Journal of Computer-Aided Molecular Design, 9 (1995) 149–159
ESCOM

J-CAMD 283

SMART: A solvent-accessible triangulated surface generator for molecular graphics and boundary element applications*

Randy J. Zauhar**

Department of Molecular and Cell Biology and Biotechnology Institute, The Pennsylvania State University, 519 Wartik Laboratory, University Park, PA 16802, U.S.A.

Received 5 July 1994
Accepted 7 October 1994

Keywords: Molecular surface; Solvent-accessible surface; Triangulation; Algorithm; Boundary element methods

Summary

An algorithm is presented for generating a representation of the solvent-accessible molecular surface as a smooth triangulated manifold. The algorithm, called SMART (*SM*ooth molecul*AR* surface *T*riangulator), divides the contact and reentrant portions of the solvent-accessible molecular surface into curvilinear three-sided elements. In contrast to the author's earlier implementation of this general approach [Zauhar, R.J. and Morgan, R.S., J. Comput. Chem., 11 (1990) 603], the SMART algorithm defines elements directly on the appropriate geometric surface types (rather than using interpolation over cubic elements), and has special features to handle highly distorted regions which often appear in deep crevices and internal cavities. While the method is designed for use with boundary element techniques in continuum electrostatics, it can also be applied to the accurate computation of molecular surface areas and volumes, and the generation of shaded surfaces for display with interactive computer graphics.

Introduction

In the past years, a wide variety of methods have been developed to model and display the surfaces and volumes of molecules [1–23]. Applications of these techniques range from the generation of dot and shaded surfaces for computer graphics display to the accurate computation of solvent-accessible surface areas for use in conformational energy calculations. Some of the available techniques treat the molecule as a collection of interlocking spheres, representing individual atoms, and take the molecular surface as the union of the exposed regions of these spheres. The spheres may be assigned the radii of their associated atom types (thus creating a van der Waals surface), or may be assigned atomic radii that are augmented by the solvent radius (leading to Lee and Richard's version of the solvent-accessible surface [1]).

In an alternative approach, first proposed by Richards [2] and later extended by Connolly [5], the molecule is again represented by interlocking spheres, but the molecular *surface* is defined by the action of a spherical probe which rolls over the molecule. (Here the probe represents a single solvent.) In those regions where the probe has a single point of contact with the molecule, the molecular surface is taken to be the van der Waals surface. Where the probe has two points of contact (i.e., when rolling along the crevice between two adjacent atoms), the molecular surface is defined by the saddle section generated by the inward face of the moving probe. Finally, where the probe has three or more points of simultaneous contact, the molecular surface is taken as the spherical polygon on the probe surface defined by the contact points. The three varieties of surface are termed 'contact', 'concave reentrant' and 'saddle', respectively. The concave reentrant and saddle regions together form the 'reentrant surface'.

The available methods for generating molecular surfaces can be further categorized by the level of detail used in the surface description. For example, 'dot surface' representations define the surface only at isolated points [5]. These algorithms provide minimal detail, and are

**Availability:* Programs (in C) for surface generation, area and volume computation are available from the author. Also available is a graphics display program which runs on Silicon Graphics workstations.

**Present address: TRIPOS, Inc., 1699 S. Hanley Road, Suite 303, St. Louis, MO 63144, U.S.A.

primarily intended for graphical presentation and the computation of approximate areas and volumes. At the other extreme, algorithms which triangulate the molecular surface [6,9,15,17,23] effectively assign coordinates to every point on the (continuous) surface, and can be used to make highly accurate estimates of molecular surface areas and volumes. They are also appropriate for use with boundary element methods which have been developed to compute electrostatic and hydration phenomena within the continuum approximation [4,7,9,15,24,25].

In this article, a new algorithm called SMART (for *SM*ooth molecul*AR* surface *T*riangulator) is described for triangulating the solvent-accessible surface into curvilinear elements. Unlike the triangulation method of Connolly [6], which constructs a polyhedral approximation to the curved surface, SMART works directly on the appropriate surfaces (whether spherical or saddle). In contrast to the author's earlier MOLSURF program [15], which used cubic elements to interpolate over the surface, the interpolation method in SMART retains the 'exact' geometries of the contact and reentrant regions. SMART also uses a more flexible method than MOLSURF to partition the surface, and includes special handling for very narrow contact regions. (These 'pinched' regions are often found in deep crevices and internal cavities, and sometimes caused the older algorithm to fail when triangulating proteins.) Moreover, a new technique is introduced to avoid the presence of self-intersecting surface. The SMART program can generate a surface decomposition that provides an accurate representation of the surface with a small number of elements; it is thus ideally suited for boundary element applications, where it is imperative to limit the number of equations associated with the surface description.

Methods

Input

The algorithm is supplied with a molecule name, a name of an atomic radii set, a probe name and an angle parameter (discussed below). SMART expects an atomic coordinate file <molecule name>.dat, a file <radii name>.rad containing a table of atomic radii, and a file <probe name>.dat from which the probe radius is obtained. The coordinate file format is the same as that used with Connolly's MS program [26], and includes a type index for each atom. The formats of the other files are described in the documentation that accompanies the SMART program.

Probe placement

The SMART algorithm begins by locating all positions where a probe can simultaneously make contact with three or more atoms. Each probe q has an associated set of contact atoms $\{c_{q1},c_{q2},c_{q3}\}$. Also, each atom has an attached list of probes that make contact with it, and the pointer to a newly generated probe is appended to the probe list of each of its contact atoms. (So, given a probe one can immediately locate all atoms it contacts, and given an atom one can find all probes it touches.) If the probe q lies within a specified cutoff distance (taken here as 1×10^{-6} unit) to a previously generated probe p, then q is *not* added to the list of probes, but the contact atom set of p ($\{c_{p1,2,3}\}$) is replaced by the union of the contact sets of p and q. (Thus, probe q is 'identified' with probe p.) The purpose of this operation is to avoid the generation of saddle regions (in the next step of the algorithm) of vanishing width. While the presence of 'superimposed' probes is rare when working with crystallographic coordinates, they frequently occur with model-built structures that include rings.

