A Simple Method for the Representation, Quantification, and Comparison of the Volumes and Shapes of Chemical Compounds

TERRY R. STOUCH and PETER C. JURS*

The Pennsylvania State University, University Park, Pennsylvania 16802

A conceptually and computationally simple method for the definition, display, quantification, and comparison of the shapes of three-dimensional mathematical molecular models is presented. Molecular or solvent-accessible volume and surface area can also be calculated. Algorithms, programming considerations, accuracy, and time and storage requirements are discussed. The method requires no extensive programming skills and could be implemented on a desk-top computer.

INTRODUCTION

Display and investigation of molecular shape has progressed from paper and pencil sketches to physical models to computational systems employing computer graphics. Mathematical computer models can be pictorially represented many ways: as stick figures, ORTEP plots, space-filling representations, and many forms of surface drawings.[1] Computer models can be overlapped with one another computationally in order to compare two structures.[2,3] Quantitative evaluation of shape and volume differences between two compounds is difficult with stick figures; space-filling representations are more informative.

The most rigorous method for quantifying the volume differences between the space-filling forms of molecular models is to integrate over the common or the unique volumes of the molecules. While integration over single atomic spheres or over two overlapping spheres is simple, integration over multiple intersecting spheres becomes a difficult task.[4] Integration as a means of comparing structures is a reasonably complex programming task, would make the display of common or excluded overlap difficult, and would be computationally complex for comparing many compounds. A simpler approach is to represent molecular volume with evenly spaced points, much as surfaces are represented by Connolly dot surfaces.[5] If properly implemented, this method allows for rapid comparison of molecular shapes and for quantification of the results. The volume and surface area of entire molecules, or portions of molecules, can be calculated. Molecular shapes or overlap or excluded volume between any number of molecules can be displayed and quantified with simple graphics devices. This method is conceptually and computationally simple and can be implemented on desk-top computers. We will present the method and algorithms for its implementation. Uses, accuracy, time expenditure, and storage requirements will be discussed.

METHODOLOGY

A molecule, represented by the three-dimensional coordinates of its atoms and their van der Waals radii, is placed in a three-dimensional grid of arbitrary density. The atomic coordinates can be determined from crystallographic data or from molecular model builders. Each point of interesection of the grid is checked to see if it lies within the molecule. If so, the point is put into a state ("on") different from that of those points that are not within the molecule ("off"). As the density of the grid increases, more points must be examined. If two molecules are superimposed previous to the bit encoding of the structure and if the grids are similarly defined, the two structures can be compared point by point. In fact, any number of superimposed structures can be compared in this way.

In order to visualize the shape of a molecule, the on points can be plotted on any device that can plot an array of single dots or other small characters at a resolution equal to or greater than that of the grid. Since the three-dimensional shapes of the molecules are coded, they can be viewed from any direction. The implementation described below makes viewing along the Cartesian coordinates convenient. Comparisons of structures can also be viewed, as is shown under Uses and Discussion.

Algorithm. The three-dimensional grid can be a considered as a rectangular box defined by high and low x, y, z Cartesian coordinate values and a constant spacing between the grid intersections (the points). The size of the box can be varied to include a molecule of any size or only a portion of a molecule. The size of the increment is variable; a smaller spacing will result in a denser grid and a more precise description of the molecule.

We orient the axes of the grid along the x, y, z axes for computational convenience. The points lie on lines parallel to the x axis, which we call rows. Each x–y plane contains a series of rows, which we call a slab. The grid consists of a series of these slabs. For example, if a small molecule is put in a grid with low x, y, z coordinate values (in Å) of (0, 0, 0), high coordinate values of (10, 10, 10), and a density of one point per linear angstrom, then each row consists of 11 points, each slab will have 11 rows, and the entire grid will consist of 11 slabs and a total of 1331 points (Figure 1). Care must be taken in assigning the coordinates that define the box. If not chosen distant enough from the molecule to include the full van der Waals radii of the atoms, some might be truncated. The grid need not encompass the entire molecule, however. At times, only portions of a molecule may be of interest. Examination of only those portions of interest will reduce both computational time and the complexity of the problem.

The algorithm that we used for coding a molecule is straightforward. It requires calculation of the coordinates of the points in the grid, calculation of the distance from that point to the points representing the atoms of the molecule, and comparison of that distance to the van der Waals radius of the atom.

The coordinates of the points in the grid are calculated starting with the point with the lowest x, y, z coordinates values by incrementing along each axis by the grid increment until the user-defined cutoffs are reached. We increment the x, y, and z coordinates respectively so that the rows of the first slab are calculated first, followed by those in the second slab, etc.

The distance between each point and the atoms in the molecule is calculated from the Euclidean distance metric (eq 1). If a point–atom distance is less than or equal to the van der Waals radius of the corresponding atom, then that point is within the atom and the status of that point is on. If not, then that point is off. A solvent-accessible shell could be computed by adding the solvent radius to the van der Waals radius of each atom. The molecular volume and surface area calculated from the point encoded description would then be the solvent-accessible volume and surface area.

$$(\text{distance})^2 = (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2 \quad (1)$$

0095-2338/86/1626-0004$01.50/0

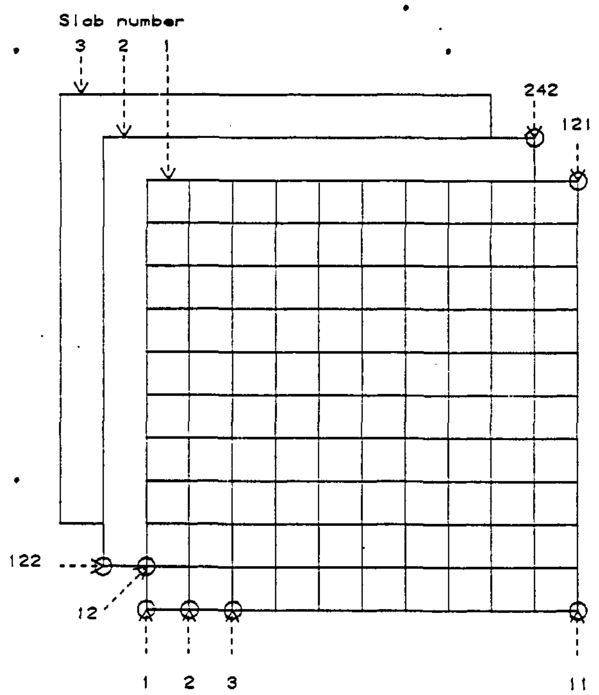

Figure 1. Schematic of the three-dimensional grid. Only three slabs are shown for clarity. Circles refer to point/bit numbers mentioned in the text.

