International Tables for Crystallography (2006). Vol. B, Section 3.3.1, pp. 360–376.

3.3. Molecular modelling and graphics

BY R. DIAMOND

3.3.1. Graphics

3.3.1.1. *Coordinate systems, notation and standards*

3.3.1.1.1. *Cartesian and crystallographic coordinates*

It is usual, for purposes of molecular modelling and of computer graphics, to adopt a Cartesian coordinate system using mutually perpendicular axes in a right-handed system using the ångström unit or the nanometre as the unit of distance along such axes, and largely to ignore the existence of crystallographic coordinates expressed as fractions of unit-cell edges. Transformations between the two are thus associated, usually, with the input and output stages of any software concerned with modelling and graphics, and it will be assumed after this section that all coordinates are Cartesian using the chosen unit of distance as the unit of coordinates. For a discussion of coordinate transformations and rotations without making this assumption see Chapter 1.1 in which formulations using co- and contravariant forms are presented.

The relationship between these systems may be written

$$\mathbf{X} = \boldsymbol{M}\mathbf{x} \quad \mathbf{x} = \boldsymbol{M}^{-1}\mathbf{X}$$

in which $\mathbf{X}$ and $\mathbf{x}$ are position vectors in direct space, written as column vectors, with $\mathbf{x}$ expressed in crystallographic fractional coordinates (dimensionless) and $\mathbf{X}$ in Cartesian coordinates (dimension of length).

There are two forms of $\boldsymbol{M}$ in common use. The first of these sets the first component of $\mathbf{X}$ parallel to $\mathbf{a}^*$ and the third parallel to $\mathbf{c}$ and is

$$\boldsymbol{M} = \begin{pmatrix} a\varphi/\sin\alpha & 0 & 0 \\ a(\cos\gamma - \cos\alpha\cos\beta)/\sin\alpha & b\sin\alpha & 0 \\ a\cos\beta & b\cos\alpha & c \end{pmatrix}$$

$$\boldsymbol{M}^{-1} = \begin{pmatrix} \sin\alpha/a\varphi & 0 & 0 \\ (\cos\alpha\cos\beta - \cos\gamma)/b\varphi\sin\alpha & 1/b\sin\alpha & 0 \\ (\cos\alpha\cos\gamma - \cos\beta)/c\varphi\sin\alpha & -1/c\tan\alpha & 1/c \end{pmatrix}$$

in which

$$\begin{aligned}\varphi &= \sqrt{1 - \cos^2\alpha - \cos^2\beta - \cos^2\gamma + 2\cos\alpha\cos\beta\cos\gamma} \\ &= \sin\alpha\sin\beta\sin\gamma^*.\end{aligned}$$

φ is equal to the volume of the unit cell divided by abc, and is unchanged by cyclic permutation of α, β and γ and of α^*, β^* and γ^*. The Cartesian and crystallographic axes have the same chirality if the positive square root is taken.

The second form sets the first component of $\mathbf{X}$ parallel to $\mathbf{a}$ and the third component of $\mathbf{X}$ parallel to $\mathbf{c}^*$ and is

$$\boldsymbol{M} = \begin{pmatrix} a & b\cos\gamma & c\cos\beta \\ 0 & b\sin\gamma & c(\cos\alpha - \cos\beta\cos\gamma)/\sin\gamma \\ 0 & 0 & c\varphi/\sin\gamma \end{pmatrix}$$

$$\boldsymbol{M}^{-1} = \begin{pmatrix} 1/a & -1/a\tan\gamma & (\cos\alpha\cos\gamma - \cos\beta)/a\varphi\sin\gamma \\ 0 & 1/b\sin\gamma & (\cos\beta\cos\gamma - \cos\alpha)/b\varphi\sin\gamma \\ 0 & 0 & \sin\gamma/c\varphi \end{pmatrix}.$$

A third form, suitable only for rhombohedral cells, is

$$\boldsymbol{M} = \frac{a}{3}\begin{pmatrix} p+2q & p-q & p-q \\ p-q & p+2q & p-q \\ p-q & p-q & p+2q \end{pmatrix}$$

$$\boldsymbol{M}^{-1} = \frac{1}{3a}\begin{pmatrix} \frac{1}{p}+\frac{2}{q} & \frac{1}{p}-\frac{1}{q} & \frac{1}{p}-\frac{1}{q} \\ \frac{1}{p}-\frac{1}{q} & \frac{1}{p}+\frac{2}{q} & \frac{1}{p}-\frac{1}{q} \\ \frac{1}{p}-\frac{1}{q} & \frac{1}{p}-\frac{1}{q} & \frac{1}{p}+\frac{2}{q} \end{pmatrix}$$

in which

$$p = \pm\sqrt{1 + 2\cos\alpha} \quad q = \pm\sqrt{1 - \cos\alpha},$$

which preserves the equivalence of axes. Here the chiralities of the Cartesian and crystallographic axes are the same if p is chosen positive, and different otherwise, and the two sets of axes coincide in projection along the triad if q is chosen positive and are π out of phase otherwise.

3.3.1.1.2. *Homogeneous coordinates*

Homogeneous coordinates have found wide application in computer graphics. For some equipment their use is essential, and they are of value analytically even if the available hardware does not require their use.

Homogeneous coordinates employ four quantities, X, Y, Z and W, to define the position of a point, rather than three. The fourth coordinate has a scaling function so that it is the quantity X/W (as delivered to the display hardware) which controls the left–right positioning of the point within the picture. A point with $|X/W| < 1$ is in the picture, normally, and those with $|X/W| > 1$ are outside it, but see Section 3.3.1.3.5.

There are many reasons why homogeneous coordinates may be adopted, among them the following:

(i) X, Y, Z and W may be held as integers, thus enabling fast arithmetic whilst offering much of the flexibility of floating-point working. A single W value may be common to a whole array of X, Y, Z values.

(ii) Perspective transformations can be implemented without the need for any division. Only high-speed matrix multiplication using integer arithmetic is necessary, provided only that the drawing hardware can provide displacements proportional to the ratio of two signals, X and W or Y and W. Rotation, translation, scaling and the application of perspective are all affected by operations of the same form, namely multiplication of a four-vector by a 4×4 matrix. The hardware may thus be kept relatively simple since only one type of operation needs to be provided for.

(iii) Since kX, kY, kZ, kW represents the same point as X, Y, Z, W, the hardware may be arranged to maximize resolution without risk of integer overflow.

For analytical purposes it is convenient to regard homogeneous transformations in terms of partitioned matrices

$$\begin{pmatrix} \boldsymbol{M} & \mathbf{V} \\ \mathbf{U} & N \end{pmatrix}\begin{pmatrix} \mathbf{X} \\ W \end{pmatrix},$$

where $\boldsymbol{M}$ is a 3×3 matrix, $\mathbf{V}$ and $\mathbf{X}$ are three-element column vectors, $\mathbf{U}$ is a three-element row vector and N and W are scalars.

Matrices and vectors which are equivalent under the considerations of (iii) above will be related by the sign $\simeq$ in what follows.

Hardware systems which use true floating-point representations have less need of homogeneous coordinates and for these N and W may normally be set to unity.

3.3.1.1.3. *Notation*

In this chapter the conventions of matrix algebra will be adhered to except where it is convenient to show operations on elements of vectors, matrices and tensors, where a subscript notation will be used with a modified summation convention in which summation is over lower-case subscripts *only*. Thus the equation

$$x_I = A_{Ij}X_j$$

is to be read 'For any I, x_I is $A_{Ij}X_j$ summed over j'.

Subscripts using the letter i or later in the alphabet will relate to the usual three dimensions and imply a three-term summation. Subscripts a to h are not necessarily so limited, and, in particular, the subscript a is used to imply summation over atoms of which there may be an arbitrary number.

We shall use the superscript T to denote a transpose, and also use the Kronecker delta, δ_{IJ}, which is 1 if $I = J$ and zero otherwise, and the tensor ε_{IJK} which is 1 if I, J and K are a cyclic permutation of 1, 2, 3, -1 if an anticyclic permutation, and zero otherwise.

$$\varepsilon_{IJK} = (I - J)(J - K)(K - I)/2 \quad 1 \leq I, J, K \leq 3.$$

A useful identity is then

$$\varepsilon_{iJK}\varepsilon_{iLM} = \delta_{JL}\delta_{KM} - \delta_{JM}\delta_{KL}.$$

Single modulus signs surrounding the symbol for a square matrix denote its determinant, and around a vector denote its length.

The symbol $\simeq$ is defined in the previous section.

3.3.1.1.4. *Standards*

The sections of this chapter concerned with graphics are primarily concerned with the mathematical aspects of graphics programming as they confront the applications programmer. The implementations outlined in the final section have all, so far as the author is aware, been developed *ab initio* by their inventors to deal with these aspects using their own and unrelated techniques and protocols. It is clear, however, that standards are now emerging, and it is to be hoped that future developments in applications software will handle the graphics aspects through one or other of these standards.

First among these standards is the Graphical Kernel System, GKS, defined in *American National Standards Institute, American National Standard for Information Processing Systems – Computer Graphics – Graphical Kernel System (GKS) Functional Description* (1985) and described and illustrated by Hopgood *et al.* (1986) and Enderle *et al*. (1984). GKS became a full International Standards Organization (ISO) standard in 1985, and its purpose is to standardize the interface between application software and the graphics system, thus enhancing portability of software. Specifications for Fortran, Pascal and Ada formulations are at an advanced stage of development. Its value to crystallographers is limited by the fact that it is only two-dimensional. A three-dimensional extension known as GKS-3D, defined in *International Standards Organisation, International Standard Information Processing Systems – Computer Graphics – Graphical Kernel System for Three Dimensions (GKS-3D), Functional Description* (1988) became an ISO standard in 1988. Perhaps of greatest interest to crystallographers, however, is the Programmers' Hierarchical Interactive Graphics System (PHIGS) (Brown, 1985; Abi-Ezzi & Bunshaft, 1986) since this allows hierarchical segmentation of picture content to exist in both the applications software and the graphics device in a related manner, which GKS does not. Some graphics devices now available support this type of working and its exploitation indicates the choice of PHIGS. Furthermore, Fortran implementations of GKS and GKS-3D require points to be stored in arrays dimensioned as $X(N)$, $Y(N)$, $Z(N)$ which may be equivalenced (in the Fortran sense) to $XYZ(N, 3)$ but not to $XYZ(3, N)$, which may not be convenient. PHIGS also became an International Standard in 1988: *American National Standards Institute, American National Standard for Information Processing Systems – Computer Graphics – Programmer's Hierarchical Graphics System (PHIGS) Functional Description, Archive File Format, Clear-Text Encoding of Archive File* (1988). PHIGS has also been extended to support the capability of raster-graphics machines to represent reflections, shadows, see-through effects *etc.* in a version known as PHIGS+ (van Dam, 1988).

Increasingly, manufacturers of graphics equipment are orienting their products towards one or other of these standards. While these standards are not the subject of this chapter it is recommended that they be studied before investing in equipment.

In addition to these standards, related standards are evolving under the auspices of the ISO for defining images in a file-storage, or metafile, form, and for the interface between the device-independent and device-dependent parts of a graphics package. Arnold & Bono (1988) describe the ANSI and ISO Computer Graphics Metafile standard which provides for the definition of (two-dimensional) images. The definition of three-dimensional scenes requires the use of (PHIGS) archive files.

3.3.1.2. *Orthogonal (or rotation) matrices*

It is a basic requirement for any graphics or molecular-modelling system to be able to control and manipulate the orientation of the structures involved and this is achieved using orthogonal matrices which are the subject of these sections.

3.3.1.2.1. *General form*

If a vector $\mathbf{v}$ is expressed in terms of its components resolved onto an axial set of vectors $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ which are of unit length and mutually perpendicular and right handed in the sense that $(\mathbf{X} \times \mathbf{Y}) \cdot \mathbf{Z} = +1$, and if these components are v_I, and if a second set of axes $\mathbf{X}'$, $\mathbf{Y}'$, $\mathbf{Z}'$ is similarly established, with the same origin and chirality, and if $\mathbf{v}$ has components v'_I on these axes then

$$v'_I = a_{Ij}v_j,$$

in which a_{IJ} is the cosine of the angle between the ith primed axis and the jth unprimed axis. Evidently the elements a_{IJ} comprise a matrix $\boldsymbol{R}$, such that any row represents one of the primed axial vectors, such as $\mathbf{X}'$, expressed as components on the unprimed axes, and each column represents one of the unprimed axial vectors expressed as components on the primed axes. It follows that $\boldsymbol{R}^T = \boldsymbol{R}^{-1}$ since elements of the product $\boldsymbol{R}^T\boldsymbol{R}$ are scalar products among perpendicular unit vectors.

A real matrix whose transpose equals its inverse is said to be *orthogonal.*

Since $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$ can simultaneously be superimposed on $\mathbf{X}'$, $\mathbf{Y}'$ and $\mathbf{Z}'$ without deformation or change of scale the relationship is one of rotation, and orthogonal matrices are often referred to as rotation matrices. The operation of replacing the vector $\mathbf{v}$ by $\boldsymbol{R}\mathbf{v}$ corresponds to rotating the axes from the unprimed to the primed set with $\mathbf{v}$ itself unchanged. Equally, the same operation corresponds to retaining fixed axes and rotating the vector in the opposite sense. The second interpretation is the one more frequently helpful since conceptually it corresponds more closely to rotational operations on objects, and it is primarily in this sense that the following is written.

If three vectors $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{w}$ form the edges of a parallelepiped, then its volume V is

$$V = \mathbf{u} \cdot (\mathbf{v} \times \mathbf{w}) = \varepsilon_{ijk} u_i v_j w_k$$

and if these vectors are transformed by the matrix $\boldsymbol{R}$ as above, then the transformed volume V' is

$$V' = \varepsilon_{lmn} u'_l v'_m w'_n = \varepsilon_{lmn} a_{li} a_{mj} a_{nk} u_i v_j w_k.$$

But the determinant of $\boldsymbol{R}$ is given by

$$|\boldsymbol{R}| \varepsilon_{IJK} = \varepsilon_{lmn} a_{lI} a_{mJ} a_{nK}$$

so that

$$V' = |\boldsymbol{R}| V$$

and the determinant of $\boldsymbol{R}$ must therefore be +1 for a transformation which is a pure rotation. Nevertheless orthogonal matrices with determinant -1 exist though these do not describe a pure rotation. They may always be described as the product of a pure rotation and inversion through the origin and are referred to here as improper rotations. In what follows all references to orthogonal matrices refer to those with positive determinant only, unless stated otherwise.

An important general form of an orthogonal matrix in three dimensions was derived as equation (1.1.4.32) and is

$$\boldsymbol{R} = \begin{pmatrix} l^2 + (m^2 + n^2)\cos\theta & lm(1-\cos\theta) - n\sin\theta & nl(1-\cos\theta) + m\sin\theta \\ lm(1-\cos\theta) + n\sin\theta & m^2 + (n^2 + l^2)\cos\theta & mn(1-\cos\theta) - l\sin\theta \\ nl(1-\cos\theta) - m\sin\theta & mn(1-\cos\theta) + l\sin\theta & n^2 + (l^2 + m^2)\cos\theta \end{pmatrix}$$

or

$$R_{IJ} = (1 - \cos\theta) l_I l_J + \delta_{IJ} \cos\theta - \varepsilon_{IJk} l_k \sin\theta,$$

in which l, m and n are the direction cosines of the axis of rotation (which are the same when referred to either set of axes under either interpretation) and θ is the angle of rotation. In this form, and with $\boldsymbol{R}$ operating on column vectors on the right, the sign of θ is such that, when viewed along the rotation axis *from* the origin *towards* the point *lmn,* the object is rotated clockwise for positive θ with a fixed right-handed axial system. If, under the same viewing conditions, the axes are to be rotated clockwise through θ with the object fixed then the components of vectors in the object, on the new axes, are given by $\boldsymbol{R}$ with the same *lmn* and with θ negated. This is the transpose of $\boldsymbol{R}$, and if $\boldsymbol{R}$ is constructed from a product, as below, then each factor matrix in the product must be transposed and their order reversed to achieve this. Note that if, for a given rotation, the viewing direction from the origin is reversed, l, m, n and θ are all reversed and the matrix is unchanged.