Identification of saddle sections

The next step is the location of all saddle regions of the reentrant molecular surface. This is accomplished by considering in turn each unique pair of neighboring atoms i and j (i < j), and finding all probes that simultaneously contact both atoms. Those probes that are free to rotate into each other without bumping into other atoms are grouped in pairs, and the probes in each pair define the

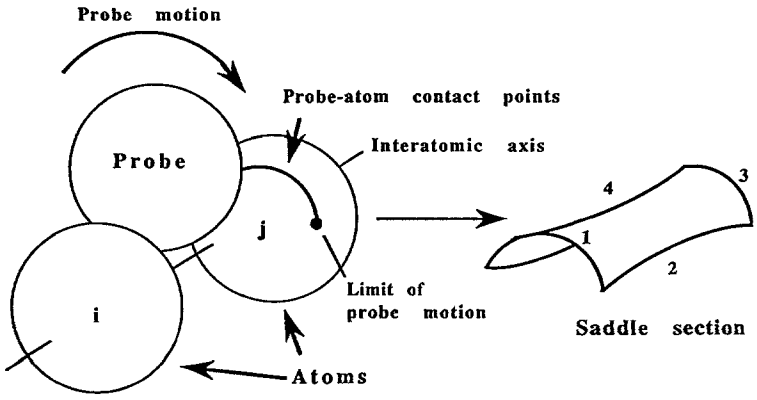


Fig. 1. Generation of a saddle section by a rolling probe. The limits of probe motion are determined by contacts with atoms other than i and j (not shown).

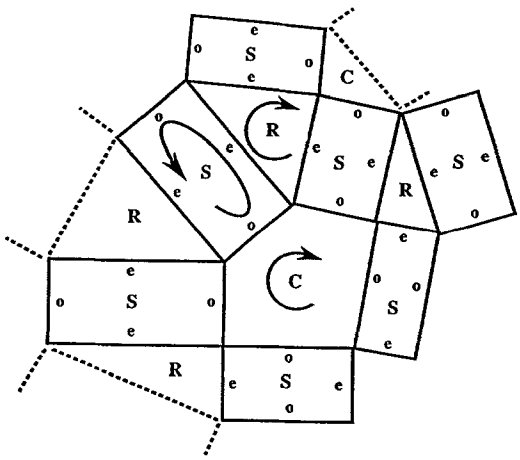


Fig. 2. Illustration of how contact and reentrant regions are formed. The rectangles labeled S represent saddle sections – they are assigned a counterclockwise orientation (looking from outside the molecule). Odd edges (labeled o) outline contact regions (C) on atoms; even edges (e) outline concave reentrant regions (R) on probes.

limits of a single saddle section (which is geometrically a section of a torus). Each saddle section is defined by the radius and center of probe rotation, the radii of the contact atoms and the angular range through which the probe can move. Figure 1 illustrates the definition of a saddle section.

The edges of the saddle region are numbered 1–4, with the odd-numbered edges defined by the point of contact between the probe and one of the atoms. Each odd-numbered edge is thus associated with a particular atom. The odd edges are sections of circular arcs, with the arc centers located on the axis between the contact atoms. (The locations of the arc centers and their radii depend upon the size and separation of the contact atoms and the radius of the solvent probe.) The even-numbered edges lie in the plane defined by the probe and the two contact atoms when the probe is at one of the limits of its motion; edges of this class span the space between the two contact atoms. Even-numbered edges are described as great circles on the surface of the probe.

The algorithm creates a linked list which relates each atom to all the saddle sections that include the atom in their definition. Similarly, lists are established for the probes, so that given a probe index, all of the saddle sections that the probe delimits can be quickly identified.

Location of contact and concave reentrant regions

As illustrated in Fig. 2, contact regions (where the probe can slide freely on the van der Waals surface) are outlined by cycles of odd-numbered saddle edges. Concave reentrant regions are defined by the probe when it simultaneously rests against three or more atoms, and are outlined by cycles of even-numbered saddle edges. The algorithm considers each atom in turn, and finds all saddle sections that contact the atom. The odd edges that outline contact regions on the atom surface are arranged into cycles with a clockwise orientation. In the same way, each probe is considered and all the saddle sections that the probe delimits are identified; the even edges of these sections are then arranged into a concave reentrant cycle with *counter*clockwise orientation on the surface of the probe. The combination of clockwise orientations for contact cycles and counterclockwise orientations for concave reentrant cycles is consistent with a counterclockwise orientation for the four edges of each saddle section.

After arcs are arranged into cycles, they are subdivided in accordance with a user-supplied angle parameter. If an arc subtends an angle that is more than twice this parameter, then it is subdivided into n smaller arcs, where n is the integer result of dividing the arc angle by the parameter. (Thus, if an arc subtends an angle of 100°, and the angle parameter is 30°, then the initial arc will be subdivided into three arcs, each making an angle of 33.3°.) The ends of the arcs (edges) are identified as numbered vertices. The subdivided cycles are then triangulated, as described in the following sections.

Contact triangulation

The next step is to partition the contact regions into three-sided curvilinear elements. As in Connolly's algorithm [6], this is accomplished by recursive subdivision of the initial contact cycles, as shown in Fig. 3. By the addition of an edge to an initial n-cycle, an l- and an m-cycle are produced (with $n = l + m - 2$). Continuing in this way, the l- and m-cycles can be further reduced in size, and the process iterated until all cycles are of size 3. However,

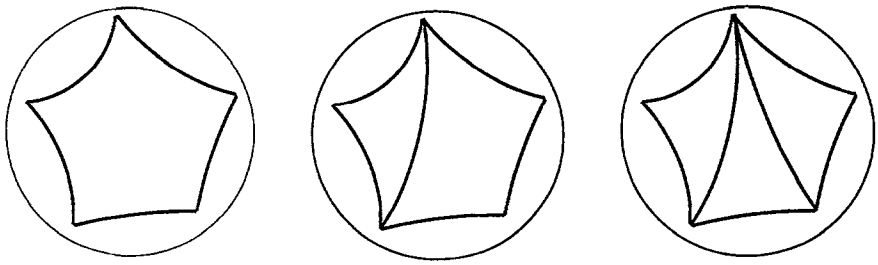


Fig. 3. Triangulation by recursive subdivision of a starting cycle. The circle represents a single atom or probe.

while the Connolly algorithm generates a polygonal approximation of the surface made up of planar triangles, the SMART algorithm subdivides cycles with edges that are generalized circular arcs, and creates a triangulation that lies on the original surface. The deficiencies of the Connolly algorithm for boundary element applications have been stated elsewhere [15]; here, it suffices to point out that the Connolly approach requires a very large number of triangles to accurately model the highly irregular contact regions that frequently appear in surfaces of macromolecules (and even, occasionally, in small molecules). By directly triangulating on the surface with curved arcs, the SMART algorithm has the geometric flexibility to partition the surface with an economy of elements.