As each point's status is established, it can be immediately summed for volume and surface area calculations. The on–off point information can also be stored for plotting, for comparisons with other, previously superimposed molecules, or for future use. The establishment of the status of the points in the grid is, by far, the most time-consuming step in this technique; comparison of molecules and calculation of molecular properties is rapid. It makes sense to store the information once it has been obtained. As we will show, the point-encoded shapes can be stored in such a way that many large (100 atoms), densely coded molecules can be stored on a standard floppy disk. Saving this information not only provides greater versatility but also reduces the complexity and size of the program and its arrays. Logical operations between preestablished arrays is far quicker than logical operations during the establishment of the status of the points. Also, if comparison are to be made between one molecule and several others, this will result in a substantial time savings over pairwise comparisons between the molecules during the establishment of the status of the points.

A bit string is a very convenient means of storing this information. Each point in the grid is represented by a bit in this string, and each on point corresponds to an on bit. Storage of the status of each point requires only one bit. Large, densely encoded molecules could then be stored without requiring massive amounts of storage space. The density and size of the grid will determine the amount of storage required for each compound. A fairly dense encoding of hydrocortisone ($C_{21}O_5H_{30}$) requires approximately 13 000 bytes of storage. Given the almost 400 000 bytes of storage on a conventional $5^1/_4$-in. floppy disk, over 20 such structures could be stored. Storage requirements will be noted further under Uses and Discussion and in Table II.

The use of bit strings has the added advantage of making comparisons of molecules direct and quick. Just as the on and off points can be compared between structures, two bit strings representing two structures can be compared on a bit by bit basis. Bitwise logical functions make this a rapid task. If these functions are not available, simple arithmetic can be used. Bit handling will be discussed under Bit Manipulation.

The molecule can now be viewed as a long string of bits with each point in the three-dimensional grid mapped onto a bit in the string. The actual method of mapping is unimportant, but it must be the same for two molecules if they are to be compared. It is also important for plotting and for volume calculations on portions of molecules. One method of mapping points onto the bit string is to position the rows of the grid in a linear string: first the rows of the first slab, lowest row to highest, then the rows of the second slab, etc. If the previously described scheme for calculating the coordinates is used, this would amount to assigning bit numbers sequentially as the points were examined. The first bit in the string would correspond to the point with the lowest x, y, z corrdinate value. The next bit would represent the second point in the first row (parallel to the x axis). Since there are 11 bits in each row, the first bit of the second row of that first slab would be represented by the 12th bit. The first bit in the first row of the second slab would be the $(11 \times 11) + 1 = 122$nd bit in the string (see Figure 1).

Once the atom coordinates, elemental types, and van der Waals radii are determined and once the starting and stopping coordinate values and grid density are set, bit encoding of the molecule takes only a few lines of code. It can be implemented by (1) three nested loops that increment the x, y, and z axes, (2) a line to increment the bit number (assuming the linear string of bits which we have used, (3) a loop to take each atom in turn, (4) a line to calculate the distance of a point from an atom, (5) a line to compare this distance with the van der Waals radius of that atom, and finally (6) a line to properly set the value of the bit in the bit string. This latter line in our program calls a bit-manipulation subroutine that will be described later. The core of the bit encoding can consist of less than 10 lines. Other lines might be added if volume or surface area calculations are desired at this stage. These computations can be reserved until the entire molecule is coded, however.

For convenience, we stored the bit string in an array of 16-bit integer words. During bit manipulation it was treated as one continuous string by the use of simple arithmetic conversions between the position of the bit in the imaginary bit string and the position within the integer array. For example, the 100th bit in the string would exist in the $100/16 + 1 = 7$th word and the $100 - (16 \times 6) = 4$th bit of that string. For the example grid above, this would correspond to the first bit in the 10th row of the first slab. Since the first nine rows were occupied, the remaining bit must be in the 10th row. Since the starting x, y, z coordinate of the grid was (0, 0, 0), this bit would correspond to a point with x and z coordinates of 0 and a y coordinate of $0 + (9 \times 1) = 9$ (the increment had a value of 1 Å). With simple arithmetic it is easy to convert between bit string, integer array, and three-dimensional coordinates. Other storage schemes could also be used. There are advantages and disadvantages in terms of input and output operations and string manipulations. Certainly, there is no need to keep the entire bit string within the program; on a small machine, it may be best to use a small string and periodic transfers of information to some storage medium.

The time required to encode a molecule is a function of the grid size, its density, the molecule's size and shape, and the fraction of the grid that the molecule occupies. The most time-consuming step of this approach is the establishment of the points' status. Once the coordinates of a grid point are calculated, the distance between it and many of the atoms in the molecule may have to be computed and evaluated before an atom is found that encompasses the point. If the point is outside the molecule, all the atoms will be checked. A substantial amount of time can be saved by initially squaring all the atomic radii and calculating only the square of the distance between the point and each atom. The same distance comparison can then be performed with one less square root operation. Time can also be saved if the grid is fit tightly about each molecule. This reduces the number of the time-consuming off points that are searched. If molecules are to be compared, however, the program must be able to orient the grids properly.

Much time can be saved if the search through the atoms is conducted intelligently rather than as a blind search through all of the atoms in the molecule. For example, all the points in a given slab have the same *z* coordinate. There is no need to calculate and compare point–atom distances for atoms whose distance from the current slab along the *z* axis is greater than their van der Waals radii. Prior to the investigation of any slab, the *z* coordinates of all the atoms can be compared to that of the slab. Any atoms far from the slab can be excluded from consideration during the evaluation of the points in that slab. The calculations for these comparisons are fewer and simpler than those for the atom–point comparisons that they replace. Also, they need be done only once per slab, rather than for all the points in the slab. This drastically reduces the average number of point–atom distances that must be calculated and evaluated; it reduces computation time for a molecule 30–60%. It is done most efficiently by sorting the atoms according to ascending *z* coordinates. As the algorithm progresses from low to high *z* values, the excluded atoms can be identified by a similar progression through the sorted list. The *x* and *y* coordinates can be handled in the same way with an additional, if lesser, time saving. Since the points in each row are calculated by incrementally increasing the *x* coordinate, there is no need to consider those atoms whose *x* coordinates disqualified them from consideration for a previous point with a lower *x* value. Hence, a sorted list of *x* values within a slab can also be used to decrease the number of atoms searched with a resulting decrease in computation time. Such preprocessing of the *x* and *z* coordinate values decreased the times reported under Uses and Discussion by between 10 and 70%.

