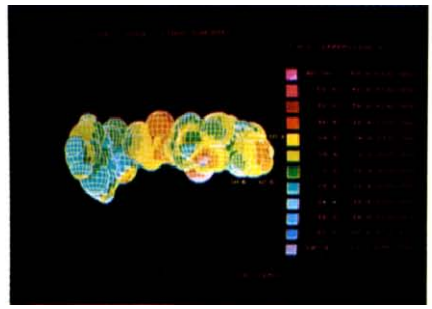
Colour plate 3g. H–on–G potential surface, where GAPDH with ionized Cys149 and His176 including citrate is the host enzyme



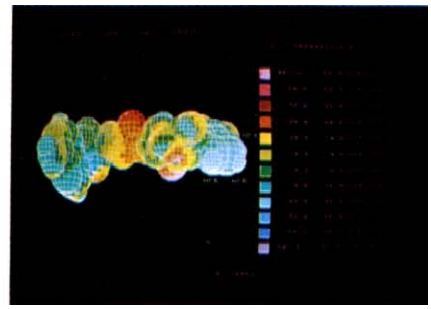
Colour plate 3h. G–on–G potential surface, where citrate is the guest molecule



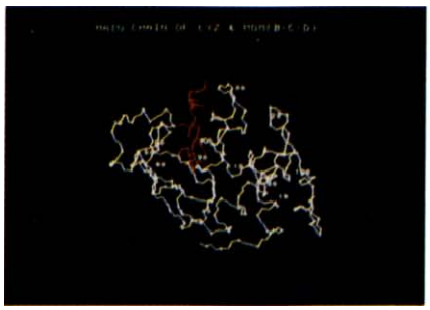
Colour plate 3i. H–on–G potential surface, where GAPDH with ionized Cys149 and His176 including NAD+ is the host enzyme and citrate is the guest molecule



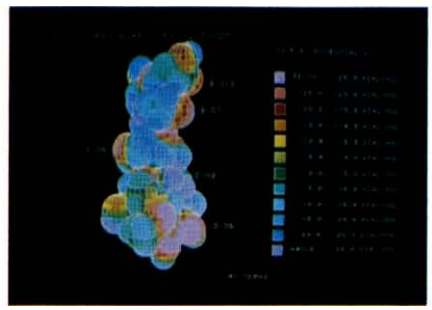
Colour plate 3j. G–H correlation surface, where the correlation was taken between colour plates 3c and 3e



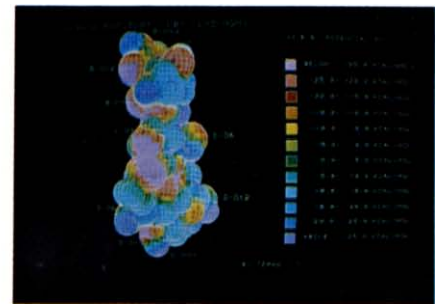
Colour plate 3k. G–H correlation surface, where the correlation was taken between colour plates 3c and 3f



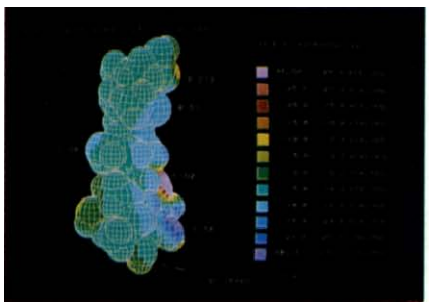
Colour plate 4a. Main chain of LYZ (white) and MGM (B–C–D) (red)



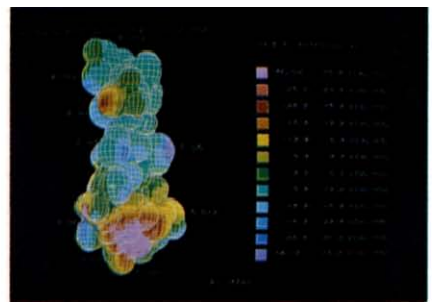
Colour plate 4b. G–on–G potential surface viewed from outside the crevice, where MGM is the guest molecule