Figure 4 illustrates the method used to generate circular arcs that span an initial contact cycle. An arc is uniquely defined by two vertices on the spherical surface and the tangent vector to the arc at one vertex. Given this information, it is a straightforward application of linear algebra to locate the center of the arc, its radius and the vectors directed from the center to the end points of the arc. The center of the arc will coincide with the sphere center if and only if the arc is a geodesic (great circle). The tangent vector at one vertex can be visualized as the generator of the arc, which must lie in the plane defined by the tangent vector and the line joining the vertices.

The generation of four possible 'candidate' arcs between opposing vertices in a contact cycle is illustrated in Figs. 5A and B. Here, the two vertices (a and b) are each associated with two pre-existing edges in the initial contact cycle, and these edges in turn define two tangent vectors ($\mathbf{a}_1$ and $\mathbf{a}_2$, $\mathbf{b}_1$ and $\mathbf{b}_2$) that lie in the tangent plane at each vertex. Since we have two end points and two tangent vectors at each end point, four circular arcs are defined, and we label the spanning arcs A_{a_1}, A_{a_2}, A_{b_1} and A_{b_2}, in accordance with the specific tangent vector used to construct each arc. In the diagram, the spanning arcs are assigned an orientation, and are taken to start at the vertex whose tangent plane contains the vector which is used to define the arc.

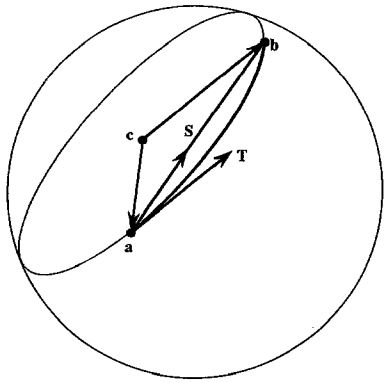


Fig. 4. Definition of a general circular arc on the surface of an atom or probe. The vector **S** is directed from vertex a to b, **T** is a tangent vector in the tangent plane at a. Together, **S** and **T** define a plane cutting through the sphere; the circular arc lies in the intersection of this plane with the sphere (atom or probe). c is the center of the arc, which will coincide with the sphere center only if the arc is a geodesic.

At vertex a, the spanning arcs A_{b_1} and A_{b_2} have associated tangent vectors *in the tangent plane of* a, which we denote $\mathbf{b}'_1$ and $\mathbf{b}'_2$. Any arc generated at b that lies within the contact region (and thus has a tangent vector within the angle formed by $\mathbf{b}_1$ and $\mathbf{b}_2$) will end at a with a tangent that lies within the smallest angle formed by $\mathbf{b}'_1$ and $\mathbf{b}'_2$. At the same time, all 'permissible' arcs starting from a must have tangent vectors that lie within the smallest angle formed by $\mathbf{a}_1$ and $\mathbf{a}_2$. Taking the intersection of the angular ranges $(\mathbf{a}_1,\mathbf{a}_2)$ and $(\mathbf{b}'_1,\mathbf{b}'_2)$, we define the *tangent window* at a, which delimits all arcs that can begin at a and remain inside the contact region (at least, near the end points). The concept is illustrated in Fig. 6. When generating an arc, the vector that bisects the tangent window is taken as the defining tangent vector.

The subdivision procedure attempts to connect all unique pairs of vertices within a starting cycle that are separated by at least one intervening vertex, and which have a non-vanishing tangent window at one of the vertices. A trial spanning arc is generated, and is immediately rejected if it intersects any pre-existing arc of the cycle

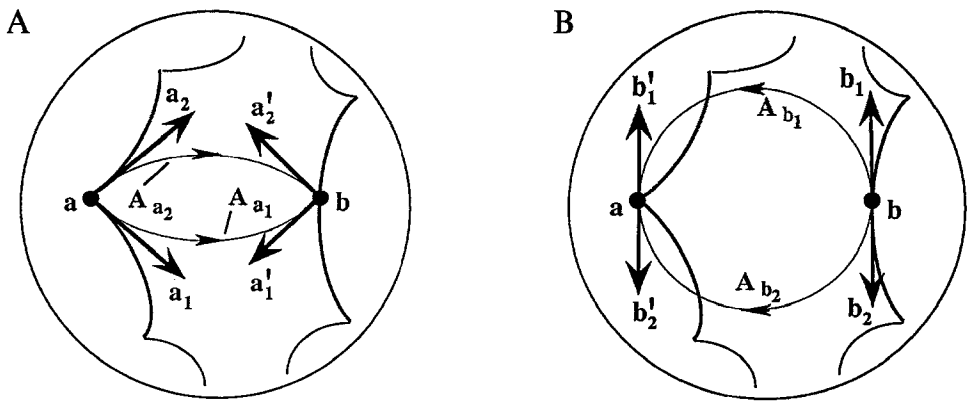


Fig. 5. 'Candidate' circular arcs connecting vertices a and b. The labeling of the arcs is described in the text.

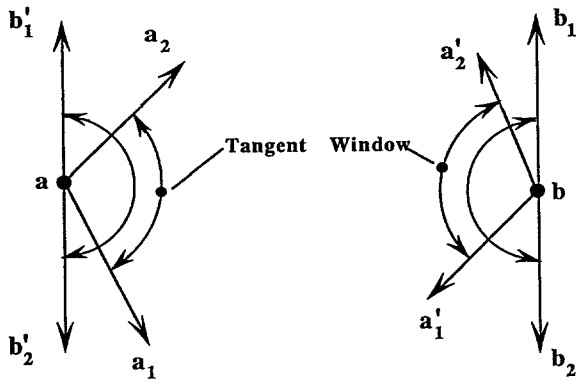


Fig. 6. Illustration of the tangent windows at vertices a and b. The two tangent planes are reoriented into the plane of the diagram. (The labeling of the tangent vectors is described in the text.) The tangent window is defined by the overlap of the angular range of permissible arcs that *originate* at a vertex, and the range of allowed arcs that *end* at the same vertex.

(except, of course, those that terminate at the end points), or if the distance of closest approach to any previous arc is less than 5% of the length of the trial arc. The latter test is a heuristic rule, used to minimize the occurrence of elements with highly distorted geometries. If a trial arc is acceptable, two 'quality factors' are computed. One factor is the smaller ratio of the perimeters of the two cycles that would be created by the addition of the new arc; the second is the ratio between the smallest angle formed by the trial arc and the pre-existing edges at either of its end points, and the constant 90°. The first factor measures how evenly the trial arc divides the original cycle, and is ideally close to unity. The second factor becomes very small if the trial arc tends to form an elongated 'needle-like' element. The trial arcs are ranked by an index found as the product of these two quality factors, and the trial arc with the highest index is selected to subdivide the cycle. The new arc is immediately subdivided if necessary, in accordance with the angle parameter discussed above.