By use of logical functions, comparison of two molecules is performed by comparing the strings on a bit-by-bit basis. An AND operation between bit strings representing two structures results in a third string that contains the volume that the two molecules share. The new string can be back-transformed into the three-space through knowledge of the position and density of the grid. That volume which is common to both molecules can then be visualized by use of some simple graphics device, or this new string can be manipulated in some way. For example, the volume could be quantified and used in molecular shape analysis, such as performed by Hopfinger.[6] Other logical functions would yield other comparisons. The OR operation on the two molecule strings would provide a composite of the structures. An exclusive-or operation would yield a third strong that would contain only unique aspects of each molecule's structure. This could than be ANDed with either of the original strings to provide that volume which is unique to either one of the structures.

This work was performed on the Chemistry department's PRIME 750 computer with 2 Mbytes of main memory. Tektronix PLOT10 software, a Retro-Graphics-enhanced Lear-Siegler ADM3A+ terminal, and a Tektronix 4662 plotter were used for graphics. The ADAPT system[7] was used for structure entry and modeling, as a source of atom preception and bit-handling subroutines, for some data analysis, for producing graphs and space-filling figures, and to calculate volumes by Pearlman's method.[4] MINITAB[8] was used for the regression analysis. The van der Waals radii used to calculate the point-encoded representation are shown in Table I.

Table I. van der Waal Radii

atom type	radius (Å)	atom type	radius (Å)
C (sp^3)	1.70	N (sp)	1.60
C (sp^2)	1.70	N (aromatic)	1.60
C (sp)	1.77	S (singly bonded)	1.80
C (allenyl)	1.70	S (doubly bonded)	1.80
C (aromatic)	1.77	F	1.50
O (singly bonded)	1.52	Cl	1.75
O (doubly bonded)	1.50	Br	1.85
N (sp^3)	1.55	I	1.97
N (sp^2)	1.55		

Bit Manipulation. The easiest way to perform the bit manipulation is to use a system or language that allows access to individual bits. If this is not possible, bit manipulation can be performed with bitwise logical functions or simple arithmetic. Bitwise logical functions, AND, OR, and XOR, are FORTRAN F77 extensions and are also available in other languages, such as extended BASIC. With them, bit manipulations can be done in a few lines of code. The bit-handling routines that we used are not complex and consist of less than 30 lines of FORTRAN code.

An AND operation performed on two binary strings will provide a third binary string that has set to 1 only those bits that were set to 1 in both of the initial strings. A bitwise OR operation will provide a third string that has a 1 in every position that was set to 1 in either of the initial strings. The XOR operation sets to 1 those positions that are 1 in one string or the other but not both.

The OR operation can be used to turn on bits in a string. For example, to change the second bit of the string 00101001 to 1, the OR operation can be performed on that string and the string 00000010 resulting 00101011. This is equivalent to adding the necessary power of 2 of the first string, an alternative procedure if bitwise logical functions are not available.

A bit's status can be checked in a similar manner with the AND operation. For instance, to check the fourth bit of the binary string representing 41, it could be ANDed with string 00001000 resulting in string 00001000, indicating that the fourth bit was set to 1. If the operation were performed with 00000100 instead of the second operand, 00000000 would result, showing that the third bit of the 41 string was set to 0. This method can be used to check bits in a bit string when calculating volume and surface area and when plotting. In both of the above procedures, the second strings can be set up initially in a FORTRAN DATA statements or within the code by some arithmetic statement. If bitwise logical functions are not available, bit manipulation can be performed by simple arithmetic.

USES AND DISCUSSION

Increasing the density of the grid will improve the accuracy with which the molecules are encoded, but it will also increase computation time. Table II shows the cpu time required to compute the point representations of 22 compounds at several grid densities as well as the disk space required for bit string storage. For these computations the grid was fit tightly about the molecule. These times should only be used for comparison among themselves and with the times required for computation of volumes and surface areas by the other method as discussed under Volume Computations. They depend on the machine used, on the form of the implementation of the algorithm, and on the other factors mentioned under Methodology. The times become large for the higher densities. A compromise must be found between precision and time expenditure.

How many points are necessary to properly encode the shape of a molecule and calculate reasonable estimates of physical properties? Marsili et al.,[9] in describing a similar system,

Table II. cpu Seconds Required for Calculations of Point-Encoded Representations

		density of grid (points per linear angstrom)			
index	name	1	2	3	4
1	benzoic acid	0.51	3.20	10.50	27.08
2	endrin	2.52	17.63	57.12	148.57
3	mitomycin C	4.26	33.64	107.82	282.89
4	formaldehyde	0.06	0.40	1.22	2.92
5	nitrogen mustard	0.51	4.09	12.50	31.23
6	benzo[*a*]pyrene	1.70	12.62	42.81	96.56
7	benzo[*a*]anthracene	1.64	11.05	37.25	84.40
8	dichloromethane	0.11	0.85	2.67	6.07
9	naphthalene	0.94	6.00	19.63	44.01
10	methanol	0.15	0.78	2.52	5.69
11	cyclohexane	0.88	6.17	19.23	45.14
12	*trans*-decalin	2.11	13.32	43.09	96.07
13	*trans,anti,trans*-perhydrophenanthrene	3.08	23.56	72.20	172.95
14	hydrocortisone	7.40	50.72	170.24	442.29
15	ethane	0.18	1.15	3.57	8.42
16	ethene	0.12	0.68	1.98	4.78
17	hexane	0.74	5.31	17.29	40.62
18	nonane	1.88	13.12	40.99	100.00
19	butane	0.52	3.61	11.99	27.50
20	neopentane	0.86	5.44	17.48	39.70
21	*p*-toluic acid	0.91	6.47	19.92	47.12
22	*p*-xylene	0.87	5.92	18.45	42.87
		26–246[a]	160–1740[a]	494–5632[a]	1116–13064[a]

[a] Storage required (bytes).

indicated that millions of points may be necessary to properly encode a molecule. Coding molecules this densely would require a great deal of computation time and would make application of this method prohibitively time consuming on small machines. How sparse can a molecule's bit-encoded representation be and still be useful for viewing shape, comparing molecules, and calculating volume and surface area? We will attempt to answer this question throughout this section.