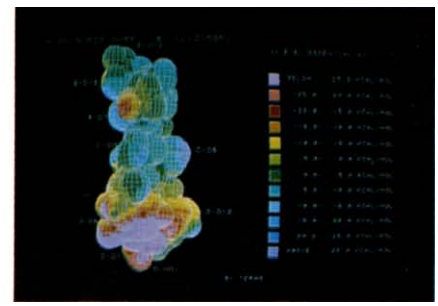
Colour plate 4c. The same G–on–G potential surface as 4b, viewed from inside the crevice



Colour plate 4d. H–on–G potential surface viewed from outside the crevice, where LYZ with undissociated Glu35 is the host enzyme and MGM is the guest molecule



Colour plate 4e. The same H–on–G potential surface as 4d, viewed from inside the crevice

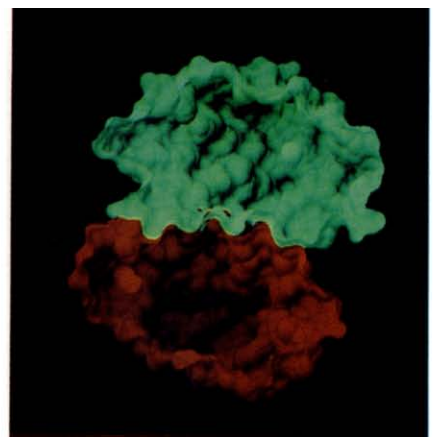


Colour plate 4f. H–on–G potential surface viewed from inside the crevice, where LYZ with dissociated Glu35 is the host enzyme

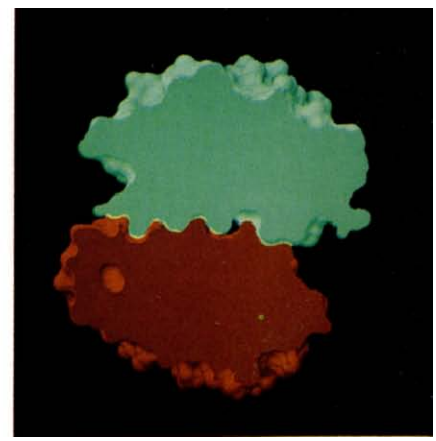
Michael L Connolly

Depth-buffer algorithms for molecular modelling

Colour plate 1. Surface of superoxide dismutase enzyme. The shading is determined by the depth of the surface

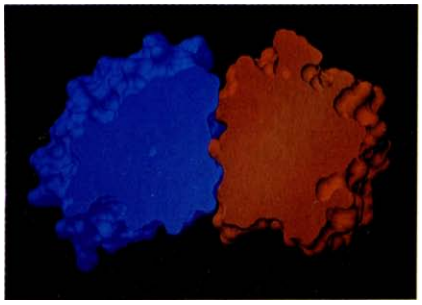


Colour plate 2. Haemoglobin dimer. Interface between alpha-1 (red) ane beta-1 (green) subunits. The surfaces are clipped and hollow

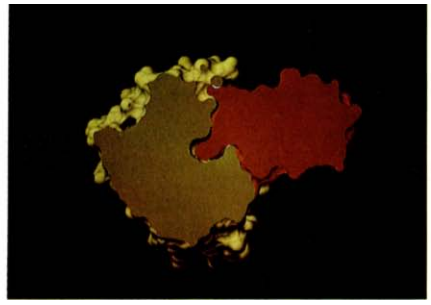


Colour plate 3. The interface as Colour plate 2, but with solid interiors

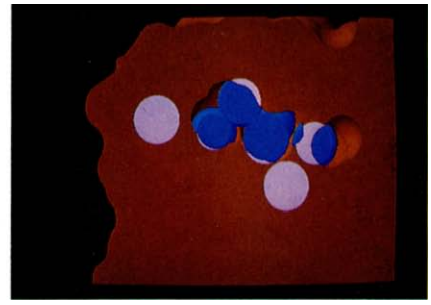
Michael L Connolly — *continued*



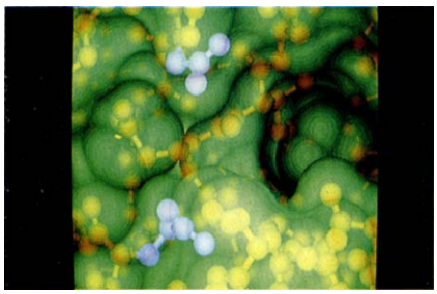
Colour plate 4. Haemoglobin interface between alpha-1 (red) and beta-2 (blue) units