If it is impossible to find a spanning connection within the cycle, then the heuristic test involving the smallest angles is relaxed – if a spanning arc can still not be constructed, then two arcs of the cycle are subdivided, and the triangulation procedure is reentered. The two arcs chosen for division will be the pair that approach each other most closely (without, of course, sharing a common vertex). As illustrated in Fig. 7, difficulties in finding a spanning arc usually arise from highly 'pinched' cycles which often occur in crevices and internal cavities. By dividing each arc in two at the point of closest approach, it becomes possible to build a spanning arc. If the distance of closest approach should be found at the end points of the arcs, then the two longest arcs in the cycle are divided at their centers.

The whole process is iterated until only 3-cycles remain in a given contact region. One additional circumstance needs to be considered: as illustrated in Fig. 8, the situation can arise that one contact cycle lies in the interior of another. If a given atom is associated with more than one contact cycle, the algorithm tests the cycles in pairs, and determines whether it is possible to connect the members of each pair by an edge that lies on the solvent-accessible surface. If so, an optimal spanning arc is generated (using the same approach as in the regular triangulation procedure), and the pair of cycles is converted into a single cycle, as shown in Fig. 8. Once all such conversions have been made, the triangulation procedure continues as described above.

Reentrant triangulation

Reentrant triangulation proceeds by essentially the same algorithm as the contact case, but the triangulation takes place on a probe rather than an atom. Concave reentrant regions have a very simple and regular geometry (typically described by a single spherical triangle), and great circles (rather than generalized circular arcs) are used to divide the cycles.

Saddle triangulation

Saddle triangulation is relatively straightforward, since the saddle sections, which are subsets of tori, are homeo-

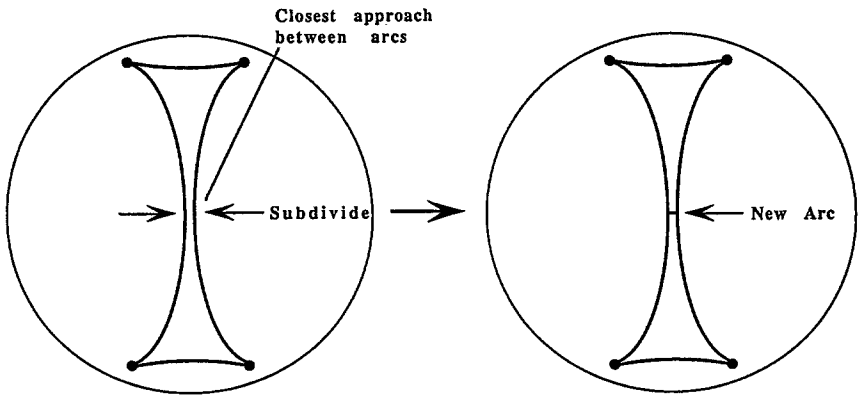


Fig. 7. Method for handling extremely narrow contact regions, often occurring in deep crevices. The pair of arcs indicated are each subdivided at their point of closest approach. A short spanning arc can then be added when the triangulation procedure is reentered.

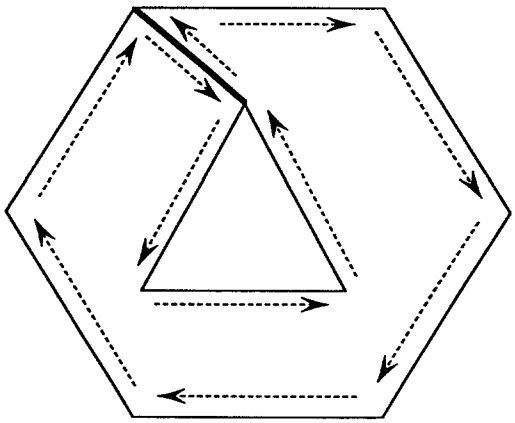


Fig. 8. A ring-shaped contact region (solvent-accessible surface is located between the two initial contact cycles, represented here as a hexagon and a triangle). The pair of cycles are converted into a single cycle by the addition of a single arc (shown as a heavy line). The final orientations of the edges in the condensed cycle are indicated by arrows. Note that the arc added to connect the original pair of cycles is used *twice*, in both the 'forward' and 'reverse' directions.

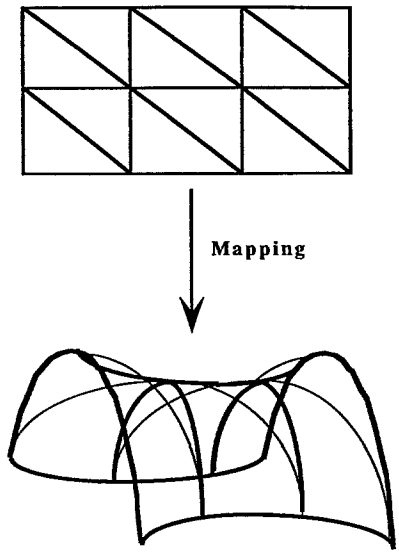


Fig. 9. Mapping from a triangulated rectangle in the plane to a triangulated saddle section in three-dimensional space.

morphic to a planar rectangle. Thus, triangulation can be carried out in the plane by a simple procedure, and the connecting edges then mapped to the toroidal surface of the saddle section, as shown in Fig. 9. The details of the mapping have been described previously [15].

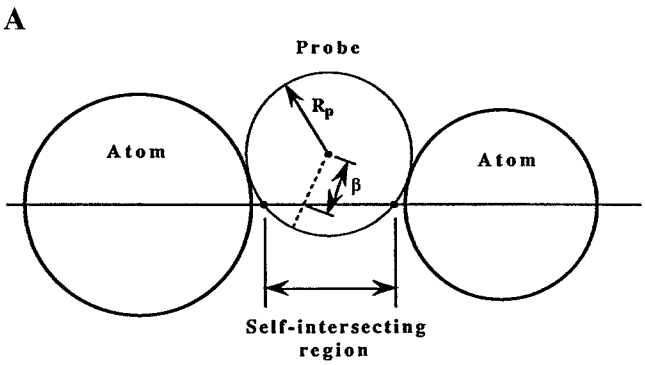
Self-intersecting surface

A difficulty in any approach that attempts to model a solvent-accessible surface is the handling of regions where the solvent probe can intersect itself. In the triangulation algorithm of Connolly [6], sections of self-intersecting surface are trimmed away, leaving a variety of cusp structures. In the SMART algorithm, occurrence of self-intersecting surface is avoided by the introduction of a novel approach in which the probe sphere can continuously deform. Figure 10A shows a probe bridging two atoms, which are spaced sufficiently far apart for the interatomic axis to pass through the probe. As the probe rotates about the axis (thus generating a saddle section), it will clearly intersect itself between the two points indicated. This can be avoided if the radius of the probe is adjusted so that it can never touch the interatomic axis. In our approach, the overall radius of the probe is not adjusted; rather, the radius becomes a *function of position* on the probe surface, and is always less than or equal to the 'full' probe radius.