Visualization of Shape. In some cases, the need to visualize a molecule may determine the density. We may require that the molecule be encoded densely enough for us to recognize structural features. Figure 2 shows *p*-toluic acid coded at several densities along with its space-filling representation. In this particular figure, the van der Waals radii used for the smooth space-filling representation were chosen for esthetics rather than for accuracy and were different from those reported in Table I. Because of this, the outlines of the point-encoded molecules will not match the space-filling version exactly in this figure. At lower densities, the shape is crudely represented. Visual comparison to similar molecules would not be informative, but this density would serve to differentiate it from grossly different molecules. A reasonable outline is achieved at a density of 2 or 3 bits per linear angstrom (bla). In terms of the display, increasing the density past this serves only to fill in this outline.

There may be cases, for example in the comparison of molecules, where actual visualization of the shapes of the molecules or shapes of the results of some comparison operations may not be necessary. In these cases, the mere presence of bulk in one region vs. another may be the only information that is required. A sparse encoding of the shapes may be sufficient if the molecules are very different. If interesting regions of the shapes of the molecules are very similar, however, it may be necessary to code the molecules densely in order to observe those differences. For example, minor conformational differences in a ring system due to different substitutions on two molecules could be hard to see unless the density was high. In that instance, computational time could be conserved by encoding only those areas of interest by defining the grid to include only those areas.

Calculation of Volume. Each point represents $1/d$ Å, where d is the density of the encoding. Summation of those volume increments will yield on estimate of the molecular volume. How does the accuracy of the volume computations change with increasing density?

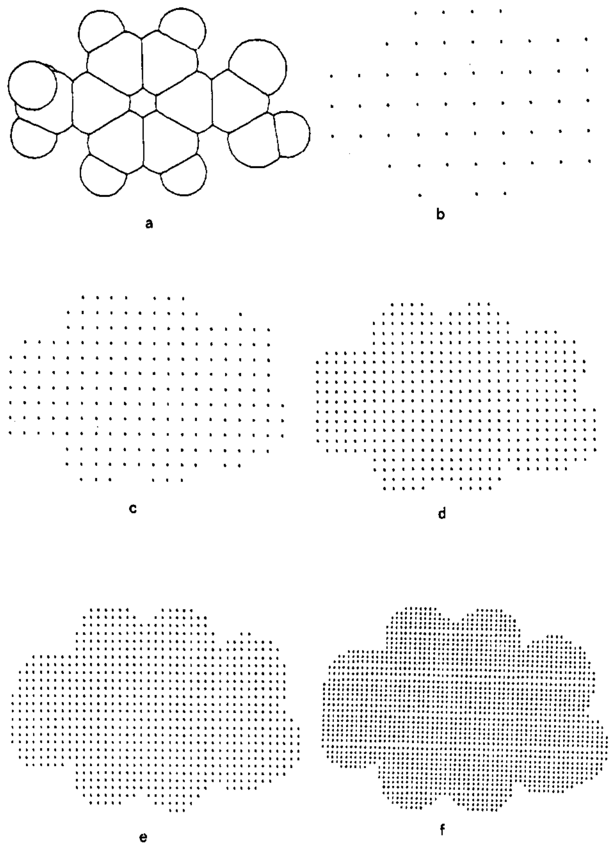

Figure 2. Space-filling and bit-encoded representations of *p*-toluic acid. Densities: (b) 1, (c) 2, (d) 3, (e) 4, and (f) 6 bla.

In order to answer these questions we have conducted two separate investigations. The first involves a series of simple hydrocarbons that were used to demonstrate the system of Marsili.[9] These authors encoded 45 of 47 compounds listed by Kier and Hall[10] and found a correlation coefficient between the sum of bits for each molecule and its heat of vaporization of 0.988. These hydrocarbons were fairly densely encoded;

Table III. Regression Summaries for Heats of Vaporization of 47 Simple Hydrocarbons

density	no. of points	R^2	SE
0.2	0–2	0.36	2.09
0.5	8–37	0.78	1.21
1.0	48–279	0.95	0.57
2.0	360–2268	0.97	0.45
4.0	2900–18 000	0.97	0.46
5.0	5655–35 191	0.97	0.46

Table IV. Regression Summaries for the Volume Calculations for 22 Compounds

density	no. of points	R^2	SE	relative error (%)
1.0	30–350	0.997	4.75	<8
2.0	250–2700	1.0	1.73	<3
3.0	800–9200	1.0	0.54	<1.4
4.0	2000–22 000	1.0	0.36	<0.43
5.0	3840–42 616	1.0	0.22	<0.33

ethane, the smallest molecule, was coded by over 3000 bits. This corresponds to a density between 4 and 5 bla. Would the correlation decreases for lower densities? We modeled all 47 of the molecules with the ADAPT model builder, MMUSER, and Allinger's MM1[11] model builder in tandem. These were then encoded at several different densities: 0.2, 0.5, 1.0, 2.0, 4.0, and 5.0 bla. The volumes were computed and regressed on the basis of vaporization (Table III). Reasonable correlations are obtained even at a density of 1. This density requires considerably less computational time than would the density used by Marsilli et al. The standard errors are plotted vs. density in Figure 3. The standard errors level off between 1 and 2 bla. In terms of correlation with heats of vaporization, little is gained by using a density greater than 1 bla. At this density, computational time is much lower than for a density of 4 or 5 bla.

Correlation with the heat of vaporization indicates that the volume calculations are providing physically meaningful numbers. However, even for such a homologous series of compounds, the heat of vaporization is not dependent only on volume. A better way to judge the volume calculations for accuracy, as well as computational time, is actual comparison to other computational estimates of volume. In the second study, the volumes calculated from the bit strings were compared with the volumes calculated by the method of Pearlman.[4] Pearlman's method is mathematically rigorous and has been found to be useful in many studies.[12] It requires, however, a large and complex program and is computationally intensive.