Any rotation about a reference axis such that two of the direction cosines are zero is termed a *primitive rotation*, and it is frequently a requirement to generate or to interpret a general rotation as a product of primitive rotations.

A second important general form is based on Eulerian angles and is the product of three such primitives. It is

$$\begin{aligned} \boldsymbol{R} &= \begin{pmatrix} \cos\varphi_3 & -\sin\varphi_3 & 0 \\ \sin\varphi_3 & \cos\varphi_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\varphi_2 & 0 & \sin\varphi_2 \\ 0 & 1 & 0 \\ -\sin\varphi_2 & 0 & \cos\varphi_2 \end{pmatrix} \begin{pmatrix} \cos\varphi_1 & -\sin\varphi_1 & 0 \\ \sin\varphi_1 & \cos\varphi_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} (\cos\varphi_3\cos\varphi_2\cos\varphi_1 & -(\cos\varphi_3\cos\varphi_2\sin\varphi_1 & \cos\varphi_3\sin\varphi_2 \\ \quad -\sin\varphi_3\sin\varphi_1) & \quad +\sin\varphi_3\cos\varphi_1) & \\ (\sin\varphi_3\cos\varphi_2\cos\varphi_1 & (-\sin\varphi_3\cos\varphi_2\sin\varphi_1 & \sin\varphi_3\sin\varphi_2 \\ \quad +\cos\varphi_3\sin\varphi_1) & \quad +\cos\varphi_3\cos\varphi_1) & \\ -\sin\varphi_2\cos\varphi_1 & \sin\varphi_2\sin\varphi_1 & \cos\varphi_2 \end{pmatrix} \end{aligned}$$

which is commonly employed in four-circle diffractometers for which $\varphi = -\varphi_1$, $\chi = \varphi_2$ and $\omega = -\varphi_3$. In terms of the fixed-axes–moving-object conceptualization this corresponds to a rotation φ_1 about Z followed by φ_2 about Y followed by φ_3 about Z. In the familiar diffractometer example, when $\chi = 0$ the φ and ω axes are both vertical and equivalent. If φ is altered first, then the χ axis is still in the direction of a fixed Y axis, but if ω is altered first it is not. Since all angles are to be rotations about fixed axes to describe a rotating object it follows that it is φ rather than ω which corresponds to φ_1. In general, when rotating parts are mounted on rotating parts the rotation closest to the moved object must be applied first, forming the right-most factor in any multiple transformation, with the rotation closest to the fixed part as the left-most factor, assuming data supplied as column vectors on the right.

Given an orthogonal matrix, in either numerical or analytical form, it may be required to discover θ and the axis of rotation, or to factorize it as a product of primitives. From the first form we see that the vector

$$v_I = \varepsilon_{Ijk} a_{jk},$$

consisting of the antisymmetric part of $\boldsymbol{R}$, has elements $-2\sin\theta$ times the direction cosines l, m, n, which establishes the direction immediately, and normalization using $l^2 + m^2 + n^2 = 1$ determines $\sin\theta$. Furthermore, the trace is $1 + 2\cos\theta$ so that the quadrant of θ is also fixed. This method fails, however, if the matrix is symmetrical, which occurs if $\theta = \pi$. In this case only the direction of the axis is required, which is given by

$$l : m : n = (a_{23})^{-1} : (a_{31})^{-1} : (a_{12})^{-1}$$

for non-zero elements, or $l = \sqrt{\frac{1}{2}(a_{11} + 1)}$ *etc.*, with the signs chosen to satisfy $a_{12} = 2lm$ *etc.*

The Eulerian form may be factorized by noting that $\tan\varphi_1 = -a_{32}/a_{31}$, $\tan\varphi_3 = a_{23}/a_{13}$, $\cos\varphi_2 = a_{33}$. There is then freedom to choose the sign of $\sin\varphi_2$, but the choice then fixes the quadrants of φ_1 and φ_3 through the elements in the last row and column, and the primitives may then be constructed. These expressions for φ_1 and φ_3 fail if $\sin\varphi_2 = 0$, in which case the rotation reduces to a primitive rotation about Z with angle $(\varphi_1 + \varphi_3), \varphi_2 = 0$, or $(\varphi_3 - \varphi_1), \varphi_2 = \pi$.

Eulerian angles are unlikely to be the best choice of primitive angles unless they are directly related to the parameters of a system, as with the diffractometer. It is often more important that the changes to primitive angles should be quasi-linearly related to θ for any small rotations, which is not the case with Eulerian angles when the required rotation axis is close to the X axis. In such a case linearized techniques for solving for the primitive angles will fail. Furthermore, if the required rotation is about Z only $(\varphi_1 + \varphi_3)$ is determinate.

Quasi-linear relationships between θ and the primitive rotations arise if the primitives are one each about X, Y and Z. Any order of the three factors may be chosen, but the choice must then be adhered to since these factors do not commute. For sufficiently small rotations the primitive rotations are then $l\theta$, $m\theta$ and $n\theta$, whilst for larger θ linearized iterative techniques for finding the primitive rotations are likely to be convergent and well conditioned.

The three-dimensional space of the angles φ_1, φ_2 and φ_3 in either case is non-linearly related to θ. In the Eulerian case the worst non-linearities occur at the origin of φ-space. Equally severe non-linearities occur in the quasi-linear case also but are 90° away from the origin and less likely to be troublesome.

Neither of the foregoing general forms of orthogonal matrix has ideally convenient properties. The first is inconvenient because it uses four non-equivalent variables l, m, n and θ, with a linking equation involving l, m and n, so that they cannot be treated as independent variables for analytical purposes. The second form (the product of primitives) is not ideal because the three angles, though independent, are not equivalent, the non-equivalence arising from the non-commutation of the primitive factors. In the remainder of this section we give two further forms of orthogonal matrix which each use three variables which are independent and strictly equivalent, and a third form using four whose squares sum to unity.

The first of these is based on the diagonal and uses the three independent variables p, q, r, from which we construct the auxiliary variables

$$P = \pm\sqrt{1+p-q-r},\ Q = \pm\sqrt{1-p+q-r},$$
$$R = \pm\sqrt{1-p-q+r},\ S = \pm\sqrt{1+p+q+r},$$

then

$$\boldsymbol{R} = \begin{pmatrix} p & \frac{1}{2}[PQ-RS] & \frac{1}{2}[PR+QS] \\ \frac{1}{2}[PQ+RS] & q & \frac{1}{2}[QR-PS] \\ \frac{1}{2}[PR-QS] & \frac{1}{2}[QR+PS] & r \end{pmatrix}$$

is orthogonal with positive determinant for any of the sixteen sign combinations. The signs of P, Q, R and S are, respectively, the signs of the direction cosines of the rotation axis and of $\sin\theta$. Using also $T = \sqrt{4-S^2}$, which may be deemed positive without loss of generality,

$$l = P/T, m = Q/T, n = R/T,\ \sin\theta = ST/2,$$
$$\cos\theta = 1 - T^2/2 = S^2/2 - 1.$$

Although p, q and r are independent, the point $[pqr]$ is bound, by the requirement that P, Q, R and S be real, to lie within a tetrahedron whose vertices are the points [111], $[1\bar{1}\bar{1}]$, $[\bar{1}1\bar{1}]$ and $[\bar{1}\bar{1}1]$, corresponding to the identity and to 180° rotations about each of the axes. The facts that the identity occurs at a vertex of the feasible region and that $(1-\cos\theta)$, rather than $\sin\theta$, is linear on p, q and r in this vicinity make this form suitable only for substantial rotations.

The second form consists in defining a rotation vector $\mathbf{r}$ with components u, v, w such that $u = lt$, $v = mt$, $w = nt$ with $t = \tan(\theta/2)$ and $\mathbf{r}\cdot\mathbf{r} = t^2$. Then the matrix

$$\boldsymbol{R} = \begin{pmatrix} \frac{1+u^2-v^2-w^2}{1+t^2} & \frac{2(uv-w)}{1+t^2} & \frac{2(uw+v)}{1+t^2} \\ \frac{2(uv+w)}{1+t^2} & \frac{1-u^2+v^2-w^2}{1+t^2} & \frac{2(vw-u)}{1+t^2} \\ \frac{2(uw-v)}{1+t^2} & \frac{2(vw+u)}{1+t^2} & \frac{1-u^2-v^2+w^2}{1+t^2} \end{pmatrix}$$

$$R_{IJ} = (1+t^2)^{-1}[\delta_{IJ}(1-u_k u_k) + 2(u_I u_J - \varepsilon_{IJl}u_l)]$$

is orthogonal and the variables u, v, w are independent, equivalent and unbounded, and, unlike the previous form, small rotations are quasi-linear on these variables. As examples, $\mathbf{r} = [100]$ gives 90° about X, $\mathbf{r} = [111]$ gives 120° about [111].

$\boldsymbol{R}$ then transforms a vector $\mathbf{d}$ according to

$$\boldsymbol{R}\mathbf{d} = \mathbf{d} + \frac{2}{1+t^2}\{(\mathbf{r}\times\mathbf{d}) + [\mathbf{r}\times(\mathbf{r}\times\mathbf{d})]\}.$$

Multiplying two such matrices together allows us to establish the manner in which the rotation vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ combine.

$$\mathbf{r} = \frac{\mathbf{r}_2 + \mathbf{r}_1 + \mathbf{r}_2\times\mathbf{r}_1}{1 - \mathbf{r}_2\cdot\mathbf{r}_1}$$

for a rotation $\mathbf{r}_1$ followed by $\mathbf{r}_2$, so that rotations expressed in terms of rotation angles and axes may be compounded into a single such rotation without the need to form and decompose a product matrix.

Note that if $\mathbf{r}_1$ and $\mathbf{r}_2$ are parallel this reduces to the formula for the tangent of the sum of two angles, and that if $\mathbf{r}_1\cdot\mathbf{r}_2 = 1$ the combined rotation is always 180°. Note, too, that reversing the order of application of the rotations reverses only the vector product.

If three rotations $\mathbf{r}_1, \mathbf{r}_2$ and $\mathbf{r}_3$ are applied successively, $\mathbf{r}_1$ first, then their combined rotation is

$$\mathbf{r} = [\mathbf{r}_3(1-\mathbf{r}_1\cdot\mathbf{r}_2) + \mathbf{r}_2(1+\mathbf{r}_3\cdot\mathbf{r}_1) + \mathbf{r}_1(1-\mathbf{r}_3\cdot\mathbf{r}_2) + \mathbf{r}_3\times\mathbf{r}_2 + \mathbf{r}_3\times\mathbf{r}_1 + \mathbf{r}_2\times\mathbf{r}_1] \times [1 - \mathbf{r}_1\cdot\mathbf{r}_2 - \mathbf{r}_2\cdot\mathbf{r}_3 - \mathbf{r}_3\cdot\mathbf{r}_1 - \mathbf{r}_3\cdot(\mathbf{r}_2\times\mathbf{r}_1)]^{-1}.$$

Note the irregular pattern of signs in the numerator.

Similar ideas, using a vector of magnitude $\sin(\theta/2)$, are developed in Aharonov *et al.* (1977).

The third form of orthogonal matrix uses four variables, λ, μ, ν and σ, which comprise a four-dimensional vector $\boldsymbol{\rho}$, such that $\lambda = ls$, $\mu = ms$, $\nu = ns$ with $s = \sin(\theta/2)$ and $\sigma = \cos(\theta/2)$. In terms of these variables

$$\boldsymbol{R} = \begin{pmatrix} (\lambda^2-\mu^2-\nu^2+\sigma^2) & 2(\lambda\mu-\nu\sigma) & 2(\lambda\nu+\mu\sigma) \\ 2(\mu\lambda+\nu\sigma) & (-\lambda^2+\mu^2-\nu^2+\sigma^2) & 2(\mu\nu-\lambda\sigma) \\ 2(\lambda\nu-\mu\sigma) & 2(\mu\nu+\lambda\sigma) & (-\lambda^2-\mu^2+\nu^2+\sigma^2) \end{pmatrix}.$$

Two further matrices $\boldsymbol{S}$ and $\boldsymbol{T}$ may be defined (Diamond, 1988),

$$\boldsymbol{S} = \begin{pmatrix} -\sigma & \nu & -\mu & \lambda \\ -\nu & -\sigma & \lambda & \mu \\ \mu & -\lambda & -\sigma & \nu \\ \lambda & \mu & \nu & \sigma \end{pmatrix} \text{ and } \boldsymbol{T} = \begin{pmatrix} \sigma & -\nu & \mu & \lambda \\ \nu & \sigma & -\lambda & \mu \\ -\mu & \lambda & \sigma & \nu \\ -\lambda & -\mu & -\nu & \sigma \end{pmatrix},$$

which are themselves orthogonal (though $\boldsymbol{S}$ has determinant -1) and which have the property that

$$\boldsymbol{S}^2 = \begin{pmatrix} \boldsymbol{R} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{pmatrix}$$

so that, for example, if homogeneous coordinates are being employed (Section 3.3.1.1.2)

$$\begin{pmatrix} x' \\ y' \\ z' \\ w \end{pmatrix} = \begin{pmatrix} -\sigma & \nu & -\mu & \lambda \\ -\nu & -\sigma & \lambda & \mu \\ \mu & -\lambda & -\sigma & \nu \\ \lambda & \mu & \nu & \sigma \end{pmatrix} \begin{pmatrix} -\sigma & \nu & -\mu & \lambda \\ -\nu & -\sigma & \lambda & \mu \\ \mu & -\lambda & -\sigma & \nu \\ \lambda & \mu & \nu & \sigma \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix}$$

is a rotation of (x, y, z, w) through the angle θ about the axis (l, m, n). With suitably pipelined hardware this forms an efficient means of applying rotations since the 'overhead' of establishing $\boldsymbol{S}$ is so trivial.

$\boldsymbol{T}$ has the property that the rotation vector $\boldsymbol{\rho}$ arising from a concatenation of n rotations is

$$\boldsymbol{\rho} = \boldsymbol{T}_n\boldsymbol{T}_{n-1}\ldots\boldsymbol{T}_1\boldsymbol{\rho}_0,$$

in which $\boldsymbol{\rho}_0^T$ is the vector (0, 0, 0, 1) which defines a null rotation. This equation may be used as a basis for factorizing a given rotation into a concatenation of rotations about designated axes (Diamond, 1990*a*).

Finally, an exact rotation of the vector $\mathbf{d}$ may be obtained without using matrices at all by writing

$$\mathbf{d} = \sum_0^\infty \mathbf{d}_n$$

in which

$$\mathbf{d}_n = \frac{1}{n}(\boldsymbol{\theta}\times\mathbf{d}_{n-1})$$

and $\mathbf{d}_0$ is the initial position which is to be rotated. Here $\boldsymbol{\theta}$ is a vector with direction cosines l, m and n, and magnitude equal to the required rotation angle in radians (Diamond, 1966). This method is particularly efficient when $|\boldsymbol{\theta}| \ll 1$ or when the number of vectors to be transformed is small since the overhead of establishing $\boldsymbol{R}$ is eliminated and the process is simple to program. It is the three-dimensional analogue of the power series for $\sin\theta$ and $\cos\theta$ and has the same convergence properties.

3.3.1.2.2. *Measurement of rotations and strains from coordinates*

Given the coordinates of a molecular fragment it is often a requirement to relate the fragment to its image in some standard orientation by a transformation which may be required to be a pure rotation, or may be required to be a combination of rotation and strain. Of the methods reviewed in this section all except (iv) are concerned with pure rotation, ignoring any strain that may be present, and give the best rigid-body superposition. In all these methods, unless inhomogeneous strain is being considered, the best possible superposition is obtained if the centroids of the two images are first brought into coincidence by translation and treated as the origin.