To control the deformation of the probe, we first define the *probe overlap* by

$$O_v = \beta - R_p \qquad (1)$$

where β is the distance from the probe to the interatomic axis along some chosen direction, and R_p is the full probe radius. The overlap is negative along those lines from the



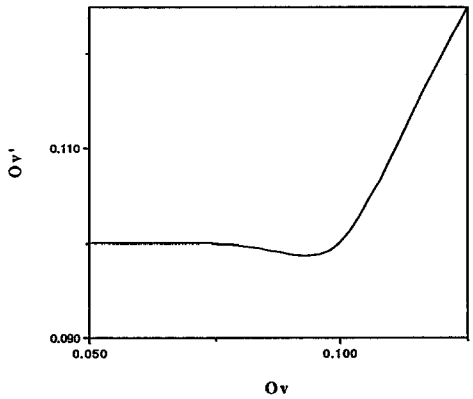


Fig. 10. (A) Creation of self-intersecting surface. A probe is shown rolling between two atoms that are sufficiently far apart for the probe to intersect the interatomic axis. At positions between the two points indicated, the surface generated by the moving probe will actually lie *inside* the probe. The distance β between the probe center and the interatomic axis is measured along a particular line (dashed) in the figure. (B) The modified overlap function.

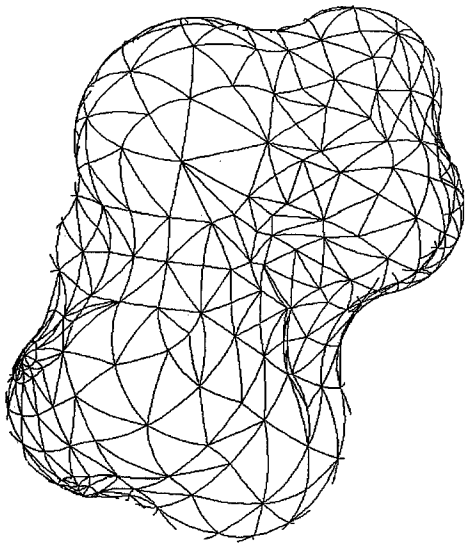


Fig. 11. Triangulation for dialanine. Atomic radii are taken from the CHARMm parameter set; the angle parameter is 30°.

probe center where self-intersecting surface will be found. The strategy is then to make R_p a function of direction, so that the overlap can never be negative. We next introduce a modified overlap function O'_v, given by

$$O'_v = (1 - f(O_v))\delta_1 + O_v \cdot f(O_v) \tag{2}$$

where

$$f(x) = \begin{cases} \dfrac{e^{-1/x}}{e^{-1/x} + e^{-1/(\delta_2 - x)}} & x > 0 \\ 0 & x \leq 0 \end{cases} \tag{3}$$

The function O'_v is shown in Fig. 10B. This expression, which appears complicated, simply connects the two functions $O'_v = \delta_1$ and $O'_v = O_v$ with a smooth transition. The transition is completed within the interval $[\delta_1, \delta_2]$ of the domain of the function. δ_1 thus serves as a limiting value for the overlap, and δ_2 controls the start of the transition between the linear and constant parts of O'_v. O'_v is in the class of 'bump' functions, which are used in differential topology to make smooth transitions between maps defined on intersecting domains [27]. It has continuous derivatives of all orders. In this work, we take $\delta_1 = 0.1$ and $\delta_2 = 0.2$.

Given the modified overlap, a modified probe radius R'_p is then found by a simple rearrangement of Eq. 1:

$$R'_p = \beta - O'_v \tag{4}$$

For the case of concave reentrant surface, the distance β is measured from the probe center to the plane defined by the three contact atoms. (It should be noted that in this case the deformation function also prevents two different probes on opposite sides of a triad of atoms from intersecting each other.) By definition, the modified probe radius is continuous when passing from concave reentrant regions to adjacent saddle sections; moreover, for reasonable choices of atomic radii, $R_p = R'_p$ at probe–atom contact points, so that the probe cannot 'fall off' the molecule. At most positions on the molecular surface, the probe radius is unmodified, and the surface conforms to the Lee and Richards [1] definition. Typically, it is only in deep crevices and pockets on the exterior surface, and in internal cavities, that the probe radius is deformed to avoid self-intersection.

The effect of the triangulation procedure is to wrap a smooth manifold (which may consist of several connected components) about a collection of spheres representing the atoms that comprise the molecule.

Algorithm output

The final product of the algorithm is a file (in binary format) which contains: (i) probe and atomic radii; (ii) a list of edges (arcs) together with their geometric specification. Two varieties of arcs are produced, those that lie on, or can be projected on, spherical surfaces (i.e., the arcs found on probe or atom surfaces), and edges in the interior of saddle sections. The arcs are numbered, and the specification of each arc includes the vertices at its end points; and (iii) a list of triples of arcs, which define the

TABLE 1
CPU TIMES FOR TRIANGULATION AT THREE DIFFERENT ELEMENT DENSITIES[a]

Molecule	Angle parameter (°)	Probe placement	Saddle location	Triangulation	Total
Calixarene	90	25.0	0.6	16.0	41.6
	60	25.0	0.6	24.6	50.2
	30	25.0	0.6	107.0	132.6
Calmodulin	90	249.6	3.5	111.8	364.9
	60	249.6	3.5	195.4	453.8
	30	249.6	3.5	1177.4	1430.5
Acetylcholinesterase	90	1222.2	15.5	395.5	1633.2
	60	1222.2	15.5	637.7	1875.5
	30	1222.2	15.5	3657.9	4895.6

[a] All times are given in seconds for a Silicon Graphics 340 workstation.