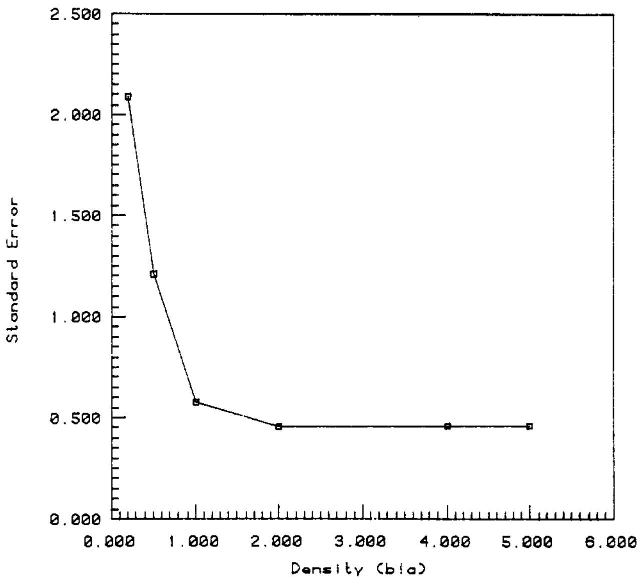

Figure 3. Standard error (Å^3) vs. density (bla). Regressions of calculated volumes onto heats of vaporization.

The structures used in the above study were all simple hydrocarbons. Correlations within a series of such similar compounds can be deceptive. In order to investigate the general utility of this method, the 22 structurally diverse compounds listed in Table II were investigated. These compounds were modeled with the ADAPT model builder, and volumes were calculated with Pearlman's method and from bit-encoded representations of density of 1, 2, 3, 4, and 5 bla. The results of regressions of the bit-encoded method of Pearlman are listed in Table IV. Standard error vs. density is plotted in Figure 4. A molecule by molecule comparison for two densities is presented in Table V for bit encodings of 1 and 4 bla. The cpu time consumed by Pearlman's method is also listed and can be compared to the corresponding times

Table V. Volumes Calculated for 22 Compounds

	Pearlman's method		point-encoded method			
			density = 1		density = 4	
index	vol[a]	cpu seconds	vol	RE[b]	vol	RE
1	114.49	4.76	113	1.30	114.41	0.07
2	244.86	22.23	245	0.05	245.92	0.43
3	275.41	27.70	269	2.32	275.23	0.06
4	30.71	0.62	30	2.30	30.81	0.34
5	118.14	7.34	117	0.96	118.19	0.03
5	118.14	7.34	117	0.96	118.19	0.03
6	244.07	12.08	236	3.30	245.13	0.43
7	227.14	11.21	219	3.58	227.61	0.20
8	56.73	1.28	61	7.53	56.77	0.07
9	134.69	6.68	137	1.71	134.63	0.04
10	36.52	1.59	36	1.41	36.47	0.13
11	101.37	9.20	101	0.36	101.41	0.03
12	157.24	17.45	167	6.21	156.97	0.17
13	213.19	25.45	215	0.84	213.38	0.08
14	340.78	42.63	348	2.11	340.56	0.06
15	45.29	2.34	43	5.04	45.38	0.19
16	39.84	1.32	43	7.93	39.78	0.14
17	112.74	8.19	107	5.08	112.64	0.08
18	163.27	12.64	163	0.16	163.38	0.06
19	79.01	5.24	80	1.25	79.22	0.26
20	95.37	8.45	90	5.62	95.39	0.02
21	130.84	6.33	127	2.93	130.44	0.30
22	120.96	6.68	116	4.10	120.86	0.08

[a] Volumes in Å^3. [b] Relative errors.

Table VI. Regression Summaries for the Surface Areas of the Compounds

				relative error		regression terms	
density	no. of points	R^2	SE	mean	SD	coeff	const
1.0	26–222	0.982	12.29	4.6	4.0	1.62	8.23
2.0	139–1097	0.996	6.10	2.2	1.7	0.346	2.27
3.0	333–2605	0.997	4.90	1.8	1.2	0.147	0.85
4.0	620–4776	0.998	3.93	1.5	0.9	0.081	0.26
5.0	992–7572	0.999	3.55	1.3	0.9	0.051	-0.41

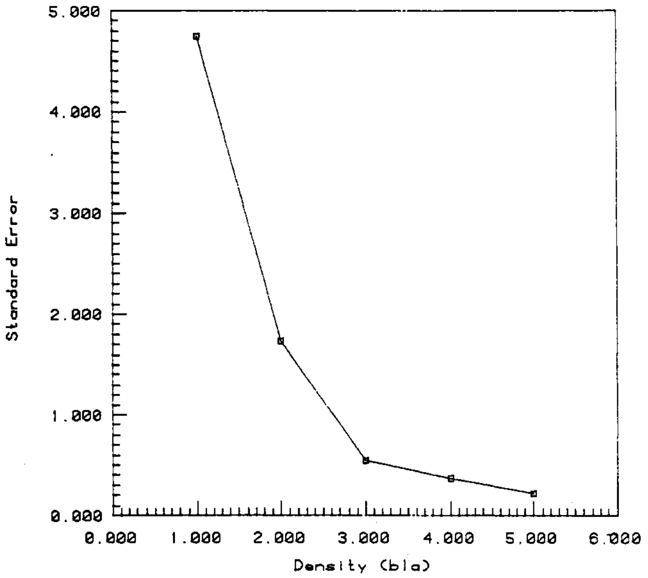

Figure 4. Standard error ($Å^3$) vs. density (bla). Regression of point-encoded volume calculations onto those of Pearlman's method.

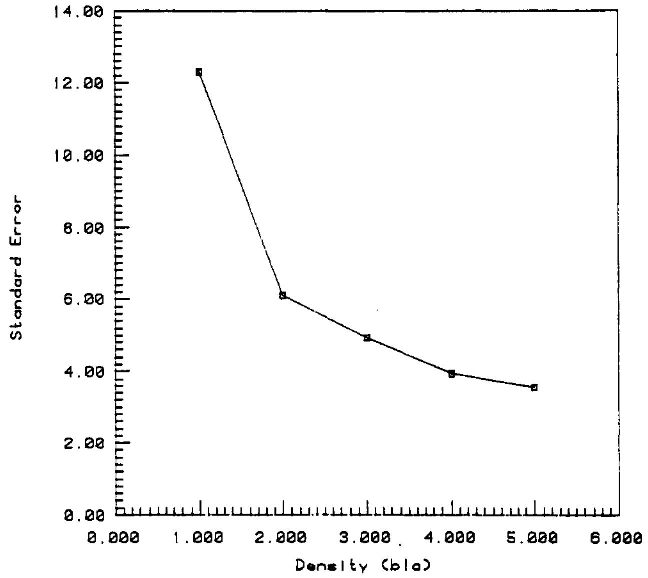

Figure 5. Standard error ($Å^2$) vs. density (bla). Regression of point-encoded surface area calculations onto those of Pearlman's method.

in Table II. Pearlman's algorithm requires the specification of a variable that influences the accuracy and time of the computation. Decreasing this lune increment will increase the accuracy of the calculations but will also increase computation time. We used a lune increment of 6°; smaller values caused only minute changes in the calculated volumes and surface areas. A larger value could have been used with a decrease in time and an unknown decrease in accuracy. As a general note, the time required by Pearlman's method does not vary as dramatically as that required by the point-encoded method for an equivalent increase in accuracy.