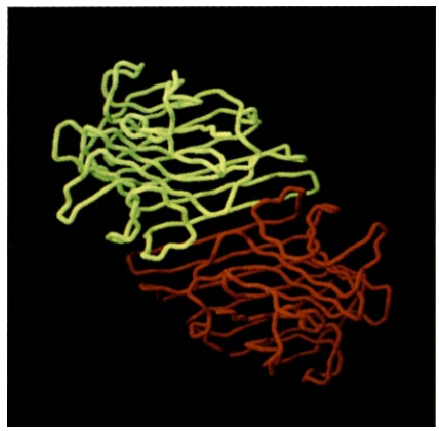
Colour plate 5. Complex between trypsin (brown) and its inhibitor (red). The molecules are solid and clipped. Overlap volumes are white



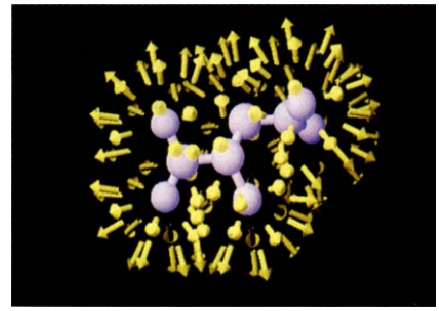
Colour plate 6. Trypsin water cluster. The waters are blue, trypsin is red. Overlapping volumes are coloured pale levender. Two waters are totally buried



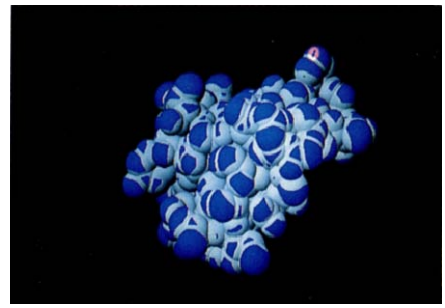
Colour plate 7. Translucent surface of the active site of lysozyme. Glu 35 and Asp 52 are highlighted



Colour plate 8. Concanavalin A dimer. The alpha-carbon chains are represented by tubes



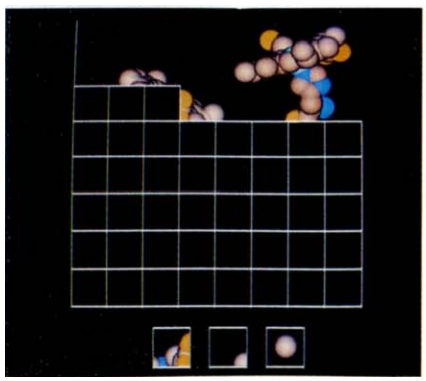
Colour plate 9. Surface normal vectors of asparagine amino acid represented as arrows



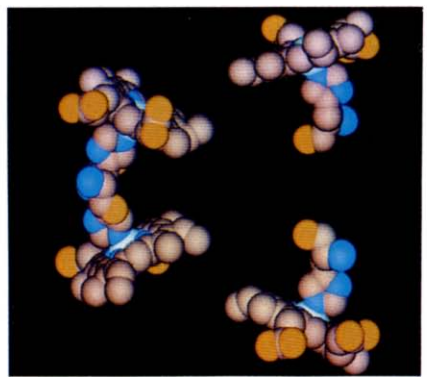
Colour plate 10. Van der Waals surface of crambin with contact surface coloured dark blue

R E Hubbard and D Fincham

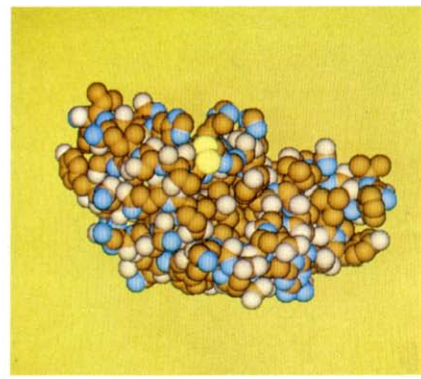
Shaded molecular surface graphics on highly parallel computer



Colour plate 1. Partly completed figure. The square grid outlines the 64 × 64 blocks in which the picture is calculated



Colour plate 2. Completed figure showing view of some atoms in haemoglobin iowa tetramer



Colour plate 3. Insulin dimer generated on the DAP at Queen Mary College, London and displayed on the PERQ/Metheus system at York