Methods (i) to (v) seek transformations which perform the superposition and impose on these, in various ways, the requirements of orthogonality for the rotational part. All these methods therefore need some defence against indeterminacy that arises in the general transformation if one or both of the fragments is planar, and, if improper rotations are to be excluded, need a defence against these also if the fragment and its image are of opposite chirality. Methods (vi) and (vii) pay no attention to the general transformation and work with variables which are intrinsically rotational in character, and always produce an orthogonal transformation with positive determinant, with no degeneracy arising from planar fragments which need no special attention. Even collinear atoms cause no problem, the superposition being performed correctly but with an arbitrary rotation about the length of the line being present in the result. These methods are therefore to be preferred over the earlier ones unless the purpose of the operation is to detect differences of chirality, although this, too, can be detected with a simple test.

In this review we adopt the same notation for all the methods which, unavoidably, means that symbols are used in ways which differ from the original publications. We use the symbol $\boldsymbol{x}$ for the vector set which is to be rotated and $\boldsymbol{X}$ for the vector set whose orientation is not to be altered, and write the residuals as

$$e_{IA} = D_{Ij}x_{jA} - X_{IA}$$

and, by choice of origin,

$$W_a x_{Ia} = W_a X_{Ia} = 0_I$$

for weights W. The quadratic residual to be minimized is

$$E = W_a e_{ia} e_{ia}$$

and we define the matrix $M_{IJ} = W_a x_{Ia} X_{Ja}$ and use $\mathbf{l}$ for the direction cosines of the rotation axis.

(i) McLachlan's first method (McLachlan, 1972, 1982) is iterative and conceptually the simplest. It sets

$$D_{IJ} = A_{Ik}R_{kJ}$$

in which $\boldsymbol{A}$ and $\boldsymbol{R}$ are both orthogonal with $\boldsymbol{R}$ being a current estimate of $\boldsymbol{D}$ and $\boldsymbol{A}$ being an adjustment which, at the beginning of each cycle, has a zero angle associated with it. One iterative cycle estimates a non-trivial $\boldsymbol{A}$, after which the product $\boldsymbol{AR}$ replaces $\boldsymbol{R}$.

$$A_{IJ} = (1 - \cos\theta)l_I l_J + \delta_{IJ}\cos\theta - \varepsilon_{IJk}l_k \sin\theta$$

and

$$\left(\frac{\partial A_{IJ}}{\partial\theta}\right)_{\theta=0} = -\varepsilon_{IJk}l_k,$$

therefore

$$\left(\frac{\partial E}{\partial\theta}\right)_{\theta=0} = 2W_a\left(\frac{\partial A_{ij}}{\partial\theta}\right)_{\theta=0} R_{jk}x_{ka}(A_{il}R_{lm}x_{ma} - X_{ia})$$
$$= 2\varepsilon_{ijl}R_{jk}M_{ki}l_l.$$

For this to vanish for all possible rotation axes $\mathbf{l}$ the vector

$$g_L = \varepsilon_{ijL}R_{jk}M_{ki}$$

must vanish, *i.e.* at the end of the iteration $\boldsymbol{R}$ must be such that the matrix

$$N_{JI} = R_{Jk}M_{kI}$$

is symmetrical. The vector $\mathbf{g}$ represents the couple exerted on the rotating body by forces $2W_A(R_{Ij}x_{jA} - X_{IA})$ acting at the atoms. Choosing

$$l_L = g_L/|\mathbf{g}|$$

gives the greatest $|\partial E/\partial\theta|_{\theta=0}$ and $(\partial E/\partial\theta)$ vanishes when

$$\tan\theta = \frac{\varepsilon_{ijk}N_{ji}l_k}{N_{pq}(l_p l_q - \delta_{pq})}$$

in which $\boldsymbol{N}$ is constructed from the current $\boldsymbol{R}$ matrix. $\boldsymbol{A}$ is then constructed from $\mathbf{l}$ and this θ and $\boldsymbol{AR}$ replaces $\boldsymbol{R}$. The process is iterative because a couple about some new axis can appear when rotation about $\mathbf{g}$ eliminates the couple about $\mathbf{g}$.

Note that for each rotation axis $\mathbf{l}$ there are two values of θ, differing by π, which reduce $|\mathbf{g}|$ to zero, corresponding to maximum and minimum values of E. The minimum is that which makes

$$\frac{\partial^2 E}{\partial\theta^2} = 2(\text{tr } N - l_i N_{ij} l_j)$$

positive. Adding π to θ alters $\boldsymbol{R}$ and $\boldsymbol{N}$ and negates this quantity.

Note, too, that the process is essentially characterized as that which makes the product $\boldsymbol{RM}$ symmetrical with $\boldsymbol{R}$ orthogonal. We return to this point in (iii).

(ii) Kabsch's method (Kabsch, 1976, 1978) minimizes E with respect to the nine elements of $\boldsymbol{D}$, subject to the six constraints

$$D_{kI}D_{kJ} - \delta_{IJ} = 0_{IJ},$$

by using an auxiliary function

$$F = L_{ij}(D_{ki}D_{kj} - \delta_{ij})$$

in which $\boldsymbol{L}$ is symmetric containing six Lagrange multipliers. The Lagrangian function

$$G = E + F$$

then has minima with respect to the elements of $\boldsymbol{D}$ at locations which are dependent, *inter alia*, on the elements of $\boldsymbol{L}$. By suitably choosing $\boldsymbol{L}$ a minimum of G may be brought into coincidence with the constrained minimum of $\boldsymbol{E}$. A minimum of G occurs where

$$\frac{\partial G}{\partial D_{IJ}} = 2D_{Ik}(S_{Jk} + L_{Jk}) - 2M_{JI} = 0_{IJ}$$

and the 9×9 matrix

$$\frac{\partial^2 G}{\partial D_{MK}\partial D_{IJ}} = 2\delta_{MI}(S_{JK} + L_{JK})$$

is positive definite, block diagonal, and has

$$S_{JK} = W_a x_{Ja} x_{Ka}$$

which is symmetrical. Thus $\boldsymbol{L}$ must be chosen so as to make the symmetric matrix $(\boldsymbol{S} + \boldsymbol{L})$ such that

$$\boldsymbol{D}(\boldsymbol{S} + \boldsymbol{L})^T = \boldsymbol{M}^T$$

with $\boldsymbol{D}$ orthogonal, or $\boldsymbol{RN} = \boldsymbol{M}^T$ with $\boldsymbol{R}$ replacing $\boldsymbol{D}$ since we are now confined to the orthogonal case, and $\boldsymbol{N}$ is symmetric and positive definite.

(iii) Comparison of the Kabsch and McLachlan methods. Using the initials of these authors as subscripts, we have seen that the Kabsch solution involves solving

$$\boldsymbol{R}_{\rm WK}\boldsymbol{N}_{\rm WK} = \boldsymbol{M}^T$$

for an orthogonal matrix $\boldsymbol{R}_{\rm WK}$ given that $\boldsymbol{N}_{\rm WK}$ is symmetrical and positive definite. Thus

$$\boldsymbol{M}\boldsymbol{M}^T = \boldsymbol{N}_{\rm WK}^T\boldsymbol{R}_{\rm WK}^T\boldsymbol{R}_{\rm WK}\boldsymbol{N}_{\rm WK} = \boldsymbol{N}_{\rm WK}^2$$

and

$$\boldsymbol{R}_{\rm WK} = \boldsymbol{M}^T(\boldsymbol{M}\boldsymbol{M}^T)^{-1/2}.$$

By comparison, the McLachlan treatment leads to an orthogonal $\boldsymbol{R}$ matrix satisfying

$$\boldsymbol{R}_{\rm ADM} = \boldsymbol{N}_{\rm ADM}\boldsymbol{M}^{-1}$$

in which $\boldsymbol{N}_{\rm ADM}$ is also symmetric and positive definite, which similarly leads to

$$\boldsymbol{R}_{\rm ADM} = (\boldsymbol{M}^T\boldsymbol{M})^{1/2}\boldsymbol{M}^{-1}.$$

These seemingly different expressions for $\boldsymbol{R}_{\rm WK}$ and $\boldsymbol{R}_{\rm ADM}$ are, in fact, equal, as the following shows

$$\boldsymbol{R}_{\rm WK} = \boldsymbol{R}_{\rm ADM}\boldsymbol{R}_{\rm ADM}^{-1}\boldsymbol{R}_{\rm WK} = \boldsymbol{R}_{\rm ADM}\boldsymbol{M}\boldsymbol{N}_{\rm ADM}^{-1}\boldsymbol{M}^T\boldsymbol{N}_{\rm WK}^{-1},$$

therefore

$$\begin{aligned}\boldsymbol{R}_{\rm WK}^T\boldsymbol{R}_{\rm WK} &= \boldsymbol{I}\\ &= \boldsymbol{N}_{\rm WK}^{-1}\boldsymbol{M}\boldsymbol{N}_{\rm ADM}^{-1}\boldsymbol{M}^T\boldsymbol{R}_{\rm ADM}^T\boldsymbol{R}_{\rm ADM}\boldsymbol{M}\boldsymbol{N}_{\rm ADM}^{-1}\boldsymbol{M}^T\boldsymbol{N}_{\rm WK}^{-1}.\end{aligned}$$

Multiplying on both sides by $\boldsymbol{N}_{\rm WK}$ gives

$$\boldsymbol{N}_{\rm WK}^2 = (\boldsymbol{M}\boldsymbol{N}_{\rm ADM}^{-1}\boldsymbol{M}^T)^2,$$

and since both $\boldsymbol{N}$ matrices are positive definite

$$\boldsymbol{N}_{\rm WK} = \boldsymbol{M}\boldsymbol{N}_{\rm ADM}^{-1}\boldsymbol{M}^T$$

and conversely

$$\boldsymbol{N}_{\rm ADM} = \boldsymbol{M}^T\boldsymbol{N}_{\rm WK}^{-1}\boldsymbol{M},$$

therefore

$$\boldsymbol{R}_{\rm WK} = \boldsymbol{M}^T\boldsymbol{M}^{T-1}\boldsymbol{N}_{\rm ADM}\boldsymbol{M}^{-1} = \boldsymbol{R}_{\rm ADM}.$$

(iv) Diamond's first method. This method (Diamond, 1976*a*) differs from the previous ones in that the transformation $\boldsymbol{D}$ is allowed to be a general transformation which is then factorized into the product of an orthogonal matrix $\boldsymbol{R}$ and a symmetrical matrix $\boldsymbol{T}$. The transformation of $\boldsymbol{x}$ to fit $\boldsymbol{X}$ is thus interpreted as the combination of homogeneous strain and pure rotation in which $\boldsymbol{x}$ is subjected to strain and the result is rotated.

$$\begin{aligned}\boldsymbol{D} &= \boldsymbol{R}\boldsymbol{T}\\ \boldsymbol{T}^2 &= \boldsymbol{D}^T\boldsymbol{D}\\ \boldsymbol{T} &= (\boldsymbol{D}^T\boldsymbol{D})^{1/2}\\ \boldsymbol{R} &= \boldsymbol{D}(\boldsymbol{D}^T\boldsymbol{D})^{-1/2}.\end{aligned}$$

Furthermore, the solution for $\boldsymbol{D}$ is

$$\boldsymbol{D} = \boldsymbol{M}^T\boldsymbol{S}^{-1}$$

(in the notation of Kabsch), so that

$$\boldsymbol{R} = \boldsymbol{M}^T\boldsymbol{S}^{-1}(\boldsymbol{S}^{-1}\boldsymbol{M}\boldsymbol{M}^T\boldsymbol{S}^{-1})^{-1/2}$$

which may be compared with the results of the previous paragraph.

Although this $\boldsymbol{R}$ matrix by itself (*i.e.* applied without $\boldsymbol{T}$) does not produce the best rotational superposition (*i.e.* smallest E), it is the one which *exactly* superposes the only three vectors in $\boldsymbol{x}$ whose mutual dispositions are conserved, on their equivalents in $\boldsymbol{X}$, so that the rotation so found is arguably the best defined one.

Alternatives based on $\boldsymbol{D} = \boldsymbol{T}\boldsymbol{R}$, $\boldsymbol{D}^{-1} = \boldsymbol{R}\boldsymbol{T}$, $\boldsymbol{D}^{-1} = \boldsymbol{T}\boldsymbol{R}$ are all easily developed, and these ideas are developed by Diamond (1976*a*) to include non-homogeneous strains also.

(v) McLachlan's second method. This method (McLachlan, 1979) is based on the properties of the 6×6 matrix

$$\begin{pmatrix}\boldsymbol{0} & \boldsymbol{M}\\ \boldsymbol{M}^T & \boldsymbol{0}\end{pmatrix}$$

and is immune to singularity of $\boldsymbol{M}$. If $\mathbf{p}$ and $\mathbf{q}$ are three-dimensional vectors such that $(\mathbf{p}^T, \mathbf{q}^T)$ is an eigenvector of this matrix then

$$\begin{pmatrix}\boldsymbol{0} & \boldsymbol{M}\\ \boldsymbol{M}^T & \boldsymbol{0}\end{pmatrix}\begin{pmatrix}\mathbf{p}\\ \mathbf{q}\end{pmatrix} = \begin{pmatrix}\boldsymbol{M}\mathbf{q}\\ \boldsymbol{M}^T\mathbf{p}\end{pmatrix} = \begin{pmatrix}\mathbf{p}\lambda\\ \mathbf{q}\lambda\end{pmatrix}.$$

If $\mathbf{q}$ is negated the second equality is maintained provided λ is also negated. Therefore an orthogonal 6×6 matrix

$$\begin{pmatrix}\boldsymbol{H} & \boldsymbol{H}\\ \boldsymbol{K} & -\boldsymbol{K}\end{pmatrix}$$

(consisting of 3×3 partitions) exists for which

$$\begin{pmatrix}\boldsymbol{H}^T & \boldsymbol{K}^T\\ \boldsymbol{H}^T & -\boldsymbol{K}^T\end{pmatrix}\begin{pmatrix}\boldsymbol{0} & \boldsymbol{M}\\ \boldsymbol{M}^T & \boldsymbol{0}\end{pmatrix}\begin{pmatrix}\boldsymbol{H} & \boldsymbol{H}\\ \boldsymbol{K} & -\boldsymbol{K}\end{pmatrix} = \begin{pmatrix}\boldsymbol{\Lambda} & \boldsymbol{0}\\ \boldsymbol{0} & -\boldsymbol{\Lambda}\end{pmatrix}$$

in which $\boldsymbol{\Lambda}$ is diagonal and contains non-negative eigenvalues. The reverse transformation shows that

$$\boldsymbol{M} = 2\boldsymbol{H}\boldsymbol{\Lambda}\boldsymbol{K}^T$$

and multiplying the eigenvectors together gives

$$\boldsymbol{H}^T\boldsymbol{H} = \boldsymbol{K}^T\boldsymbol{K} = \tfrac{1}{2}\boldsymbol{I} = \boldsymbol{H}\boldsymbol{H}^T = \boldsymbol{K}\boldsymbol{K}^T.$$

Therefore

$$2\boldsymbol{K}\boldsymbol{H}^T\boldsymbol{M} = 4\boldsymbol{K}\boldsymbol{H}^T\boldsymbol{H}\boldsymbol{\Lambda}\boldsymbol{K}^T = 2\boldsymbol{K}\boldsymbol{\Lambda}\boldsymbol{K}^T,$$

but $2\boldsymbol{K}\boldsymbol{H}^T$ is orthogonal and $2\boldsymbol{K}\boldsymbol{\Lambda}\boldsymbol{K}^T$ is symmetrical, therefore [by paragraphs (i) and (iii) above] $2\boldsymbol{K}\boldsymbol{H}^T$ is the required rotation. Similarly, forming

$$\begin{aligned}\boldsymbol{M}^T &= 2\boldsymbol{K}\boldsymbol{\Lambda}\boldsymbol{H}^T\\ 2\boldsymbol{M}^T\boldsymbol{H}\boldsymbol{\Lambda}^{-1}\boldsymbol{H}^T &= 4\boldsymbol{K}\boldsymbol{\Lambda}\boldsymbol{H}^T\boldsymbol{H}\boldsymbol{\Lambda}^{-1}\boldsymbol{H}^T = 2\boldsymbol{K}\boldsymbol{H}^T\end{aligned}$$

corresponds to the Kabsch formulation [paragraphs (ii) and (iii)] since $2\boldsymbol{H}\boldsymbol{\Lambda}^{-1}\boldsymbol{H}^T$ is symmetrical and the same rotation, $2\boldsymbol{K}\boldsymbol{H}^T$, appears.