Estimates of volume well within 10% of the volumes calculated by Pearlman's method can be achieved with a density of 1 bla. The change in the standard errors become small past a density of 3 bla where the relative errors are all less than 1.5%. Tables IV and V and Figure 4 indicate that there is little extra accuracy to be gained with a density greater than 4 bla; the two algorithms probably converge well below a density of 10 bla. The time required to calculate the point encoding at a density of 1 or 2 bla is less than that required by Pearlman's method. At higher densities, however, the computational time increases greatly. The volumes calculated at a density of 3 bla required approximately twice as much time as those calculated with Pearlman's method. For very accurate volume calculations, the major advantage of point encoding over Pearlman's method is that the former is far simpler to program.

Calculation of Surface Area. It is not a difficult task to detect those on bits that are not surrounded by other on bits, and hence are on the surface of the molecule. The sum of these bits should correlate with surface area. Direct calculation of actual surface area is not as straightforward as direct calculation to volume. For the latter, each bit could be assumed to represent some cubical volume element. The surface points, however, would not necessarily represent identical surface areas. Surface points for atoms of different sizes would represent different areas. Also, as can be seen in Figure 2, this method does not actually place the surface points directly on the surface of the atomic spheres but rather inside cubical boxes. The placement of each point will affect its contribution to the overall surface area. Also, points lying far from intersections between spheres would represent different surface areas than those next to such intersections. These errors should diminish as the density of the grid increases, but could be expected to be high for low grid densities. Since there was no simple, direct conversion between surface bits and surface area, we developed regression equations relating the sum of the surface bits to the surface area calculated by Pearlman's method.

The surface bits were summed for the 22 compounds for densities from 1 to 5 bla. Table VI and Figure 5 contain the results of the regressions. Once again, the benefits of denser coding of the structures quickly diminish. The standard errors for the surface area estimates are not as good as those for volume. This can be attributed to the approximations noted above and partially to the fact that there are far fewer surface points than volume points. In an attempt to account for the differences in the positions of the points, they were subdivided according to the number of surrounding on bits. Any points with six neighbors would not be on the surface and so would not be considered. The surface bits with five neighbors were summed separately from those having four or fewer neighbors. Those with four neighbors were summed separately from those having three or fewer, and so on. Since each point in the molecule accounts for a cubical volume, those points with only one neighbor were weighted with a value of 5, because of the five sides that would be exposed. By similar reasoning, those with two neighbors were weighted by 4, three neighbors by 3, four neighbors by 2, and one neighbor by 1. The weighted bits were summed. The results of the regressions with the weighted sum are shown in Table VII, and the standard errors

Table VII. Regression Summaries for the Surface Areas of the 22 Compounds Using Weighted Sum of Points

density	R^2	SE	mean	SD	coeff	const
1.0	0.995	6.88	4.2	3.5	0.804	-12.7
2.0	0.999	2.51	1.3	0.8	0.179	-3.69
3.0	1.0	3.34	1.5	0.9	0.078	-1.31
4.0	1.0	1.63	0.9	0.7	0.043	-2.69
5.0	1.0	1.76	0.9	0.8	0.027	-1.17

Table VIII. Calculated Surface Areas and Errors

	Pearlman's method		point-encoded method			
			density = 1		density = 4	
index	SA[a]	cpu seconds	SA	RE[b]	SA	RE
1	143.02	3.69	146.50	2.43	141.46	1.08
2	279.05	10.06	286.40	2.63	281.10	0.73
3	318.91	15.10	315.34	1.11	318.86	0.01
4	50.66	0.44	43.58	13.95	48.86	3.54
5	162.38	4.69	169.01	4.08	161.59	0.48
6	261.87	9.44	249.41	4.75	262.62	0.28
7	250.27	8.78	244.59	2.26	250.64	0.14
8	82.74	0.83	87.00	5.14	83.75	1.21
9	158.19	4.67	164.19	3.79	158.82	0.39
10	59.08	1.07	58.06	1.71	58.06	1.71
11	134.49	5.45	141.67	5.34	135.56	0.79
12	197.93	9.92	210.82	6.51	200.13	1.11
13	259.51	14.51	260.67	0.44	261.84	0.89
14	392.58	22.89	286.10	1.64	388.29	1.09
15	70.51	1.57	61.27	13.09	70.29	0.29
16	62.26	0.91	59.66	4.16	61.87	0.61
17	158.46	5.63	151.32	4.49	158.21	0.15
18	223.50	8.72	226.90	1.52	224.78	0.57
19	114.31	3.60	112.73	1.38	115.60	1.12
20	133.45	5.42	143.28	7.36	134.08	0.47
21	164.22	4.76	160.97	1.97	162.46	1.06
22	153.51	4.78	151.32	1.41	151.96	1.00

[a] Surface areas in Å^2. [b] Relative errors.

are ploted in Figure 6. This treatment decreased the standard errors considerably. A further reduction in error could be probably be obtained if the points were weighted according to elemental type. Regressions were also performed by using the sums of the individual point types with no improvement beyond these results. Table VIII contains some surface areas by molecule as well as time required for Pearlman's method. It should be noted that Pearlman's algorithm can calculate both volume and surface in less time than the sum of the time required for individual calculations. As for the volume calculations, Pearlman's method is superior to ours in terms of the speed of the volume and surface area calculations if the density of bits is high. The two advantages of our method, however, are its simplicity and its capacity to do molecular comparisons, as discussed in the next section.

In both calculations of surface area from surface points, the benefits of denser encoding become small past a density of 3 bla. Figures 5 and 6 indicate that there is probably no gain in accuracy past a density of 4 bla. These regression equations could be made more accurate for specific types of compounds by using very similar compounds to generate the equation. Alternately, many diverse compounds could be used to generate a very general equation.