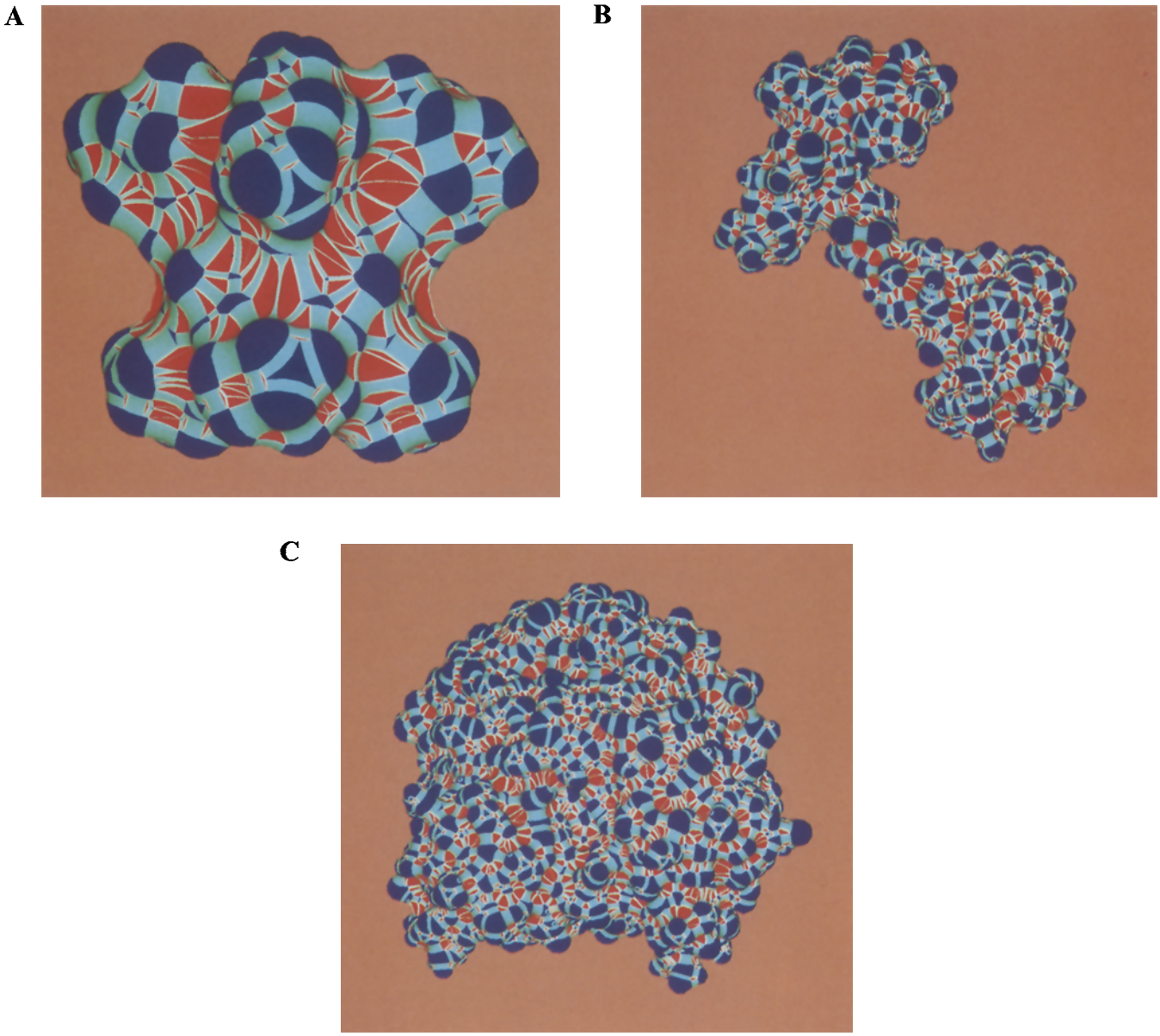


Fig. 12. Shaded surfaces for (A) calixarene; (B) calmodulin; and (C) acetylcholinesterase. Atomic radii were taken from the CHARMm parameter set; the angle parameter was set to 90°. Contact regions are shown in dark blue, concave reentrant in red, and saddle regions in light blue.

3-cycles (elements) that cover the solvent-accessible surface.

Applications

Figure 11 shows a triangulation of dialanine, illustrating the typical appearance of the surface elements generated by SMART. Shaded surfaces for calixarene (A. Varnek, private communication), calmodulin [28] and acetylcholinesterase [29] are shown in Fig. 12. The three molecules contain 180, 1412 and 5093 atoms, respectively, which were assigned van der Waals radii from the CHARMm parameter set [30]. (For the proteins, positions of polar hydrogens were assigned using CHARMm.) A probe radius of 1.4 Å and an angle parameter of 90° were assumed throughout. A shaded representation is created by approximating each curvilinear element with 10 flat triangles, and color-coding these in accordance with the surface type (contact, concave reentrant or saddle) of the element. The graphic renderings are generated using an auxiliary computer program, which allows interactive rotation of the displayed structures.

The CPU times required for generating the three surfaces shown in Fig. 12 are compared in Table 1. Also shown are timings that correspond to angle parameters of 60° and 30°, and thus to higher densities of surface elements. The times are broken down into the fractions needed for probe placement, saddle location, and triangu-