Note that the determinant of the orthogonal matrix so found is twice the product of the determinants of $\boldsymbol{H}$ and of $\boldsymbol{K}$, and since the positive eigenvalues are collected into $\boldsymbol{\Lambda}$ it follows that the sign of the determinant of $\boldsymbol{M}$ is the same as the sign of the determinant of the resulting orthogonal matrix. If this is negative it means that the best superposition is obtained if one vector set is inverted and that $\boldsymbol{x}$ and $\boldsymbol{X}$ are of opposite chirality.

Expanding the expression for E, the weighted sum of squares of errors, for an orthogonal transformation shows that this is least when the trace of the product $\boldsymbol{R}\boldsymbol{M}$ is greatest. In this treatment

$$\mathrm{tr}(\boldsymbol{R}\boldsymbol{M}) = \mathrm{tr}(2\boldsymbol{K}\boldsymbol{H}^T \cdot 2\boldsymbol{H}\boldsymbol{\Lambda}\boldsymbol{K}^T) = \mathrm{tr}(2\boldsymbol{K}\boldsymbol{\Lambda}\boldsymbol{K}^T) = \mathrm{tr}(\boldsymbol{\Lambda}).$$

Hence, if the eigenvalues in $\boldsymbol{\Lambda}$ and $-\boldsymbol{\Lambda}$ are arranged in decreasing

order of modulus, and if the determinant of $\boldsymbol{M}$ is negative, then exchanging the third and sixth columns of

$$\begin{pmatrix} \boldsymbol{H} & \boldsymbol{H} \\ \boldsymbol{K} & -\boldsymbol{K} \end{pmatrix}$$

produces a product $2\boldsymbol{K}\boldsymbol{H}^T$ with positive determinant (*i.e.* a proper rotation) at minimum cost in residual. Similarly, if $\boldsymbol{M}$ is singular and one or more eigenvalues in $\boldsymbol{\Lambda}$ vanishes it is necessary only to complete an orthonormal set of eigenvectors such that the determinants of $\boldsymbol{H}$ and $\boldsymbol{K}$ have the same sign.

(vi) MacKay's method. MacKay (1984) was the first to consider the rotational superposition problem in terms of the vector $\mathbf{r}$ of Section 3.3.1.2.1. Using quaternion algebra he showed that if a vector $\mathbf{x}$ is rotated to $\mathbf{X} = \boldsymbol{R}\mathbf{x}$ then

$$(\mathbf{X} - \mathbf{x}) = \mathbf{r} \times (\mathbf{X} + \mathbf{x}),$$

where $|\mathbf{r}| = \tan(\theta/2)$ and the direction of $\mathbf{r}$ is the axis of rotation, as may also be shown from elementary considerations. MacKay then solves this for the vector $\mathbf{r}$ by least squares given the vector pairs $\mathbf{X}$ and $\mathbf{x}$. The individual errors are

$$e_{IA} = \varepsilon_{Ijk} r_j (X_{kA} + x_{kA}) - (X_{IA} - x_{IA})$$

and

$$E = W_a e_{ia} e_{ia}.$$

Setting $\partial E/\partial r_P = 0_P$ gives

$$\begin{aligned} W_a \varepsilon_{iPk} \varepsilon_{ilm} r_l (X_{ka} + x_{ka})(X_{ma} + x_{ma}) \\ = W_a \varepsilon_{iPk} (X_{ka} + x_{ka})(X_{ia} - x_{ia}) \end{aligned}$$

which reduces to

$$2\mathbf{V} = -(\boldsymbol{Q} + \boldsymbol{Q}_0)\mathbf{r}$$

in which

$$\begin{aligned} \boldsymbol{Q} &= \boldsymbol{M} + \boldsymbol{M}^T - 2\boldsymbol{I}\,\mathrm{tr}\,\boldsymbol{M} \\ \boldsymbol{Q}_0 &= \boldsymbol{S} + \boldsymbol{S}' - \boldsymbol{I}(\mathrm{tr}\,\boldsymbol{S} + \mathrm{tr}\,\boldsymbol{S}') \\ V_I &= \varepsilon_{Ijk} M_{jk} \\ S_{IJ} &= W_a x_{Ia} x_{Ja} \\ S'_{IJ} &= W_a X_{Ia} X_{Ja}. \end{aligned}$$

Thus a direct solution for $\mathbf{r}$ is obtained,

$$\mathbf{r} = -2(\boldsymbol{Q}_0 + \boldsymbol{Q})^{-1}\mathbf{V},$$

the elements of which are u, v and w, and may be used to construct the orthogonal matrix as in Section 3.3.1.2.1. $\boldsymbol{Q} + \boldsymbol{Q}_0$ may be obtained directly from $\boldsymbol{X} + \boldsymbol{x}$.

If the requisite rotation is 180°, $(\boldsymbol{Q}_0 + \boldsymbol{Q})$ is singular and cannot be inverted. In this case any row or column of the adjoint of $(\boldsymbol{Q}_0 + \boldsymbol{Q})$ is a vector in the direction of the axis. Normalizing this vector to unity, giving $\boldsymbol{l}$, gives the requisite orthogonal matrix as

$$\boldsymbol{R} = 2\boldsymbol{l}\boldsymbol{l}^T - \boldsymbol{I}.$$

Note that MacKay's residual E is quadratic in $\mathbf{r}$. E therefore has one minimum and no maximum, and the minimum is reached on the first cycle of least squares. By contrast, the objective function E that is minimized in methods (i), (ii), (v) and (vii) has one minimum, one maximum and two saddle points in the space of the vector $\mathbf{r}$, as shown in (vii).

It may be shown (Diamond, 1989) that if MacKay's solution vector $\mathbf{r}$ is denoted by $\mathbf{r}_M$ and that given by the other methods [except (iv)] by $\mathbf{r}_O$ then

$$\mathbf{r}_M = \mathbf{r}_O - \boldsymbol{A}^{-1}\boldsymbol{B}\mathbf{r}_O$$

in which $\boldsymbol{A}$ and $\boldsymbol{B}$ are real symmetric, positive semi-definite. $\boldsymbol{A}$ is positive definite unless all the individual vector sums $(\mathbf{X} + \mathbf{x})$ are parallel, as can happen when the best rotation is 180°. Thus the MacKay method only gives the same result as the other methods if:

(*a*) the initial orientation is optimal, for then $\mathbf{r}_O = \mathbf{0}$, or

(*b*) perfect fitting is possible, for then $\boldsymbol{B} = \mathbf{0}$, or

(*c*) all the residual vectors (after fitting by $\mathbf{r}_O$) are parallel to $\mathbf{r}_O$, for then $\boldsymbol{B}$ is singular such that $\boldsymbol{B}\mathbf{r}_O = \mathbf{0}$. In general, $|\mathbf{r}_M| \leq |\mathbf{r}_O|$. $\mathbf{r}_O$ may be found by iterating $\mathbf{r}_M$, but $\boldsymbol{x}$ must be replaced by $\boldsymbol{R}\boldsymbol{x}$ on each iteration.

(vii) Diamond's second method. This is closely related to MacKay's method, but uses a four-dimensional vector $\boldsymbol{\rho}$ with components λ, μ, ν and σ in which λ, μ and ν are the direction cosines of the rotation axis multiplied by $\sin(\theta/2)$ and σ is $\cos(\theta/2)$. In terms of such a vector Diamond (1988) showed that

$$E = E_0 - 2\boldsymbol{\rho}^T\boldsymbol{P}\boldsymbol{\rho}$$

in which E is the weighted sum of squares of coordinate differences, as before, E_0 is its value before any rotation is applied and $\boldsymbol{P}$ is the matrix

$$\boldsymbol{P} = \begin{pmatrix} \boldsymbol{Q} & \mathbf{V} \\ \mathbf{V}^T & 0 \end{pmatrix}.$$

The rotation matrix $\boldsymbol{R}$ corresponding to the vector $\boldsymbol{\rho}$ is then the last of the forms for $\boldsymbol{R}$ given in Section 3.3.1.2.1.

The minimum E is therefore E_0 minus twice the largest eigenvalue of $\boldsymbol{P}$ since $\boldsymbol{\rho}^T\boldsymbol{\rho} = 1$, and a stationary value of E occurs when $\boldsymbol{\rho}$ is any of the four eigenvectors of $\boldsymbol{P}$. E thus has a maximum, a minimum and two saddle points, in general, and its value may be determined before any coordinates are transformed. Diamond also showed that the orientations giving these stationary values are related by the operations of 222 symmetry. Equivalent results have also been obtained by Kearsley (1989).

As an alternative to solving a 4×4 eigenproblem, Diamond also showed that the vector $\mathbf{r}$, as in MacKay's solution, may be obtained by iterating

$$\begin{aligned} \alpha_0 &= E_0/2 \\ \mathbf{r}_n &= (\alpha_n \boldsymbol{I} - \boldsymbol{Q})^{-1}\mathbf{V} \\ \alpha_{n+1} &= \frac{\mathbf{V} \cdot \mathbf{r}_n + \alpha_n r_n^2}{1 + r_n^2} \end{aligned}$$

which has the property that if $\boldsymbol{X}$ and $\boldsymbol{x}$ are exactly superposable then α_0 is the exact solution and no iteration is necessary. If $\boldsymbol{X}$ and $\boldsymbol{x}$ are similar but not exactly superposable then a small number of iterations may be required to reach a stable $\mathbf{r}$ vector, though the matrix $\boldsymbol{Q}_0$ is not required. As in MacKay's solution, $(\alpha\boldsymbol{I} - \boldsymbol{Q})$ is singular at the end of the iteration if the required rotation is 180°, but the MacKay and Diamond methods both have the advantage that improper rotations are never generated by these means, and methods based on $\boldsymbol{P}$ and $\boldsymbol{\rho}$ rather than $\boldsymbol{Q}$ and $\mathbf{r}$ are trouble-free for 180° rotations. The iterative loop in this method does not require $\boldsymbol{R}\boldsymbol{x}$ to be redetermined on each cycle.

Finally, it may be shown that if p_1, p_2, p_3, p_4 are the eigenvalues of $\boldsymbol{P}$ arranged in descending order and

$$p_1 - p_2 - p_3 + p_4$$

is negative, then a closer superposition may be obtained by reversing the chirality of one of the vector sets, and the $\boldsymbol{R}$ matrix constructed from $\boldsymbol{\rho}_4$ optimally superimposes $\boldsymbol{R}\boldsymbol{x}$ onto $-\boldsymbol{X}$, the enantiomer of $\boldsymbol{X}$ (Diamond, 1990*b*).

3.3.1.2.3. *Orthogonalization of impure rotations*

There are several ways of deriving a strictly orthogonal matrix from a given approximately orthogonal matrix, among them the following.

(i) The Gram–Schmidt process. This is probably the simplest and the easiest to compute. If the given matrix consists of three column vectors $\mathbf{v}_1, \mathbf{v}_2$ and $\mathbf{v}_3$ (later referred to as primers) which are to be replaced by three column vectors $\mathbf{u}_1, \mathbf{u}_2$ and $\mathbf{u}_3$ then the process is

$$\begin{aligned}\mathbf{u}_1 &= \mathbf{v}_1/|\mathbf{v}_1| \\ \mathbf{u}_2 &= \mathbf{v}_2 - (\mathbf{u}_1 \cdot \mathbf{v}_2)\mathbf{u}_1 \\ \mathbf{u}_2 &= \mathbf{u}_2/|\mathbf{u}_2| \\ \mathbf{u}_3 &= \mathbf{v}_3 - (\mathbf{u}_1 \cdot \mathbf{v}_3)\mathbf{u}_1 - (\mathbf{u}_2 \cdot \mathbf{v}_3)\mathbf{u}_2 \\ \mathbf{u}_3 &= \mathbf{u}_3/|\mathbf{u}_3|.\end{aligned}$$

As successive vectors are established, each vector $\mathbf{v}$ has subtracted from it its components in the directions of established vectors, and the remainder is normalized. The method will fail at the normalization step if the vectors $\mathbf{v}$ are not linearly independent. Otherwise, the process may be extended to any number of dimensions.

The weakness of the method is that, though $\mathbf{u}_1$ differs from $\mathbf{v}_1$ only in scale, $\mathbf{u}_N$ may differ grossly from $\mathbf{v}_N$ as the various columns are not treated equivalently.

(ii) A preferable method which treats all vectors equivalently is to iteratively replace the matrix $\boldsymbol{M}$ by $\frac{1}{2}(\boldsymbol{M} + \boldsymbol{M}^{T-1})$.

Defining the residual matrix $\boldsymbol{E}$ as

$$\boldsymbol{E} = \boldsymbol{M}\boldsymbol{M}^T - \boldsymbol{I},$$

then on each iteration $\boldsymbol{E}$ is replaced by

$$\boldsymbol{E}^2(\boldsymbol{M}\boldsymbol{M}^T)^{-1}/4$$

and convergence necessarily ensues.

(iii) A third method resolves $\boldsymbol{M}$ into its symmetric and antisymmetric parts

$$\boldsymbol{S} = \tfrac{1}{2}(\boldsymbol{M} + \boldsymbol{M}^T), \quad \boldsymbol{A} = \tfrac{1}{2}(\boldsymbol{M} - \boldsymbol{M}^T), \quad \boldsymbol{M} = \boldsymbol{S} + \boldsymbol{A}$$

and constructs an orthogonal matrix for which only $\boldsymbol{S}$ is altered. $\boldsymbol{A}$ determines l, m, n and θ as shown in Section 3.3.1.2.1, and from these a new $\boldsymbol{S}$ may be constructed.

(iv) A fourth method is to treat the general matrix $\boldsymbol{M}$ as a combination of pure strain and pure rotation. Setting

$$\boldsymbol{M} = \boldsymbol{R}\boldsymbol{T}$$

with $\boldsymbol{R}$ orthogonal and $\boldsymbol{T}$ symmetrical gives

$$\boldsymbol{T} = (\boldsymbol{M}^T\boldsymbol{M})^{1/2}, \quad \boldsymbol{R} = \boldsymbol{M}(\boldsymbol{M}^T\boldsymbol{M})^{-1/2}.$$

The rotation so found is the one which exactly superposes those three mutually perpendicular directions which remain mutually perpendicular under the transformation $\boldsymbol{M}$.

$\boldsymbol{T} - \boldsymbol{I}$ is then the strain tensor of an unrotated body.

Writing $\boldsymbol{M} = \boldsymbol{T}\boldsymbol{R}$, $\boldsymbol{T} = (\boldsymbol{M}\boldsymbol{M}^T)^{1/2}$, $\boldsymbol{R} = (\boldsymbol{M}\boldsymbol{M}^T)^{-1/2}\boldsymbol{M}$ may also be useful, in which $\boldsymbol{T} - \boldsymbol{I}$ is the strain tensor of a rotated body. See also Section 3.3.1.2.2 (iv).

3.3.1.2.4. *Eigenvalues and eigenvectors of orthogonal matrices*

If $\boldsymbol{R}$ is the orthogonal matrix given in Section 3.3.1.2.1 in terms of the direction cosines l, m and n of the axis of rotation, then it is clear that (l, m, n) is an eigenvector of $\boldsymbol{R}$ with eigenvalue unity because

$$\boldsymbol{R}\begin{pmatrix} l \\ m \\ n \end{pmatrix} = \begin{pmatrix} l \\ m \\ n \end{pmatrix}.$$

Consideration of the determinant $|\boldsymbol{R} - \lambda\boldsymbol{I}| = 0$ shows that the sum of the three eigenvalues is $1 + 2\cos\theta$ and that their product is unity. Hence the three eigenvalues are 1, $e^{i\theta}$ and $e^{-i\theta}$. Since $\boldsymbol{R}$ is real, its product with any real vector is also real, yet its product with an eigenvector must, in general, be complex. Thus the eigenvectors must themselves be complex.