Molecular Comparison. Perhaps the most interesting of this method's capabilities is its capacity to compare the shapes of different molecules. The differences and similarities between the shapes of molecules can be visualized and/or quantified. This can be done for any number of molecules. If the bit string method is used and bitwise logical functions are available, this comparison is very rapid. As stated before, in addition to atomic coordinates, this function requires that the molecules first be overlapped in space in the orientations that are of interest. Then the grids must either be specified identically, or else, some means of converting between grid types must be developed. The easiest approach is to have the grids for all the involved molecules specified with the same starting and stopping coordinates and of the same density. Once the bit strings are developed, comparisons can be done directly.

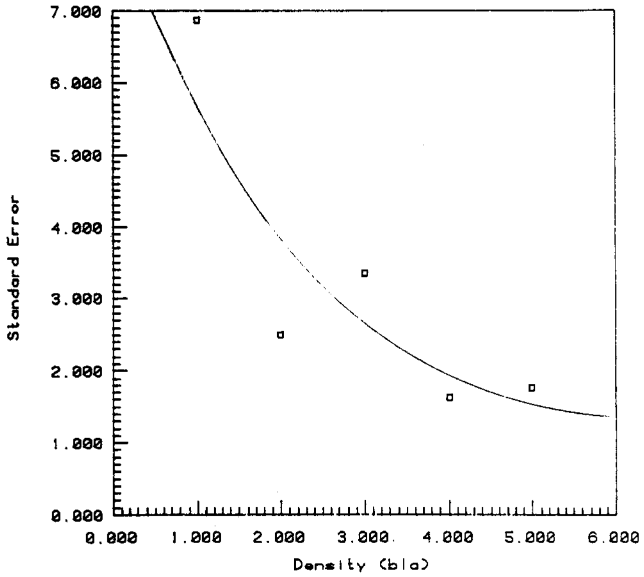

Figure 6. Standard error (Å^2) vs. density (bla). Regression of point-encoded surface area calculations onto those of Pearlman's method. Bit-encoded surface area calculated with a weighted summation of point types.

In Figure 7 are the space-filling (a and b) and bit-encoded (c and d; $d = 6$) representations of p-toluic acid and m-xylene. The benzene rings were overlapped as was one of m-xylene's methyl groups and the carboxyl group of the acid. The grids for these two molecules were identically specified, and the two molecules were compared to yield several different comparison "pseudomolecules". Figure 7e shows the result of an OR

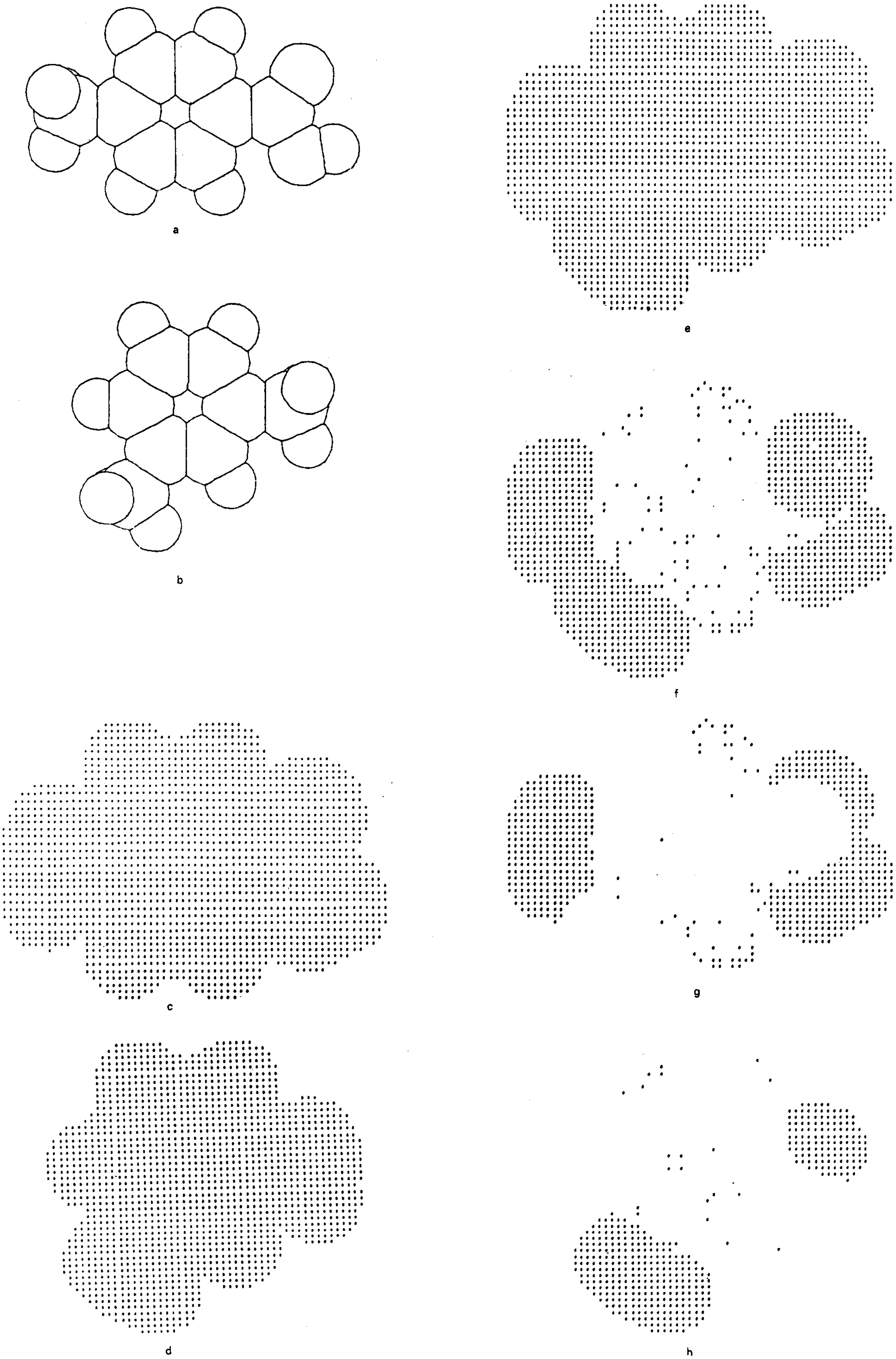

Figure 7. Comparison of the shapes of *p*-toluic acid (a and c) and *m*-xylene (b and d): combined volume (e); unshared volumes (f); exclusive volume of *p*-toluic acid (g) and *m*-xylene (h).