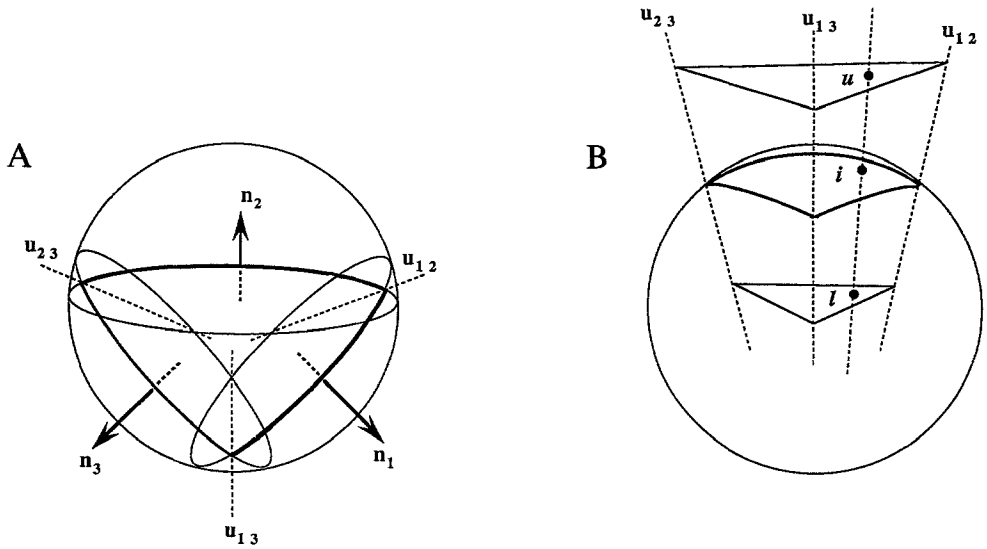


Fig. 13. Construction used to locate points on surface elements. (A) An element is a cycle of three arcs, which appear as heavy curved lines. Each arc defines a plane that cuts through the atom (or probe) that it lies on. The resulting cross sections appear as ellipses in the perspective of the figure, with normal vectors labeled $\mathbf{n}_1$, $\mathbf{n}_2$ and $\mathbf{n}_3$. Two neighboring cross sections intersect along a line; the cross sections shown in the figure form pairwise intersections along three lines, u_{12}, u_{13} and u_{23}. (B) Upper- and lower-bounding triangles are constructed parallel to the plane defined by the three nodes (corners) of the element. A point on the standard unit triangle (not shown) is mapped to a point l on the lower triangle and a point u on the upper triangle; these points in turn locate a unique point i on the surface of the element. It is easy to show that the mapping is one-to-one and onto. In this way, points on each surface element are assigned local coordinates.

lation (the first two contributions being independent of the angle parameter). As could be expected, these times are much longer than required for generation of a dot surface, since not only do probe positions have to be located, but a large amount of computation is needed to carry out the triangulation.

It is seen that surface generation can be completed in a reasonable time for a small protein (6 min for calmodulin), and is not prohibitive even for a large structure (27 min for acetylcholinesterase), provided that an angle parameter of 90° is adopted. CPU times are only modestly increased when the angle is adjusted to 60°. When the parameter is reduced to 30°, however, the density of surface elements (discussed below) grows dramatically, and the required computing time is increased by about a factor of four compared to the 90° case. In practice, it is far more efficient to increase element density by subdividing a coarse triangulation, a technique that is available with the author's previous MOLSURF [15] programs. Work is now underway to port the element subdivision approach to SMART, so that the use of a relatively large angle parameter should be the norm in future applications of the algorithm.

It is a straightforward matter to integrate over the surface elements generated by SMART. Figure 13 illustrates the technique used to establish a mapping from a standard unit triangle in the plane to a contact (or concave reentrant) element in three-dimensional space. Each contact element is bounded by three planes, determined by the circular arcs that define the element. In addition, we can construct a plane through the three vertices of the element, and make two bounding planes parallel to this

TABLE 2
ELEMENT DENSITIES, SURFACE AREAS AND MOLECULAR VOLUMES

Molecule	Angle parameter (°)	#Vertices	#Arcs	#Elements	Vertex density (Å^{-2})	Surface area (Å^2)	Δ%[a]	Molecular volume (Å^3)	Δ%[a]
Calixarene	90	1 294	3 876	2 584	1.1	1 141.6	–	1 877.3	–
	60	1 601	4 797	3 198	1.4	1 136.9	0.4	1 865.1	0.7
	30	3 797	11 382	7 588	3.3	1 136.2	0.06	1 861.9	0.2
Calmodulin	90	5 913	17 690	11 784	0.7	8 354.9	–	20 092.8	–
	60	8 195	24 470	16 304	1.0	8 330.5	0.3	20 004.4	0.4
	30	22 894	68 639	45 750	2.7	8 331.0	0.006	19 990.1	0.07
Acetylcholin-	90	15 145	45 281	30 168	0.8	18 160.7	–	74 052.4	–
esterase	60	20 215	60 366	40 224	1.1	18 109.2	0.3	73 701.4	0.5
	30	53 657	160 812	107 186	3.0	18 111.7	0.01	73 603.6	0.1

[a] Percent difference between each value and its predecessor in the column.

one, as shown in Fig. 13. The result is an irregular pentahedron that encloses the surface element. As shown in the figure, it is then possible to establish a geometric construction that uniquely maps points from the top and bottom bounding plane to the surface of the curvilinear element (and, by composition of maps, from the standard triangle to the element). This mapping is both one-to-one and onto, and by computing the Jacobian of the map it is then possible to perform integration over the surface element. In this work, integration is carried out using numerical quadrature with symmetric integration points [31]. (The method for integration over saddle elements has been described previously [15].) In the case of reentrant regions, it should be noted that the mapping technique is adjusted to take into account probe deformation when it arises.

Given a method for integrating over the surface, it is a straightforward extension to compute molecular volumes. By the divergence theorem,

$$V = \int_V (1)\,dV = \int_V \left[\nabla^2\left(\frac{1}{6}r^2\right)\right] dV = \int_S \left[\nabla\left(\frac{1}{6}r^2\right)\cdot\mathbf{n}\right] dA \quad (5)$$

or

$$V = \int_S \left[\frac{1}{3}r\cdot\mathbf{n}\right] dA \quad (6)$$

where r represents the position on the molecular surface S, with corresponding unit normal **n**. The integration over the volume V is thus transformed to a surface integral, and can be evaluated by the techniques described above.

Table 2 shows the results of area and volume computations for the three molecules displayed in Fig. 12, again using three values for the angle parameter (90°, 60° and 30°). The calculations are performed by an auxiliary program that reads the surface file produced by SMART. It is seen that areas converge to within about 0.01% and volumes to within about 0.1%. This is a much higher accuracy than is attainable with dot surface methods, and is one advantage of a description that specifies coordinates for all points on the molecular surface. It is interesting to note that the moderate increase in element density that results from adjusting the angle parameter from 90° to 60° leads to areas and volumes that are essentially identical to those computed with a parameter of 30°. The latter correspond to the highest densities used in this work, and this smallest value of the parameter would seem to represent computational 'overkill' if our objective is to compute accurate areas and volumes*.

Finally, it should be pointed out that the program, which computes molecular areas and volumes, also generates a separate list of *atomic* areas, which are computed by summing the areas of the contact elements of each atom, and by equally dividing the areas of reentrant elements among the atoms they are associated with. The atomic areas (which have not been presented here in the interest of brevity) show convergence similar to that of the total molecular surface area. The computed areas can be used with any of the available techniques that estimate contributions to the free energy of hydration, by using calculated accessible areas in conjunction with atomic solvation parameters [32–34].