The remaining two eigenvectors $\mathbf{u}$ may be found using the results of Section 3.3.1.2.1 (*q.v.*) according to

$$\boldsymbol{R}\mathbf{u} = \mathbf{u} + \frac{2}{1+t^2}\{(\mathbf{r} \times \mathbf{u}) + [\mathbf{r} \times (\mathbf{r} \times \mathbf{u})]\} = \mathbf{u}e^{\pm i\theta} = \mathbf{u}\frac{1 \pm it}{1 \mp it},$$

which is solved by any vector of the form

$$\mathbf{u} = \mathbf{l} \times \mathbf{v} \mp i\mathbf{l} \times (\mathbf{l} \times \mathbf{v})$$

for any real vector $\mathbf{v}$, where $\mathbf{l}$ is the normalized axis vector, $\mathbf{l}t = \mathbf{r}$, $|\mathbf{l}| = 1$, $t = \tan(\theta/2)$. Eigenvectors for the two eigenvalues may have unrelated $\mathbf{v}$ vectors though the sign choices are coupled. If the vector $\mathbf{v}$ is rotated about $\mathbf{l}$ through an angle φ the corresponding vector $\mathbf{u}$ is multiplied by $e^{-i\varphi}$ and remains an eigenvector. Using superscript signs to denote the sign of θ in the eigenvalue with which each vector is associated, the matrix

$$\boldsymbol{U} = (\mathbf{l},\ \mathbf{u}^+,\ \mathbf{u}^-)$$

has the properties that

$$\boldsymbol{R}\boldsymbol{U} = \boldsymbol{U}\begin{pmatrix} 1 & 0 & 0 \\ 0 & e^{i\theta} & 0 \\ 0 & 0 & e^{-i\theta} \end{pmatrix}$$

and

$$\boldsymbol{U}^{*T}\boldsymbol{U} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2|\mathbf{l} \times \mathbf{v}^+|^2 & 0 \\ 0 & 0 & 2|\mathbf{l} \times \mathbf{v}^-|^2 \end{pmatrix}$$

which places restrictions on $\mathbf{v}$ if this is to be the identity. Note that the 23 element vanishes even in the absence of any relationship between $\mathbf{v}^+$ and $\mathbf{v}^-$.

A convenient form for $\boldsymbol{U}$, symmetrical in the elements of $\mathbf{l}$, is obtained by setting $\mathbf{v}^+ = \mathbf{v}^- = [111]$ and is

$$\boldsymbol{U} = \begin{pmatrix} l & \{(m-n) - i[l(l+m+n)-1]\}/d & \{(m-n) + i[l(l+m+n)-1]\}/d \\ m & \{(n-l) - i[m(l+m+n)-1]\}/d & \{(n-l) + i[m(l+m+n)-1]\}/d \\ n & \{(l-m) - i[n(l+m+n)-1]\}/d & \{(l-m) + i[n(l+m+n)-1]\}/d \end{pmatrix}$$

in which the normalizing denominator is given by

$$d = 2\sqrt{1 - lm - mn - nl}.$$

3.3.1.3. *Projection transformations and spaces*

In the following section we address the question of the relationship between the coordinates of a molecular model and the corresponding coordinates on the screen of the graphics device. A good introduction to this topic is given by Newman & Sproull (1973), and Foley *et al.* (1990) give a comprehensive account of the field, including recent developments, especially those arising from the development of raster-graphics technologies.

3.3.1.3.1. *Definitions*

Typically, the coordinates, $\mathbf{X}$, of points in an object to be drawn are held in homogeneous Cartesian form as described in Section 3.3.1.1.2. Such coordinates are said to be in *data space* or world

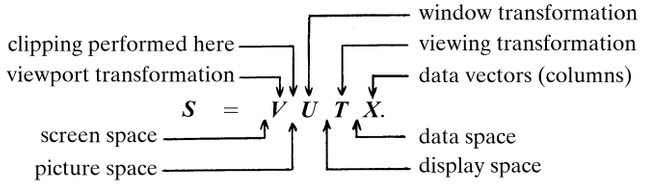
coordinates and this coordinate system is generally a constant aspect of the problem.

In order to view these data in convenient ways such coordinates may be subjected to a 4×4 *viewing transformation* $\boldsymbol{T}$, affecting orientation, scale *etc.*, the resulting coordinates $\boldsymbol{T}\mathbf{X}$ being then in *display space*. Here, and throughout what follows, we treat position vectors as columns with transformation matrices as factors on the left, though some writers do the reverse.

In general, only some portion of display space which lies inside a certain frustum of a pyramid is required to fall within the picture. The pyramid may be thought of as having the observer's eye at its vertex, with a rectangular base corresponding to the picture area. This volume is called a *window*. A transformation, $\boldsymbol{U}$, which takes *display-space* coordinates as input and generates vectors (X, Y, Z, W) for which X/W and $Y/W = \pm 1$ for points on the left, right, top and bottom boundaries of the window and for which Z/W takes particular values on the front and back planes of the window, is said to be a *windowing transformation*. In machines for which Z/W controls intensity depth cueing, the range of Z/W corresponding to the window is likely to be 0 to 1 rather than -1 to 1. Coordinates obtained by multiplying display-space coordinates by $\boldsymbol{U}$ are termed *picture-space* coordinates. Mathematically, $\boldsymbol{U}$ is a 4×4 matrix like any other, but functionally it is special. Declaring a transformation to be a windowing transformation implies that only resulting points having $|X|, |Y| < W$ and positive $Z < W$ are to be plotted. Machines with clipping hardware to truncate lines which run out of the picture perform clipping on the output from the windowing transformation.

Finally, the picture has to be drawn in some rectangular portion of the screen which is allocated for the purpose. Such an area is termed a *viewport* and is defined in terms of *screen coordinates* which are defined absolutely for the hardware in question as $\pm n$ for full-screen deflection, where n is declared by the manufacturer. *Screen coordinates* are obtained from picture coordinates with a *viewport transformation*, $\boldsymbol{V}$.*

To summarize, screen coordinates, $\mathbf{S}$, are given by



3.3.1.3.2. *Translation*

The transformation

$$\begin{pmatrix} N\boldsymbol{I} & \mathbf{V} \\ \mathbf{0}^T & N \end{pmatrix}\begin{pmatrix} \mathbf{X} \\ W \end{pmatrix} = \begin{pmatrix} \mathbf{X}N + \mathbf{V}W \\ NW \end{pmatrix} \simeq \begin{pmatrix} \mathbf{X} + \mathbf{V}W/N \\ W \end{pmatrix} \simeq \begin{pmatrix} \mathbf{X}/W + \mathbf{V}/N \\ 1 \end{pmatrix}$$

evidently corresponds to the addition of the vector $\mathbf{V}W/N$ to the components of $\mathbf{X}$ or of $\mathbf{V}/N$ to the components of $\mathbf{X}/W$. ($\boldsymbol{I}$ is the identity.) Displacements may thus be affected by expressing the required displacement vector in homogeneous coordinates with any suitable choice of N (commonly, $N = W$), with $\mathbf{V}$ scaled to correspond to this choice, and loading the 4×4 transformation matrix as indicated above.

* In recent years it has become increasingly common, especially in two-dimensional work, to apply the term 'window' to what is here called a viewport, but in this chapter we use these terms in the manner described in the text.

3.3.1.3.3. *Rotation*

Rotation about the origin is achieved by

$$\begin{pmatrix} N\boldsymbol{R} & \mathbf{0} \\ \mathbf{0}^T & N \end{pmatrix}\begin{pmatrix} \mathbf{X} \\ W \end{pmatrix} = \begin{pmatrix} N\boldsymbol{R}\mathbf{X} \\ NW \end{pmatrix} \simeq \begin{pmatrix} \boldsymbol{R}\mathbf{X} \\ W \end{pmatrix},$$

in which $\boldsymbol{R}$ is an orthogonal 3×3 matrix. $\boldsymbol{R}$ necessarily has elements not exceeding one in modulus. For machines using integer arithmetic, therefore, N would be chosen large enough (usually half the largest possible integer) for the product $N\boldsymbol{R}$ to be well represented in the available word length. Characteristically, N affects resolution but not scale.

3.3.1.3.4. *Scale*

The transformation

$$\begin{pmatrix} SN\boldsymbol{I} & \mathbf{0} \\ \mathbf{0}^T & N \end{pmatrix}\begin{pmatrix} \mathbf{X} \\ W \end{pmatrix} = \begin{pmatrix} SN\mathbf{X} \\ NW \end{pmatrix} \simeq \begin{pmatrix} S\mathbf{X} \\ W \end{pmatrix}$$

scales the vector $(\mathbf{X}, W)$ by the factor S. For integer working and $|S| < 1$, N should be set to the largest representable integer. For $|S| > 1$ the product SN should be the largest representable integer, N being reduced accordingly.

3.3.1.3.5. *Windowing and perspective*

It is necessary at this point to relate the discussion to the axial system inherent in the graphics device employed. One common system adopts X horizontal and to the right when viewing the screen, Y vertically upwards in the plane of the screen, and Z normal to X and Y with $+Z$ into the screen. This is, unfortunately, a left-handed system in that $(\mathbf{X} \times \mathbf{Y}) \cdot \mathbf{Z}$ is negative. Since it is usual in crystallographic work to use right-handed axial systems it is necessary to incorporate a matrix of the form

$$\begin{pmatrix} W & 0 & 0 & 0 \\ 0 & W & 0 & 0 \\ 0 & 0 & -W & 0 \\ 0 & 0 & 0 & W \end{pmatrix}$$

either as the left-most factor in the matrix $\boldsymbol{T}$ or as the right-most factor in the windowing transformation $\boldsymbol{U}$ (see Section 3.3.1.3.1). The latter choice is to be preferred and is adopted here. The former choice leads to complications if transformations in display space will be required. Display-space coordinates are necessarily referred to this axial system.

Let L, R, T, B, N and F be the left, right, top, bottom, near and far boundaries of the windowed volume $(L < R, T > B, N < F)$, S be the Z coordinate of the screen, and C, D and E be the coordinates of the observer's eye position, all ten of these parameters being referred to the origin of display space as origin, which may be anywhere in relation to the hardware. L, R, T and B are to be evaluated in the screen plane. All ten parameters may be referred to their own fourth coordinate, V, meaning that the point (X, Y, Z, W) in display space will be on the left boundary of the picture if $X/W = L/V$ when $Z/W = S/V$. V may be freely chosen so that all eleven quantities and all elements of $\boldsymbol{U}$ suit the word length of the machine. These relationships are illustrated in Fig. 3.3.1.1.

Since

$$(X, Y, Z, W) \simeq \left(\frac{XV}{W}, \frac{YV}{W}, \frac{ZV}{W}, V\right),$$

XV/W is a display-space coordinate on the same scale as the window parameters. This must be plotted on the screen at an X coordinate (on the scale of the window parameters) which is the weighted mean of XV/W and C, the weights being $(S - E)$ and

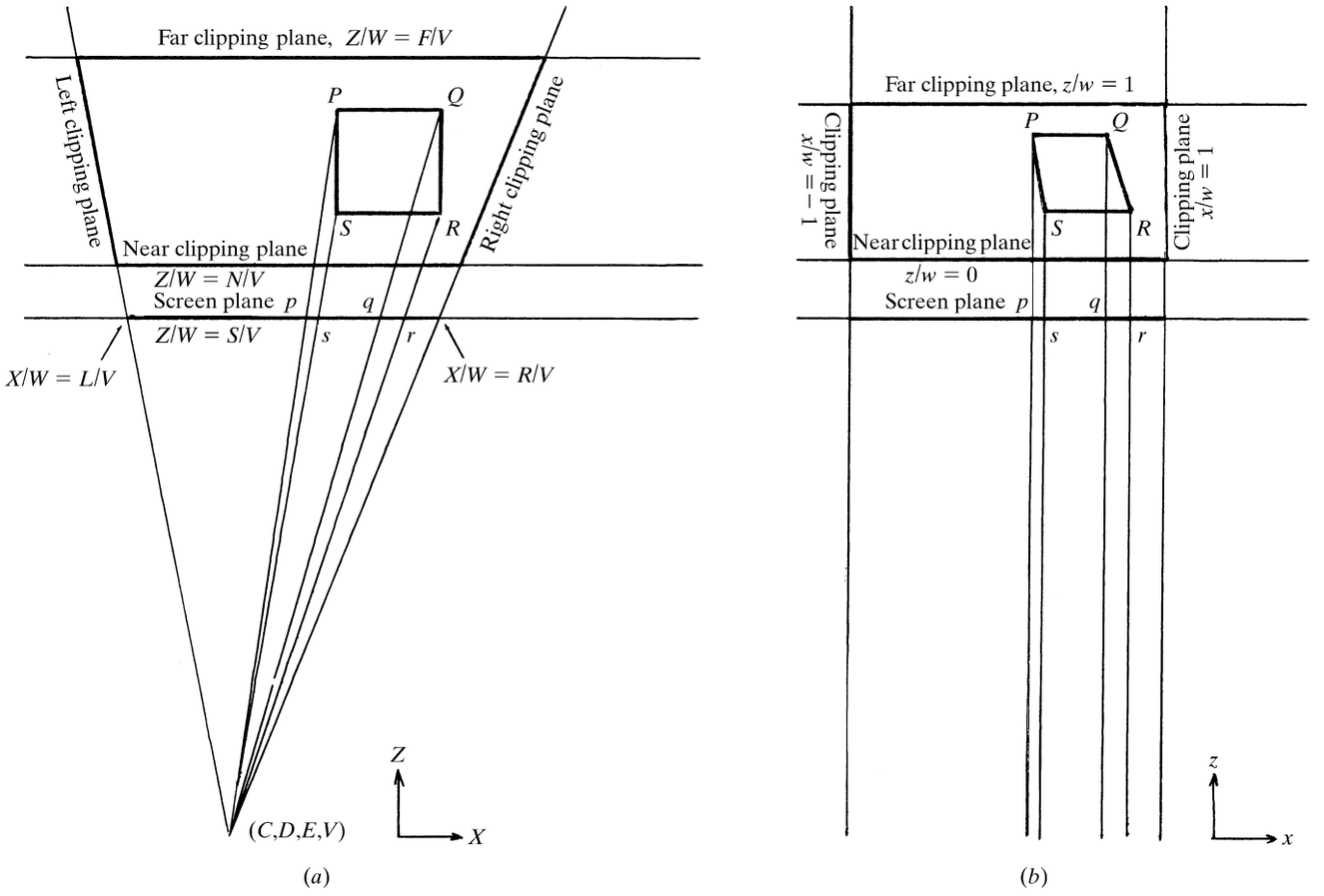


Fig. 3.3.1.1. The relationship between display-space coordinates (X, Y, Z, W) and picture-space coordinates (x, y, z, w) derived from them by the window transformation, $\boldsymbol{U}$. (*a*) Display space (in X, Z projection) showing a square object P, Q, R, S for display viewed from the position (C, D, E, V). The bold trapezium is the window (volume) and the bold line is the viewport portion of the screen. The points P, Q, R and S must be plotted at p, q, r and s to give the correct impression of the object. (*b*) Picture space (in x, z projection). The window is mapped to a rectangle and all sight lines are parallel to the z axis, but the object P, Q, R, S is no longer square. The distribution of p, q, r and s is identical in the two cases. Note that z/w values are not linear on Z/W, and that the origin of picture space arises at the midpoint of the near clipping plane, regardless of the location of the origin of display space. The figure is accurately to scale for coincident viewport positions. The words 'Left clipping plane', if part of the scene in display space, would currently be obscured, but would come into view if the eye moved to the right, increasing C, as the left clipping plane would pivot about the point L/V in the screen plane.