operation of the two structures. This is a composite of both; the ring, *p*- and *m*-methyl groups, and carboxyl–methyl composite are obvious. Figure 7f shows the results of the XOR operation. The XOR operation yields a pseudomolecule that consists of volume elements unique to one structure or another. If this is then ANDed with each of the strings of the original molecule, unique volumes for each result. Panels g and h of Figure 7 show the unique volumes for *p*-toluic acid and *m*-xylene, respectively. The unique methyl groups are clear. The carboxy of the *p*-toluic acid and the other methyl group of the *m*-xylene partially occupy the same space. The carbonyl oxygen and part of the hydroxy group of the acid do not overlap any portion of the other molecule. Two of the methyl hydrogens of the xylene stick out of the plane of the drawing and do not overlap any portion of the acid group and so are unique to the xylene molecule. Any of these pseudomolecules could be taken on and compared with other molecules or pseudomolecules. If a series of molecules were ORed together, a shell of occupation would result. If they were ANDed, only that volume common to all the molecules would remain. The physical significance of such results would depend on the quality of the models and the assumptions inherent in the overlap.

Through knowledge of the relationship between point coordinates and bit position, portions of molecules or pseudomolecules can be isolated and used for further study and comparison. Individual volumes of the molecular space could then be examined in isolation.

Quantification of the volume of portions of a molecule or pseudomolecule may be valuable as an aid to visualization or as a quantitative index of similarity in some specific region of the molecule. The volume of the *p*-methyl group of *p*-toluic acid was calculated to be 15.3 Å. This is the volume of the methyl group minus that of the hydrogen, which would have been at that location in the *m*-xylene. This gives a quantitative index of the difference in bulk at a specific site. While this example has little chemical value in itself, similar information could prove useful in structure–activity analysis. The exact information required will vary between studies, and this method is versatile enough to yield many different indices of similarity and difference.

At high densities the bit encoding can be time consuming, but comparison, even between many molecules, can be done rapidly by this method. Also, qualitative information may suffice for some problems. In that case, a sparse encoding may yield sufficient information and save on time during the encoding state of analysis.

CONCLUSIONS

We have presented a simple computational system for the definition and manipulation of molecular shape that can also be used to calculate molecular volume and surface area. We have elucidated the relationship between visualization of shape and the accuracy of the physical property calculations and the density of the shape definition, the one adjustable parameter affecting these calculations. This method can provide estimates of volume and surface area that are as accurate as those of other computational systems. This method can deal equally easily with whole molecules, molecular fragments, pseudomolecules formed from fragments or by comparison of two or more structures, or fragments of these. Bit strings representing even densely encoded structures could be stored on disk without requiring massive amounts of storage. It could be generally useful for molecular-shape analysis and application to structure–activity analysis. Elementary programming skills will suffice for the implementation of this system, and the algorithms and programming considerations have been detailed. The mathematics and graphics are simple and do not require advanced or expensive equipment. Even with time-saving refinements, this method will be time consuming at high densities. Still, the simplicity and availability of this method may compensate for this drawback. In some instances where high accuracy is necessary, time can be saved by examining only isolated portions of a molecule, if this approach is amenable to the study.

This method does not provide information that cannot be obtained from other systems. Other methods are available for graphics display, volume and surface area calculations, calculation and display of excluded or overlap volumes, and molecular-shape analysis. Often, however, each of these steps requires large, complex programs. Also, the systems that have been developed to perform these functions have progressively become less available to those with a casual interest due to commercial ventures. This method sould be valuable to those who do not have the resources to obtain expensive hardware and software systems but who still want to investigate molecular shape.

There are many other ways of implementing the general methodology shown here. The algorithms and programming steps we have outlined are direct and simple. Many variations could be made in this way alone that would have both benefits and drawbacks.

Of course, this method cannot be used in isolation. Three-dimensional atomic coordinates must be provided, and if comparisons are to be made, structures must be overlapped through some other means. Any conformational analysis is outside its realm. If used in structure–activity relationship studies, unless the problem is simple or is controlled solely by steric factors, it cannot stand alone; other information will also be necessary. This method is useful only in dealing with the shapes of rigid molecules of predetermined conformation. For this purpose, however, it should prove generally useful.

ACKNOWLEDGMENT

This research was supported by the U.S. Environmental Protection Agency under Cooperative Research Agreement CR 807531. The contents do not necessarily reflect the views of the Agency, and no official endorsement should be inferred. The PRIME 750 computer used in these studies was purchased, in part, with the support of the National Science Foundation.

REFERENCES AND NOTES

(1) Connolly, Michael L. *Science (Washington, D.C.)* **1983**, *221* (4612), 709–713.

(2) Nyburg, S. C. *Acta Crystallogr., Sect. B: Struct. Crystallogr. Cryst. Chem.* **1974**, *B30*, 251–253.

(3) Barino, Luisa *Comput. Chem.* **1981**, *5* (2–3), 85–90.

(4) Pearlman, R. S. In "Physical Chemical Properties of Drugs"; Yalkowsky, S. H.; Sinkula, A. A.; Valvani, S. C., Eds.; Marcel Dekker: New York, 1980; pp 321–347.

(5) Connolly, Michael L. *Science (Washington, D.C.)* **1983**, *221* (4612), 709–713.

(6) Hopfinger, A. J. *J. Am. Chem. Soc.* **1980**, *102*, 7196–7206. Hopfinger, A. J.; Potenzone, R., Jr. *Mol. Pharmacol.* **1982**, *21*, 187–195.

(7) Stuper, A. J.; Brugger, W. E.; Jurs, P. C. "Computer Assisted Studies of Chemical Structure and Biological Function"; Wiley-Interscience: New York, 1979.

(8) Ryan, Thomas A., Jr.; Joiner, Brian L.; Ryan, Barbara F. "Minitab Handbook", 2nd ed.; Duxbury Press: Boston, 1985.

(9) Marsili, Mario; Floersheim, Philipp; Dreiding, Andre S. *Comput. Chem.* **1983**, *7* (4), 175–181.

(10) Kier, L. B.; Hall, L. H.; "Molecular Connectivity in Chemistry and Drug Research"; Academic Press: New York, 1976.

(11) QCPE Program 395.

(12) Henry, D. R.; Jurs, P. C., unpublished results.