Discussion

It should be stressed that the SMART algorithm was designed primarily for use with boundary element methods for computing electrostatic and hydration effects in biomolecules. In this light, the speed of the algorithm is not a primary concern, since the intended application imposes rather severe requirements on the surface to be generated – in particular, that it be smooth with a continuous normal, possess a reasonably uniform triangulation, and nowhere intersect itself. In contrast, the program recently described by Varshney et al. [35] for the computation of a Richards–Connolly-type solvent-accessible surface is very fast, but is less suitable for boundary element applications; it uses Connolly's triangulation procedure [6], the deficiencies of which for boundary element work have been previously noted [15], and, more importantly, lacks a general procedure for handling self-intersecting surface. However, this algorithm would appear to represent a significant advance for applications involving molecular visualization, or the rapid computation of surface areas.

The SMART algorithm is well adapted to its intended domain of application, as it generates a surface with a continuous normal, and a triangulation which is relatively uniform and free of high densities of elements in localized regions. The algorithm has been shown to work reliably with molecules of a wide range of sizes, and has special features to handle highly distorted contact regions that sometimes caused the older MOLSURF program to fail. Work is presently underway to adapt the new surface model to electrostatic methods already in use [15].

The surfaces generated by SMART can be immediately used for computations of molecular surface areas and volumes, with an accuracy much greater than that possible using dot surfaces. The surfaces can also be displayed

*One of the referees has suggested the use of spherical trigonometry to rapidly compute the areas of contact elements. While this would certainly be a viable approach, it would be applicable only to a subset of the elements generated by SMART; more importantly, the development of SMART is motivated by boundary element applications, where an integrand other than unity must be evaluated over each element, and where a numerical integration procedure is still necessary.

on a graphics workstation (although if one is only interested in visualization, other methods are undoubtedly simpler and more economical).

Conclusions

A method has been developed for defining the solvent-accessible surface as a smooth manifold, and partitioning the surface into curvilinear elements. Self-intersection of the surface is avoided by a novel technique that employs a continuously deformable probe. The resulting surface has a continuous normal, and is well suited for use with boundary element methods for computing electrostatic and hydration effects, or with techniques that estimate free energies of solvation on the basis of accessible surface area. Methods are present for performing numerical integration on the surface, and it is seen that surface areas and volumes can be calculated to very high resolution. The surfaces are also suitable for display either as meshes (triangulations), or as shaded renderings.

Acknowledgements

My thanks to Dr. Alex Varnek (University of Strasbourg) for providing the coordinates for calixarene, and for many stimulating communications. Also thanks to Dr. Steve Armentrout (Penn State Department of Mathematics) for many helpful discussions.

References

1 Lee, B. and Richards, F.M., J. Mol. Biol., 55 (1971) 379.
2 Richards, F.M., Annu. Rev. Biophys. Bioeng., 6 (1977) 151.
3 Wodak, S.J. and Janin, J., Proc. Natl. Acad. Sci. USA, 77 (1980) 1736.
4 Miertus, S., Scrocco, E. and Tomasi, J., Chem. Phys., 55 (1981) 117.
5 Connolly, M.L., J. Appl. Crystallogr., 16 (1983) 548.
6 Connolly, M.L., J. Appl. Crystallogr., 18 (1985) 499.
7 Zauhar, R.J. and Morgan, R.S., J. Mol. Biol., 186 (1985) 815.
8 Gibson, K.D. and Scheraga, H.A., Mol. Phys., 62 (1987) 1247.
9 Zauhar, R.J. and Morgan, R.S., J. Comput. Chem., 9 (1988) 171.
10 Meyer, A.Y., J. Comput. Chem., 9 (1988) 18.
11 Hasel, W., Hendrickson, T.F. and Still, W.C., Tetrahedron Comput. Methodol., 1 (1988) 103.
12 Higo, J. and Go, N., J. Comput. Chem., 10 (1989) 376.
13 Karfunkel, H.R. and Eyraud, V., J. Comput. Chem., 10 (1989) 628.
14 Pascual-Ahuir, J.L. and Silla, E., J. Comput. Chem., 11 (1990) 1047.
15 Zauhar, R.J. and Morgan, R.S., J. Comput. Chem., 11 (1990) 603.
16 Klein, T.E., Huang, C.C., Pettersen, E.F., Couch, G.S., Ferrin, T.E. and Langridge, R., J. Mol. Graph., 8 (1990) 16.
17 Heiden, W., Schlenkrich, M. and Brickmann, J., J. Comput.-Aided Mol. Design, 4 (1990) 255.
18 Perrot, G. and Maigret, B., J. Mol. Graph., 8 (1990) 141.
19 Silla, E., Tunon, I. and Pascual-Ahuir, J.L., J. Comput. Chem., 12 (1991) 1077.
20 Kundrot, C.E., Ponder, J.W. and Richards, F.M., J. Comput. Chem., 12 (1991) 402.
21 Dodd, L.R. and Theodorou, D., Mol. Phys., 72 (1991) 1313.
22 Wang, H. and Levinthal, C., J. Comput. Chem., 12 (1991) 868.
23 Heiden, W., Goetze, T. and Brickmann, J., J. Comput. Chem., 14 (1993) 246.
24 Yoon, B.J. and Lenhoff, A.M., J. Comput. Chem., 11 (1990) 1080.
25 Juffer, A.H., Botta, E.F.F., Van Keulen, B.A.M., Van der Ploeg, A. and Berendsen, J.C., J. Comput. Phys., 97 (1991) 144.
26 Connolly, M.L., QCPE Program No. 429, Quantum Chemistry Program Exchange, University of Indiana, Bloomington, IN.
27 Brocker, Th. and Janich, K., Introduction to Differential Topology, 1st ed., Cambridge University Press, Cambridge, 1982, p. 63.
28 Protein Data Bank entry 4CLN: Taylor, D.A., Sack, J.S., Maune, J.F., Beckingham, K. and Quiocho, F.A., J. Biol. Chem., 266 (1991) 21375.
29 Protein Data Bank entry 1ACE: Sussman, J.L., Harel, M. and Silman, I., Science, 253 (1991) 872.
30 Brooks, B.R., Bruccoleri, R.E., Olafson, B.D., States, D.J., Swaminathan, S. and Karplus, M., J. Comput. Chem., 4 (1983) 187.
31 Akin, J.E., Application and Implementation of Finite Element Methods, Academic Press, London, 1982.
32 Eisenberg, D. and McLachlan, A.D., Nature, 319 (1986) 199.
33 Ooi, T., Oobatake, M., Nemethy, G. and Scheraga, H., Proc. Natl. Acad. Sci. USA, 84 (1987) 3086.
34 Hermann, R.B., J. Comput. Chem., 14 (1993) 741.
35 Varshney, A., Brooks Jr., F.P. and Wright, W.V., IEEE Comput. Graph. Appl., 14 (1994) 19.