$(ZV/W - S)$, respectively. This provides perspective because the weighted mean is at the point where the straight line from (X, Y, Z, W) to the eye intersects the screen. This then has to be mapped into the L-to-R interval, so that picture-space coordinates (x, y, z, w) are given by

$$\begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix} = \begin{pmatrix} \frac{2(S-E)V}{(R-L)} & 0 & \frac{(2C-R-L)V}{(R-L)} & \frac{(R+L)E-2SC}{(R-L)} \\ 0 & \frac{2(S-E)V}{(T-B)} & \frac{(2D-T-B)V}{(T-B)} & \frac{(T+B)E-2SD}{(T-B)} \\ 0 & 0 & \frac{(F-E)V}{(F-N)} & \frac{-N(F-E)}{(F-N)} \\ 0 & 0 & V & -E \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ W \end{pmatrix}$$

which provides for $|x/w|$ and $|y/w|$ to be unity on the picture boundaries, which is usually a requirement of the clipping hardware, and for $0 < z/w < 1$, zero being for the near-plane boundary. Even though z/w is not linear on Z/W, straight lines and planes in display space transform to straight lines and planes in picture space, the non-linearity affecting only distances. Thus vector-drawing machines are not disadvantaged by the introduction of perspective.

Note that the dimensionality of X/W must equal that of S/V and that this may be regarded as length or as a pure number, but that in either case x/w is dimensionless, consistent with the stipulation that the picture boundaries be defined by the pure number ± 1.

The above matrix is $\boldsymbol{U}$ and is suited to left-handed hardware systems. Note that only the last column of $\boldsymbol{U}$ (the translational part) is sensitive to the location of the origin of display space and that if the eye is on the normal to the picture centre then $C = \frac{1}{2}(R + L)$, $D = \frac{1}{2}(T + B)$ and simplifications result. If C, D and E can be continuously monitored then dynamic parallax as well as perspective may be obtained (Diamond *et al.*, 1982).

If data space is referred to right-handed axes, the viewing transformation $\boldsymbol{T}$ involves only proper rotations and the hardware uses a left-handed axial system then elements in the third column of $\boldsymbol{U}$ should be negated, as explained in the opening paragraph.

To provide for orthographic projection, multiply every element of $\boldsymbol{U}$ by $-K/E$ and then let $E \rightarrow -\infty$, choosing some positive K to suit the word length of the machine [see Section 3.3.1.1.2 (iii)]. The

result is

$$\boldsymbol{U}' \simeq \begin{pmatrix} \frac{2KV}{(R-L)} & 0 & 0 & \frac{-K(R+L)}{(R-L)} \\ 0 & \frac{2KV}{(T-B)} & 0 & \frac{-K(T+B)}{(T-B)} \\ 0 & 0 & \frac{KV}{(F-N)} & \frac{-KN}{(F-N)} \\ 0 & 0 & 0 & K \end{pmatrix},$$

which is the orthographic window.

It may be convenient in some applications to separate the functions of windowing and the application of perspective, and to write

$$\boldsymbol{U} = \boldsymbol{U}'\boldsymbol{P},$$

where $\boldsymbol{U}$ and $\boldsymbol{U}'$ are as above and $\boldsymbol{P}$ is a perspective transformation given by

$$\boldsymbol{P} = (\boldsymbol{U}')^{-1}\boldsymbol{U} \simeq \begin{pmatrix} S-E & 0 & C & -SC/V \\ 0 & S-E & D & -SD/V \\ 0 & 0 & F-E+N & -NF/V \\ 0 & 0 & V & -E \end{pmatrix},$$

which involves F and N but not R, L, T or B. In this form the action of $\boldsymbol{P}$ may be thought of as compressing distant parts of display space prior to an orthographic projection by $\boldsymbol{U}'$ into picture space.

Other factorizations of $\boldsymbol{U}$ are possible, for example

$$\boldsymbol{U} = \boldsymbol{U}''\boldsymbol{P}'$$

with

$$\boldsymbol{U}'' \simeq \begin{pmatrix} \frac{2KV}{R-L} & 0 & 0 & \frac{-K(R+L)}{(R-L)} \\ 0 & \frac{2KV}{T-B} & 0 & \frac{-K(T+B)}{(T-B)} \\ 0 & 0 & \frac{KV(N-E)(F-E)}{E^2(F-N)} & \frac{KN(F-E)}{E(F-N)} \\ 0 & 0 & 0 & K \end{pmatrix}$$

$$\boldsymbol{P}' \simeq \begin{pmatrix} S-E & 0 & C & -SC/V \\ 0 & S-E & D & -SD/V \\ 0 & 0 & -E & 0 \\ 0 & 0 & V & -E \end{pmatrix},$$

which renders $\boldsymbol{P}'$ independent of all six boundary planes, but $\boldsymbol{U}''$ is no longer independent of E. It is not possible to factorize $\boldsymbol{U}$ so that the left factor is a function only of the boundary planes and the right factor a function only of eye and screen positions.

Note that as $E \to -\infty$, $\boldsymbol{U}'' \to \boldsymbol{U}'$, $\boldsymbol{P}$ and $\boldsymbol{P}' \to -\boldsymbol{I}E \simeq \boldsymbol{I}$.

3.3.1.3.6. *Stereoviews*

Assuming that left- and right-eye views are to be presented through the same viewport (next section) or that their viewports are to be superimposed by an external optical system, *e.g.* Ortony mirrors, then stereopairs are obtained by using appropriate eye coordinates in the $\boldsymbol{U}$ matrix of the previous section. However, $\boldsymbol{U}$ may be factorized according to

$$\boldsymbol{U} = \boldsymbol{U}'''\boldsymbol{S}$$

in which $\boldsymbol{U}'''$ is the matrix $\boldsymbol{U}$ obtained by setting (C, D, E, V) to correspond to the point midway between the viewer's eyes and

$$\boldsymbol{S} = \begin{pmatrix} 1 & 0 & c/(S-E) & -cS/(S-E)V \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \simeq \begin{pmatrix} V & 0 & cV/(S-E) & -cS/(S-E) \\ 0 & V & 0 & 0 \\ 0 & 0 & V & 0 \\ 0 & 0 & 0 & V \end{pmatrix}$$

in which $(c, 0, 0, V)$ is the position of the right eye relative to the mean eye position, and the left-eye view is obtained by negating c.

Stereo is often approximated by introducing a rotation about the Y axis of $\pm\sin^{-1}[c/(S-E)]$ to the views or $\sin^{-1}[2c/(S-E)]$ to one of them. The first corresponds to

$$\boldsymbol{S} = \begin{pmatrix} \sqrt{1-\sigma^2} & 0 & \sigma & 0 \\ 0 & 1 & 0 & 0 \\ -\sigma & 0 & \sqrt{1-\sigma^2} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

with $\sigma = c/(S-E)$. The main difference is in the resulting Z value, which only affects depth cueing and z clipping. The X translation which arises if $S \neq 0$ is also suppressed, but this is not likely to be noticeable. σ is often treated as a constant, such as $\sin 3^\circ$.

The distinction in principle between the true $\boldsymbol{S}$ and the rotational approximation is that with the true $\boldsymbol{S}$ the eye moves relative to the screen and the displayed object, whereas with the approximation the eye and the screen are moved relative to the displayed object, in going from one view to the other.

Strobing of left and right images may conveniently be accomplished with an electro-optic liquid-crystal shutter as described by Harris *et al.* (1985). The shutter is switched by the display itself, thus solving the synchronization problem in a manner free of inertia.

A further discussion of stereopairs may be found in Johnson (1970) and in Thomas (1993), the second of which generalizes the treatment to allow for the possible presence of an optical system.

3.3.1.3.7. *Viewports*

The window transformation of the previous two sections has been constructed to yield picture coordinates (X, Y, Z, W) (formerly called x, y, z, w) such that a point having X/W or $Y/W = \pm 1$ is on the boundary of the picture, and the clipping hardware operates on this basis. However, the edges of the picture need not be at the edges of the screen and a viewport transformation, $\boldsymbol{V}$, is therefore needed to position the picture in the requisite part of the screen.

$$\boldsymbol{V} = \begin{pmatrix} (r-l)/2 & 0 & 0 & (r+l)/2 \\ 0 & (t-b)/2 & 0 & (t+b)/2 \\ 0 & 0 & n & 0 \\ 0 & 0 & 0 & n \end{pmatrix},$$

where r, l, t and b are now the right, left, top and bottom boundaries of the picture area, or viewport, expressed in screen coordinates, and n is the full-screen deflection value. Thus a point with $X/W = 1$ in picture space plots on the screen with an X coordinate which is a fraction r/n of full-screen deflection to the right. Z/W is unchanged

by $\boldsymbol{V}$ and is used only to control intensity in a technique known as depth cueing.

It is necessary, of course, to arrange for the aspect ratio of the viewport, $(r-l)/(t-b)$, to equal that of the window otherwise distortions are introduced.

3.3.1.3.8. *Compound transformations*

In this section we consider the viewing transformation $\boldsymbol{T}$ of Section 3.3.1.3.1 and its construction in terms of translation, rotation and scaling, Sections 3.3.1.3.2–4. We use $\boldsymbol{T}'$ to denote a new transformation in terms of the prevailing transformation $\boldsymbol{T}$.

We note first that any 4×4 matrix of the form

$$\begin{pmatrix} U\boldsymbol{R} & \mathbf{V} \\ \mathbf{0}^T & W \end{pmatrix},$$

with U a scalar, may be factorized according to

$$\begin{pmatrix} U\boldsymbol{R} & \mathbf{V} \\ \mathbf{0}^T & W \end{pmatrix} \simeq \begin{pmatrix} U\boldsymbol{I} & \mathbf{0} \\ \mathbf{0}^T & W \end{pmatrix}\begin{pmatrix} U\boldsymbol{I} & \mathbf{V} \\ \mathbf{0}^T & U \end{pmatrix}\begin{pmatrix} U\boldsymbol{R} & \mathbf{0} \\ \mathbf{0}^T & U \end{pmatrix}$$

and also that multiplying

$$\begin{pmatrix} U\boldsymbol{R} & \mathbf{V} \\ \mathbf{0}^T & W \end{pmatrix}$$

by an isotropic scaling matrix, a rotation, or a translation, either on the left or on the right, yields a product matrix of the same form, and its inverse

$$\begin{pmatrix} W\boldsymbol{R}^T & -\boldsymbol{R}^T\mathbf{V} \\ \mathbf{0}^T & U \end{pmatrix}$$

is also of this form, *i.e.* any combination of these three operations in any order may be reduced by the above factorization to a rotation about the original origin, a translation (which defines a new origin) and an expansion or contraction about the new origin, applied in that order.

If

$$\begin{pmatrix} N\boldsymbol{R} & \mathbf{0} \\ \mathbf{0}^T & N \end{pmatrix}$$

is a rotation matrix as in Section 3.3.1.3.3, its application produces a rotation about an axis through the origin defined only in the space in which it is applied. For example, if

$$\boldsymbol{R} = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$\boldsymbol{T}'\begin{pmatrix} \mathbf{X} \\ W \end{pmatrix} = \boldsymbol{T}\begin{pmatrix} N\boldsymbol{R} & \mathbf{0} \\ \mathbf{0}^T & N \end{pmatrix}\begin{pmatrix} \mathbf{X} \\ W \end{pmatrix}$$

rotates the image about the z axis of data space, whatever the prevailing viewing transformation, $\boldsymbol{T}$.

Forming

$$\begin{pmatrix} N\boldsymbol{R} & \mathbf{0} \\ \mathbf{0}^T & N \end{pmatrix}\boldsymbol{T}\begin{pmatrix} \mathbf{X} \\ W \end{pmatrix}$$

rotates the image about the z axis of display space, *i.e.* the normal to the tube face under the usual conventions, whatever the prevailing $\boldsymbol{T}$. Furthermore, if this rotation is to appear to be about some chosen position in the picture, *e.g.* the centre, then the window transformation $\boldsymbol{U}$, Section 3.3.1.3.5, must place the origin of display space there by setting $F > S = R + L = T + B = 0 > N$, in the notation of that section.

If a rotation is to be about a point

$$\begin{pmatrix} \mathbf{V} \\ N \end{pmatrix}$$

then

$$\begin{aligned}\boldsymbol{T}' &= \begin{pmatrix} N\mathbf{I} & \mathbf{V} \\ \mathbf{0}^T & N \end{pmatrix}\begin{pmatrix} N'\boldsymbol{R} & \mathbf{0} \\ \mathbf{0}^T & N' \end{pmatrix}\begin{pmatrix} N\boldsymbol{I} & -\mathbf{V} \\ \mathbf{0}^T & N \end{pmatrix}\boldsymbol{T} \\ &\simeq \begin{pmatrix} N\boldsymbol{R} & \mathbf{V}-\boldsymbol{R}\mathbf{V} \\ \mathbf{0}^T & N \end{pmatrix}\boldsymbol{T}\end{aligned}$$

or

$$\begin{aligned}\boldsymbol{T}' &= \boldsymbol{T}\begin{pmatrix} N\boldsymbol{I} & \mathbf{V} \\ \mathbf{0}^T & N \end{pmatrix}\begin{pmatrix} N'\boldsymbol{R} & \mathbf{0} \\ \mathbf{0}^T & N' \end{pmatrix}\begin{pmatrix} N\boldsymbol{I} & -\mathbf{V} \\ \mathbf{0}^T & N \end{pmatrix} \\ &\simeq \boldsymbol{T}\begin{pmatrix} N\boldsymbol{R} & \mathbf{V}-\boldsymbol{R}\mathbf{V} \\ \mathbf{0}^T & N \end{pmatrix}\end{aligned}$$

according to whether $\boldsymbol{R}$ and $\mathbf{V}$ are both defined in display space or both in data space. If the rotation is defined in display space and the position of the centre of rotation is defined in data space, then the first form of $\boldsymbol{T}'$ must be used, in which $\mathbf{V}$ is first computed from

$$\begin{pmatrix} \mathbf{V} \\ N \end{pmatrix} = \boldsymbol{T}\begin{pmatrix} \mathbf{U} \\ W \end{pmatrix}$$

for a rotation centre at

$$\begin{pmatrix} \mathbf{U} \\ W \end{pmatrix}$$

in data space.

For continuous rotations defined in display space it is usually worthwhile to bring the centre of rotation to the origin of display space and leave it there, *i.e.* to omit the left-most factor in the first expression for $\boldsymbol{T}'$. Incremental rotations can then be made by further rotational factors on the left without further attention to $\mathbf{V}$. When continuous rotations are implemented by repeated multiplication of $\boldsymbol{T}$ by a rotation matrix, say thirty times a second for a minute or so, the orthogonality of the top-left partition of $\boldsymbol{T}$ may become degraded by accumulation of round-off error and this should be corrected occasionally by one of the methods of Section 3.3.1.2.3.

It is sometimes a requirement, depending on hardware capabilities, to affect a transformation in display space when access to data space is all that is readily available. In such a case

$$\boldsymbol{T}' = \boldsymbol{T}_1\boldsymbol{T} = \boldsymbol{T}\boldsymbol{T}_2,$$

where $\boldsymbol{T}_1$ is the required alteration to the prevailing viewing transformation $\boldsymbol{T}$ and $\boldsymbol{T}_2$ is the data-space equivalent,

$$\begin{aligned}\boldsymbol{T}_2 = \boldsymbol{T}^{-1}\boldsymbol{T}_1\boldsymbol{T} &= \begin{pmatrix} U\boldsymbol{R} & \mathbf{V} \\ \mathbf{0}^T & W \end{pmatrix}^{-1}\begin{pmatrix} U_1\boldsymbol{R}_1 & \mathbf{V}_1 \\ \mathbf{0}^T & W_1 \end{pmatrix}\begin{pmatrix} U\boldsymbol{R} & \mathbf{V} \\ \mathbf{0}^T & W \end{pmatrix} \\ &\simeq \begin{pmatrix} UU_1\boldsymbol{R}^T\boldsymbol{R}_1\boldsymbol{R} & \boldsymbol{R}^T(U_1\boldsymbol{R}_1\mathbf{V} + W\mathbf{V}_1 - \boldsymbol{W}_1\mathbf{V}) \\ \mathbf{0}^T & UW_1 \end{pmatrix}.\end{aligned}$$

An important special case is when $\boldsymbol{T}_1$ is to effect a rotation about the origin of display space without change of scale, so that $\mathbf{V}_1 = 0, U_1 = W_1 = W$, for then

$$\boldsymbol{T}_2 \simeq \begin{pmatrix} U\boldsymbol{R}^T\boldsymbol{R}_1\boldsymbol{R} & \boldsymbol{R}^T(\boldsymbol{R}_1 - \boldsymbol{I})\mathbf{V} \\ \mathbf{0}^T & U \end{pmatrix}.$$

If $\mathbf{r}$ is the required axis of rotation of $\boldsymbol{R}_1$ in display space then the axis of rotation of $\boldsymbol{R}^T\boldsymbol{R}_1\boldsymbol{R}$ in data space is $\mathbf{s} = \boldsymbol{R}^T\mathbf{r}$ since $\boldsymbol{R}^T\boldsymbol{R}_1\boldsymbol{R}\mathbf{s} = \mathbf{s}$. This gives a particularly simple result if $\boldsymbol{R}_1$ is to be a primitive rotation for then $\mathbf{s}$ is the relevant row of $\boldsymbol{R}$, and $\boldsymbol{R}^T\boldsymbol{R}_1\boldsymbol{R}$

can be constructed directly from this and the required angle of rotation.

3.3.1.3.9. *Inverse transformations*

It is frequently a requirement to be able to identify a feature or position in data space from its position on the screen. Facilities for identifying an existing feature on the screen are in many instances provided by the manufacturer as a 'hit' function which correlates the position indicated on the screen by the user (with a tablet or light pen) with the action of drawing and flags the corresponding item in the drawing internally as having been hit. In other instances it may be necessary to be able to indicate a position in data space independently of any drawn feature and this may be done by setting two or more non-parallel sight lines through the displayed volume and finding their best point of intersection in data space.

In Section 3.3.1.3.1 the relationship between data-space coordinates and screen-space coordinates was given as

$$\mathbf{S} = \boldsymbol{VUT}\mathbf{X};$$

hence data-space coordinates are given by

$$\mathbf{X} = \boldsymbol{T}^{-1}\boldsymbol{U}^{-1}\boldsymbol{V}^{-1}\mathbf{S}.$$

A line of sight through the displayed volume passing through the point

$$\begin{pmatrix} x \\ y \end{pmatrix}$$

on the screen is the line joining the two position vectors

$$\boldsymbol{S} = \begin{pmatrix} x & x \\ y & y \\ o & n \\ n & n \end{pmatrix}$$

in screen-space coordinates, as in Section 3.3.1.3.7, from which the corresponding two points in data space may be obtained using

$$\boldsymbol{V}^{-1} \simeq \begin{pmatrix} \dfrac{2n}{r-l} & 0 & 0 & \dfrac{-(r+l)}{(r-l)} \\ 0 & \dfrac{2n}{t-b} & 0 & -\dfrac{(t+b)}{(t-b)} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

and

$$\boldsymbol{U}^{-1} \simeq \begin{pmatrix} \dfrac{R-L}{2(S-E)} & 0 & \dfrac{-C(F-N)}{(F-E)(N-E)} & \dfrac{(R+L)(N-E)-2C(N-S)}{2(N-E)(S-E)} \\ 0 & \dfrac{T-B}{2(S-E)} & \dfrac{-D(F-N)}{(F-E)(N-E)} & \dfrac{(T+B)(N-E)-2D(N-S)}{2(N-E)(S-E)} \\ 0 & 0 & \dfrac{-E(F-N)}{(F-E)(N-E)} & \dfrac{N}{(N-E)} \\ 0 & 0 & \dfrac{-V(F-N)}{(F-E)(N-E)} & \dfrac{V}{(N-E)} \end{pmatrix}$$

in the notation of Section 3.3.1.3.5, and $\boldsymbol{T}^{-1}$ was given in Section 3.3.1.3.8. If orthographic projection is being used $(E = -\infty)$ then $\boldsymbol{U}^{-1}$ simplifies to

$$\boldsymbol{U}'^{-1} \simeq \begin{pmatrix} \dfrac{R-L}{2} & 0 & 0 & \dfrac{R+L}{2} \\ 0 & \dfrac{T-B}{2} & 0 & \dfrac{T+B}{2} \\ 0 & 0 & F-N & N \\ 0 & 0 & 0 & V \end{pmatrix}.$$

Each of these inverse matrices may be suitably scaled to suit the word length of the machine [Section 3.3.1.1.2 (iii)].

Having determined the end points of one sight line in data space the viewing transformation $\boldsymbol{T}$ may then be changed and the required position marked again through the screen in the new orientation. Each such operation generates a pair of points in data space, expressed in homogeneous form, with a variety of values for the fourth coordinate. Each such point must then be converted to three dimensions in the form $(X/W,\ Y/W,\ Z/W)$, and for each sight line any (three-dimensional) point $\mathbf{p}_A$ on the line and the direction $\mathbf{q}_A$ of the line are established. For each sight line a rank 2 projector matrix $\boldsymbol{M}_A$ of order 3 is formed as

$$\boldsymbol{M}_A = \boldsymbol{I} - \mathbf{q}_A\mathbf{q}_A^T/\mathbf{q}_A^T\mathbf{q}_A$$

and the best point of intersection of the sight lines is given by

$$\left(\sum_a \boldsymbol{M}_a\right)^{-1}\left(\sum_a \boldsymbol{M}_a\mathbf{p}_a\right),$$

to which three-vector a fourth coordinate of unity may be applied.

3.3.1.3.10. *The three-axis joystick*

The three-axis joystick is a device which depends on compound transformations for its exploitation. As it is usually mounted it consists of a vertical shaft, mounted at its lower end, which can rotate about its own length (the Y axis of display space, Section 3.3.1.3.1), its angular setting, φ, being measured by a shaft encoder in its mounting. At the top of this shaft is a knee-joint coupling to a second shaft. The first angle φ is set to zero when the second shaft is in some selected direction, *e.g.* normal to the screen and towards the viewer, and goes positive if the second shaft is moved clockwise when seen from above. The knee joint itself contains a shaft encoder, providing an angle, ψ, which may be set to zero when the second shaft is horizontal and goes positive when its free end is raised. A knob on the tip of the second shaft can then rotate about an axis along the second shaft, driving a third shaft encoder providing an angle θ. The device may then be used to produce a rotation of the object on the screen about an axis parallel to the second shaft through an angle given by the knob. The necessary transformation is then

$$\begin{aligned} \boldsymbol{R} = &\begin{pmatrix} \cos\varphi & 0 & -\sin\varphi \\ 0 & 1 & 0 \\ \sin\varphi & 0 & \cos\varphi \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{pmatrix} \\ &\times\begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi & \cos\psi \end{pmatrix} \\ &\times\begin{pmatrix} \cos\varphi & 0 & \sin\varphi \\ 0 & 1 & 0 \\ -\sin\varphi & 0 & \cos\varphi \end{pmatrix} \end{aligned}$$

which is

$$\begin{pmatrix} c^2\psi s^2\varphi + (1-c^2\psi s^2\varphi)c\theta & -s\psi c\psi s\varphi(1-c\theta) - c\psi c\varphi s\theta & -c^2\psi s\varphi c\varphi(1-c\theta) + s\psi s\theta \\ -s\psi c\psi s\varphi(1-c\theta) + c\psi c\varphi s\theta & s^2\psi + c^2\psi c\theta & s\psi c\psi c\varphi(1-c\theta) + c\psi s\varphi s\theta \\ -c^2\psi s\varphi c\varphi(1-c\theta) - s\psi s\theta & s\psi c\psi c\varphi(1-c\theta) - c\psi s\varphi s\theta & c^2\psi c^2\varphi + (1-c^2\psi c^2\varphi)c\theta \end{pmatrix}$$

in which cos and sin are abbreviated to c and s, which is the standard form with $l = -\cos\psi\sin\varphi$, $m = \sin\psi$, $n = \cos\psi\cos\varphi$.

3.3.1.3.11. *Other useful rotations*

If rotations in display space are to be controlled by trackerball or tablet then there are two measures available, an x and a y, which can define an axis of rotation in the plane of the screen and an angle θ. If x and y are suitably scaled coordinates of a pen on a tablet then the rotation

$$\begin{pmatrix} \dfrac{y^2+x^2c}{x^2+y^2} & \dfrac{-xy(1-c)}{x^2+y^2} & x\sqrt{x^2+y^2} \\ \dfrac{-xy(1-c)}{x^2+y^2} & \dfrac{x^2+y^2c}{x^2+y^2} & y\sqrt{x^2+y^2} \\ -x\sqrt{x^2+y^2} & -y\sqrt{x^2+y^2} & c \end{pmatrix}$$

with $c = \sqrt{1-(x^2+y^2)^2}$ is about an axis in the xy plane (*i.e.* the screen face) normal to (x,y) and with $\sin\theta = x^2+y^2$. Applied repetitively this gives a quadratic velocity characteristic. Similarly, if an atom at (x,y,z,w) in display space is to be brought onto the z axis by a rotation with its axis in the xy plane the necessary matrix, in homogeneous form, is

$$\begin{pmatrix} \dfrac{x^2z+y^2r}{x^2+y^2} & \dfrac{-xy(r-z)}{x^2+y^2} & -x & 0 \\ \dfrac{-xy(r-z)}{x^2+y^2} & \dfrac{x^2r+y^2z}{x^2+y^2} & -y & 0 \\ x & y & z & 0 \\ 0 & 0 & 0 & r \end{pmatrix}$$

with $r = \sqrt{x^2+y^2+z^2}$.

3.3.1.3.12. *Symmetry*

In Section 3.3.1.1.1 it was pointed out that it is usual to express coordinates for graphical purposes in Cartesian coordinates in ångström units or nanometres. Symmetry, however, is best expressed in crystallographic fractional coordinates. If a molecule, with Cartesian coordinates, is being displayed, and a symmetry-related neighbour is also to be displayed, then the data-space coordinates must be multiplied by

$$\begin{pmatrix} \boldsymbol{W} & \mathbf{T} \\ \boldsymbol{0}^T & W \end{pmatrix}\begin{pmatrix} \boldsymbol{M} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{pmatrix}\mathscr{S}\begin{pmatrix} \boldsymbol{M}^{-1} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{pmatrix}\begin{pmatrix} \boldsymbol{W} & -\mathbf{T} \\ \boldsymbol{0}^T & W \end{pmatrix},$$

where

$$\begin{pmatrix} \mathbf{T} \\ W \end{pmatrix}$$

are the data-space coordinates of the crystallographic origin, $\boldsymbol{M}$ and $\boldsymbol{M}^{-1}$ are as in Section 3.3.1.1.1 and $\mathscr{S}$ is a crystallographic symmetry operator in homogeneous coordinates, expressed relative to the same crystallographic origin.

For example, in $P2_1$ with the origin on the screw dyad along **b**,

$$\mathscr{S} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & \frac{1}{2} \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

and

$$\begin{pmatrix} \boldsymbol{M} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{pmatrix}\mathscr{S}\begin{pmatrix} \boldsymbol{M}^{-1} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{pmatrix} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & \frac{1}{2}b \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

$\mathscr{S}$ comprises a proper or improper rotational partition, $\boldsymbol{S}$, in the upper-left 3×3 in the sense that $\boldsymbol{MSM}^{-1}$ is orthogonal, and with the associated fractional lattice translation in the last column, with the last row always consisting of three zeros and 1 at the 4, 4 position. See *IT* A (1983, Chapters 5.3 and 8.1) for a fuller discussion of symmetry using augmented (*i.e.* 4×4) matrices.

3.3.1.4. *Modelling transformations*

The two sections under this heading are concerned only with the graphical aspects of conformational changes. Determination of such changes is considered under Section 3.3.2.2.

3.3.1.4.1. *Rotation about a bond*

It is a common requirement in molecular modelling to be able to rotate part of a molecule relative to the remainder about a bond between two atoms.

If four atoms are bonded 1–2–3–4 then the dihedral angle in the bond 2–3 is zero if the four atoms are *cis* planar, and a rotation in the 2–3 bond is, by convention (IUPAC–IUB Commission on Biochemical Nomenclature, 1970), positive if, when looking along the 2–3 bond, the far end rotates clockwise relative to the near end. This is valid for either viewing direction. This sign convention, when applied to the $\boldsymbol{R}$ matrix of Section 3.3.1.2.1, leads to the following statement.

If one of the two atoms is selected as the near atom and the direction cosines are those of the vector from the near atom to the far atom, and if the matrix is to rotate material attached to the far atom (with the reference axes fixed), then a positive rotation in the foregoing sense is generated by a positive θ.

Rotation about a bond normally involves compounding $\boldsymbol{R}$ with translations in the manner of Section 3.3.1.3.8.

3.3.1.4.2. *Stacked transformations*

A flexible molecule may require to be drawn in any of a number of conformations which are related to one another by, for example, rotations about single bonds, changes of bond angles or changes of bond lengths, all of which changes may be brought about by the application of suitable homogeneous transformations during the drawing of the molecule (Section 3.3.1.3.8). With suitable organization, this may be done without necessarily altering the coordinates of the atoms in the coordinate list, only the transformations being manipulated during drawing.

The use of transformations in the manner shown below is straightforward for simply connected structures or structures containing only rigid rings. Flexible rings may be similarly handled provided that the matrices employed are consistent with the consequential constraints as described in Section 3.3.2.2.1, though this requirement may make real-time folding of flexible rings difficult.

Any simply connected structure may be organized as a tree with a node at each branch point and with an arbitrary number of sites of

conformational change between one node and the next. We shall call such sites and their associated matrices 'conformons'. The technique then depends on the stacking technique in which matrices are stored and later recovered in the reverse order of their storage.

One begins at some reference point deemed to be fixed in data space and at this point one stacks the prevailing viewing transformation. From this reference point one advances through the molecule along the structural tree and as each conformon is encountered its matrix is calculated. The product of the prevailing matrix with the conformon matrix is formed and stacked, and this product becomes the prevailing matrix. This product is constructed with the conformon matrix as a factor on the right, *i.e.* in data space as defined in Section 3.3.1.3.1, and is calculated using the coordinates of the molecule in their unmodified form, *i.e.* before any shape changes are brought about.

This progression leads eventually to an extremity of the tree. At this point drawing is commenced using the prevailing matrix and working backwards towards the fixed root, unstacking (or 'popping') a matrix as each conformon is passed until a node is reached, which, in general, will occur only part way back to the root. On reaching such a node drawing is suspended and one advances along the newly found branch as before, stacking matrices, until another extremity is reached when drawing towards the root is resumed. This alternation of stacking matrices while moving away from the root and drawing and unstacking matrices while moving towards the root is continued until the whole tree is traversed.

This process is illustrated schematically in Fig. 3.3.1.2 for a simple tree with one node, numbered 1, and three conformons at *a*, *b* and *c*. One enters the tree with a current viewing transformation $\boldsymbol{T}$ and progresses upwards from the fixed lower extremity. When the conformon at *a* is encountered, $\boldsymbol{T}$ is stacked and the product $\boldsymbol{TM}_a$ is formed. Continuing up the tree, at node 1 either branch may be chosen; we choose the left and, on reaching *b*, $\boldsymbol{TM}_a$ is stacked and $\boldsymbol{TM}_a\boldsymbol{M}_b$ is formed. On reaching the tip drawing down to *b* is done with this transformation, $\boldsymbol{TM}_a$ is then unstacked and drawing continues with this matrix until node 1 is reached. The other branch is then followed to *c* whereupon $\boldsymbol{TM}_a$ is again stacked and the product $\boldsymbol{TM}_a\boldsymbol{M}_c$ is formed. From the tip down as far as *c* is drawn with this matrix, whereupon $\boldsymbol{TM}_a$ is unstacked and drawing continues down to *a*, where $\boldsymbol{T}$ is unstacked before drawing the section nearest the root.

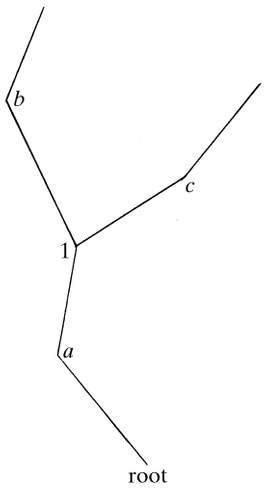


Fig. 3.3.1.2. Schematic representation of a simple branched-chain molecule with a stationary root and two extremities. The positions marked *a*, *b* and *c* are the loci of possible conformational change, here called conformons, and there is a single, numbered branch point.

With this organization the matrices associated with *b* and *c* are unaffected by changes in the conformation at *a*, notwithstanding the fact that changes at *a* alter the direction of the axis of rotation at *b* or *c*.

Two other approaches are also possible. One of these is to start at the tip of the left branch, replace the coordinates of atoms between *b* and the tip by $\boldsymbol{M}_b\mathbf{X}$, and later replace all coordinates between the tip and *a* by $\boldsymbol{M}_a\mathbf{X}$, with a similar treatment for the other branch. The advantage of this is that no storage is required for stacked matrices, but the disadvantage is that atoms near the tips of the tree have to be reprocessed for every conformon. It also modifies the stored coordinates, which may or may not be desirable.

The second alternative is to draw upwards from the root using $\boldsymbol{T}$ until *a* is reached, then using $\boldsymbol{TM}_a$ until *b* is reached, then using $\boldsymbol{TM}'_b\boldsymbol{M}_a$ to the tip, but in this formulation $\boldsymbol{M}'_b$ must be based on the geometry that exists at *b* after the transformation $\boldsymbol{M}_a$ has been applied to this region of the molecule, *i.e.* $\boldsymbol{M}'_b$ is characteristic of the final conformation rather than the initial one.

3.3.1.5. *Drawing techniques*

3.3.1.5.1. *Types of hardware*

There are two main types of graphical hardware in use for interactive work, in addition to plotters used for batch work. These main types are raster and vector. In raster equipment the screen is scanned as in television, with a grid of points, called pixels, addressed sequentially as the scan proceeds. Associated with each pixel is a word of memory, usually containing something in the range of 1 to 24 bits per pixel, which controls the colour and intensity to be displayed. Many computer terminals have one bit per pixel (said to be 'single-plane' systems) and these are essentially monochrome and have no grey scale. Four-plane systems are cheap and popular and commonly provide 4-bit by 4-bit look-up tables between the pixel memory and the monitor with one such table for each of the colours red, green and blue. If these tables are each loaded identically then 16 levels of monochrome grey scale are available, but if they are loaded differently 16 different colours are available simultaneously chosen from a total of 4096 possibilities. Four-plane systems are adequate for many applications where colour is used for coding, but are inadequate if colour is intended also to provide realism, where brilliance and saturation must be varied as well as hue. For these applications eight-plane systems are commonly used which permit 256 colours chosen from 16 million using three look-up tables, though the limitations of these can also be felt and full colour is only regarded as being available in 24-plane systems.

Raster-graphics devices are ideal for drawing objects represented by opaque surfaces which can be endowed with realistic reflecting properties (Max, 1984) and they have been successfully used to give effects of transparency. They are also capable of representing shadows, though these are generally difficult to calculate (see Section 3.3.1.5.5). Many devices of this type provide vectorization, area fill and anti-aliasing. Vectorization provides automatically for the loading of relevant pixels on a straight line between specified points. Area fill automatically fills any irregular pre-defined polygon on the screen with a uniform colour with the user specifying only the colour and one point within the polygon. Anti-aliasing is the term used for a technique which softens the staircase effect that may be seen on a line which runs at a small angle to a vertical or horizontal row of pixels.

The main drawback with this type of equipment is that it is slow compared to vector machines. Only relatively simple objects can be displayed with smooth rotation in real time as transformed coordinates have to be converted to pixel addresses and the

previous frame needs to be deleted with each new frame unless it is known that each new frame will specify every pixel. However, the technology is advancing rapidly and these restrictions are already disappearing.

Vector machines, on the other hand, are specialized to drawing straight lines between specified points by driving the electron beam along such lines. No time is wasted on blank areas of the screen. Dots may be drawn with arbitrary coordinates, in any order, but areas, if they are to be filled, must be done with a ruling technique which is very seldom done. Images produced by vector machines are naturally transparent in that foreground does not obscure background, which makes them ideal for seeing into representations of molecular structure.

3.3.1.5.2. *Optimization of line drawings*

A line drawing consisting of n line segments may be specified by anything from $(n + 1)$ to $2n$ position vectors depending on whether the lines are end-to-end connected or independent. Appreciable gains in both processing time and storage requirements may be made in complicated drawings by arranging for line segments to be end-to-end connected as far as possible, and an algorithm for doing this is outlined below. For further details see Diamond (1984*a*).

Supposing that a list of nodal points (atoms if a covalent skeleton is being drawn) exists within a computer with each node appearing only once and that the line segments to be drawn between them are already determined, then at each node there are, generally, both forward and backward connections, forward connections being those to nodes further down the list. A quantity D is calculated at each node which is the number of forward connections minus the number of backward connections. At the commencement of drawing, the first connected node in the list must have a positive D, the last must have a negative D, the sum of all D values must be zero and the sum of the positive ones is the number of strokes required to draw the drawing, a 'stroke' being a sequence of end-to-end connected line segments drawn without interruption. The total number of position vectors required to specify the drawing is then the number of nodes plus the number of strokes plus the number of rings minus one.

Drawing should then be done by scanning the list of nodes from the top looking for a positive D (usually found at the first node), commencing a stroke at this node and decrementing its D value by 1. This stroke is continued from node to node using the specified connections until a negative D is encountered, at which point the stroke is terminated and the D value at the terminating node is incremented by 1. This is done even though this terminating node may also possess some forward connections, as the total number of strokes required is not minimized by keeping a stroke going as far as possible, but by terminating a stroke as soon as it reaches a node at which *some* stroke is bound to terminate.

The next stroke is initiated by resuming the scan for positive D values at the point in the node list where the previous stroke began. If this scan encounters a zero D value at a node which has not hitherto been drawn to, or drawn from, then the node concerned is isolated and not connected to any other, and such nodes may require to be drawn with some special symbol. The expression already given for the number of vectors required is valid in the presence of isolated nodes if drawing an isolated node is allowed one position vector, this vector not being counted as a stroke.

The number of strokes generated by this algorithm is sensitive to the order in which the nodes are listed, but if this resembles a natural order then the number of strokes generated is usually close to the minimum, which is half the number of nodes having an odd number of connections. For example, the letter E has six nodes, four of which have an odd number of connections, so it may be drawn with two strokes.

3.3.1.5.3. *Representation of surfaces by lines*

The commonest means of representing surfaces, especially contour surfaces, is to consider evenly spaced serial sections and to perform two-dimensional contouring on each section. Repeating this on serial sections in two other orientations then provides a good representation of the surface in three dimensions when all such contours are displayed. The density is normally cited on a grid with submultiples of **a**, **b** and **c** as grid vectors, inverse linear interpolation being used between adjacent grid points to locate points on the contour. For vector-graphics applications it is expedient to connect such points with straight lines; some equipment may be capable of connecting them with splines though this is burdensome or impossible if real-time rotation of the scene is required. Precalculation of splines stored as short vectors is always possible if the proliferation of vectors is acceptable. For efficient drawing it is necessary for the line segments of a contour to be end-to-end connected, which means that it is necessary to contour by following contours wherever they go and not by scanning the grid. Algorithms which function in this way have been given by Heap & Pink (1969) and Diamond (1982*a*). Contouring by grid scanning followed by line connection by the methods of the previous section would be possible but less efficient. Further contouring methods are described by Sutcliffe (1980) and Cockrell (1983).

For raster-graphics devices there is little disadvantage in using curved contours though many raster devices now have vectorizing hardware for loading a line of pixels given only the end points. For these devices well shaped contours may be computed readily, using only linear arithmetic and a grid-scanning approach (Gossling, 1967). Others have colour-coded each pixel according to the density, which provides a contoured visual impression without performing contouring (Hubbard, 1983).

3.3.1.5.4. *Representation of surfaces by dots*

Connolly (Langridge *et al*., 1981; Connolly, 1983*a*,*b*) represents surfaces by placing dots on the surface with an approximately uniform superficial density. Connolly's algorithm was developed to display solvent-accessible surfaces of macromolecules and provides for curved concave portions where surface atoms meet. Pearl & Honegger (1983) have developed a similar algorithm, based on a grid, which generates only convex portions which meet in cusps, but is faster to compute than the Connolly surface. Bash *et al.* (1983) have produced a van der Waals surface algorithm fast enough to permit real-time changes to the structure without tearing the surface.

It has become customary to use a dot representation to display computed surfaces, such as the surface at a van der Waals radius from atomic centres, and to use lines to represent experimentally determined surfaces, especially density contours.

3.3.1.5.5. *Representation of surfaces by shading*

Many techniques have been developed, mainly for raster-graphics devices, for representing molecular surfaces and these have been very well reviewed by Max (1984).

The simplest technique in this class consists in representing each atom by a uniform disc, or high polygon, which can be colour-coded and area-filled by the firmware of the device. If such atoms are sorted on their z coordinate and drawn in order, furthest ones first, so that nearer ones partly or completely overwrite the further ones then the result is a simple representation of the molecule as seen from the front. This technique is fast and has its uses when a rapid schematic is all that is required. In one sense it is wasteful to process distant atoms when they are going to be overwritten by foreground atoms, but front-to-back processing requires the boundaries of visible parts

of partially obscured atoms near the front to be determined before they can be painted or, alternatively, every pixel must be tested before loading to see if it is already loaded. Not only does this approach give a uniform rendering over the whole area of one atom, it also gives a boundary between overlapping atoms with almost equal z values which completes the circle of the nearer atom, though it should be an arc of an ellipse when the atoms are drawn with radii exceeding their covalent radii.

Greater realism is achieved by establishing a z buffer, which is an additional area of memory with one word per pixel, in which is stored the z value of the currently loaded feature in each pixel. Treatments which take account of the sphericity are then possible and correct arcs of intersection for interpenetrating spheres and more complicated entities arise naturally through loading a colour value into a pixel only if the z coordinate is less than that of the currently loaded value. This z buffer and the associated x, y coordinates should be in picture space or screen space rather than display space since only after the application of perspective can points with the same x/w and y/w coordinates obscure one another.

It is usual in such systems to vary the intensity of colour within one atom by darkening it towards the circumference on the basis of the z coordinate. Some systems augment this impression of sphericity by highlighting. The simplest form of highlighting is an extension of the uniform disc treatment in which additional, brighter discs, possibly off centre, are associated with each atom. More general highlighting (Phong, 1975) is computed from four unit vectors, these being the normal to the surface, the direction to a light source, the direction to the viewer and the normalized vector sum of these last two. Intensity levels may then be set as the sum of three terms: a constant, a term proportional to the scalar product of the first two vectors (if positive) and a term proportional to a high power of the scalar product of the first and last vectors; the higher the power the glossier the surface appears to be. This final term normally adds a white term, rather than the surface colour, supposing the light source to be white.

Shadows may also be rendered to give even greater realism. In addition to the z buffer and (x, y) frame buffer a second z buffer for z' values associated with x' and y' is also required. These coordinates are then related by $x' = x + \alpha z$, $y' = y + \beta z$, $z' = z$. The second buffer is a ray buffer since $x'y'$ are the coordinates with which an illuminating ray passing through (xyz) passes through the $z = 0$ plane, and z', stored at x', y', records the depth at which this ray encounters material. Thus any two pixels $(x_1y_1z_1)$ and $(x_2y_2z_2)$ are on the same illuminating ray if their x' and y' values are equal and the one with smaller z' shadows the other. Processing a pixel at $(x_1y_1z_1)$ therefore involves first determining its visibility on the basis of the z buffer, as before, then, whether or not it is visible, setting $z'_1 = z_1$ and considering the value of z' currently stored at $x'y'$, which we call z'_2.

If $z'_1 < z'_2$ then $x_1y_1z_1$ is in light and must be loaded accordingly. From z'_2 we find the previously processed pixel $(x_2y_2z_2)$ which is now in shade and which was in light when originally processed, so that the colour value stored at x_2y_2 needs to be altered *unless* the pixel at x_2y_2 is now $(x_2y_2z_3)$ with $z_3 < z_2$, in which case the pixel $(x_2y_2z_2)$ which has now become shadowed by $(x_1y_1z_1)$ has, in the meantime, been obscured by $(x_2y_2z_3)$ which is not shadowed by $(x_1y_1z_1)$ and no change is therefore needed. In either event z'_1 then replaces z'_2.

If $z'_1 > z'_2$ then $(x_1y_1z_1)$, if visible, is in shade and must be coloured accordingly, and in this case z'_2 is not superseded.

This shadowing scheme corresponds to illumination by a light source at infinity in picture space or, equivalently, with a z coordinate equal to that of the eye in display space. For its implementation x, y and z may be in any convenient coordinate system, *e.g.* pixel addresses, but if x and y are expressed with the range -1 to 1 and z with the range 0 to 1 corresponding to the window then they may be identified as the quantities $x/w, y/w$ and z/w of picture space (Section 3.3.1.3.1).

If, in the notation of Section 3.3.1.3.5, the light source is placed at (P, Q, E, V) in display space and a ray leaves it in the direction (p, q, r, V) then

$$x' = \frac{p}{r} \cdot \frac{2(S-E)}{(R-L)} + \frac{2(S-E)(P-C)}{(N-E)(R-L)} + \frac{2C-R-L}{R-L},$$

which varies only with beam direction,

$$\alpha = \frac{2(S-E)(F-N)(P-C)}{(F-E)(N-E)(R-L)}$$

and similarly for y' and β.

3.3.1.5.6. *Advanced hidden-line and hidden-surface algorithms*

Hidden surfaces may be handled quite generally with the z-buffer technique described in the previous section but this technique becomes very inefficient with very complicated scenes. Faster techniques have been developed to handle computations in real time (*e.g.* 25 frames s^{-1}) on raster machines when both the viewpoint and parts of the environment are moving and substantial complexity is required. These techniques generally represent surfaces by a number of points in the surface, connected by lines to form panels. Many algorithms require these panels to be planar and some require them to be triangular. Of those that permit polygonal panels, most require the polygons to be convex with no re-entrant angles. Yet others are limited to cases where the objects themselves are convex. Some can handle interpenetrating surfaces, others exclude these. Some make enormous gains in efficiency if the objects in the scene are separable by the insertion of planes between them and degrade to lower efficiency if required, for example, to draw a chain. Some are especially suited to vector machines and others to raster machines, the latter capitalizing on the finite resolution of such systems. In all of these the basic entities for consideration are entire panels or edges, and in some cases vertices, point-by-point treatment of the entire surface being avoided until after all decisions are made concerning what is or is not visible.

All of these algorithms strive to derive economies from the notion of 'coherence'. The fact that, in a cine context, one frame is likely to be similar to the previous frame is referred to as 'frame coherence'. In raster scans line coherence also exists, and other kinds of coherence can also be identified. The presence of any form of coherence may enable the computation to be concerned primarily with changes in the situation, rather than with the totality of the situation so that, for example, computation is required where one edge crosses in front of another, but only trivial actions are involved so long as scan lines encounter the projections of edges in the same order.

The choice of technique from among many possibilities may even depend on the viewpoint if the scene has a statistical anisotropy. For example, the depiction of a city seen from a viewpoint near ground level involves many hidden surfaces. Distant buildings may be hidden many times over. The same scene depicted from an aerial viewpoint shows many more surfaces and fewer overlaps. This difference may swing the balance of advantage between an algorithm which sorts first on z or one which leaves that till last.

These advanced techniques have, so far, found little application in crystallography, but this may change. Ten such techniques are critically reviewed and compared by Sutherland *et al.* (1974), and three of these are described in detail by Newman & Sproull (1973).

references