Fast Similarity Search in Three-Dimensional Structure Databases

Xiong Wang and Jason T. L. Wang*

Department of Computer and Information Science, New Jersey Institute of Technology, Newark, New Jersey 07102

Received July 29, 1999

Given a database *D* of three-dimensional (3D) molecular structures and a target molecule *Q*, the similarity search problem is to find the molecules *O* in *D* that match *Q* after allowing for an arbitrary number of whole-structure rotations and translations as well as a certain number of edit operations. The edit operations include relabeling an atom, deleting an atom, and inserting an atom. This search operation arises in many biochemical applications. In this paper we study the similarity search problem and a class of related queries. We present a computer vision based technique, called geometric hashing, for processing these queries. Experimental results on a database of 3D molecular structures obtained from the National Cancer Institute indicate the good performance of the presented technique.

1. INTRODUCTION

Given a database *D* of three-dimensional (3D) molecular structures and a target molecule *Q*, the similarity search problem is to find the molecules *O* in *D* that *approximately* match *Q*; i.e., *O* matches *Q* after allowing for an arbitrary number of whole-structure rotations and translations as well as a certain number of edit operations. The edit operations include relabeling an atom, deleting an atom, and inserting an atom. These edit operations are an extension of the edit operations for strings,[26] trees,[28] and two-dimensional (2D) graphs.[47]

Each atom in a molecule has a 3D coordinate. Each atom also has a name, which is derived from the name of the underlying atomic element. We assume that each atom is identified by a unique, user-assigned number in the molecule. The molecule can be divided into one or more *rigid* substructures. For example, a ring is a rigid substructure. Formally, a rigid substructure is a subgraph in which no rotation is possible if its component atoms are spatially fixed with respect to one another. Notice that the rigid substructure as a whole can be rotated (we refer to this as a "whole-structure" rotation or simply a rotation when the context is clear). That is to say, the relative position of an atom in the substructure and an atom outside the substructure can be changed under the rotation. Thus if we consider a molecule as a 3D graph in which each atom is a node and each bond is an edge, a block[21] of the graph could be a rigid substructure; two rigid substructures may be connected by an edge and they may be rotatable with respect to each other around the edge.

As an example, consider the molecule *O* in Figure 1 containing two rigid substructures. Atoms in the substructures are numbered 0, 1, 2, 3, 4, 5 and 6, 7, 8, 9, 10, respectively. Atom names are enclosed in parentheses; they are hypothetical ones solely used for illustration purposes. Table 1 shows

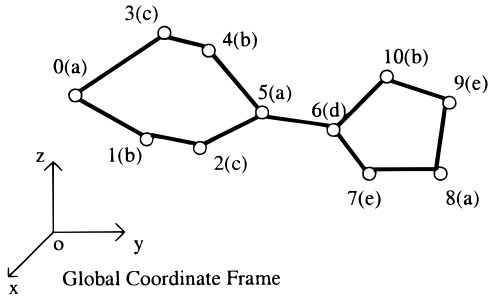

Figure 1. An example molecule.

Table 1. Identification Numbers, Names, and Global Coordinates of the Atoms of the Molecule in Figure 1

atom no.	atom name	global coordinates
0	a	(1.0178, 1.0048, 2.5101)
1	b	(1.2021, 2.0410, 2.0020)
2	c	(1.3960, 2.9864, 2.0006)
3	c	(0.7126, 2.0490, 3.1921)
4	b	(0.7610, 2.7125, 3.0124)
5	a	(1.0097, 3.6478, 2.2660)
6	d	(1.1329, 4.5002, 2.2024)
7	e	(1.5309, 5.2026, 1.7191)
8	a	(1.4529, 6.1015, 1.5712)
9	e	(1.0356, 6.0030, 2.2820)
10	b	(0.7359, 5.0571, 2.6857)

the 3D coordinates of the atoms with respect to the global coordinate frame. We divide the molecule into two rigid substructures: S_0 and S_1. S_0 consists of atoms numbered 0, 1, 2, 3, 4, and 5 as well as bonds connecting the atoms (Figure 2a). S_1 consists of atoms numbered 6, 7, 8, 9, and 10 as well as bonds connecting them (Figure 2b). The two substructures are rotatable with respect to each other around the bond {5, 6} that connects S_0 and S_1. We refer to {5, 6} as a *common bond*. In general, a common bond is one that connects two rigid substructures in a molecule. Note that a rigid substructure is not necessarily complete. [A complete graph is one where every node has a direct connection to every other node; that is, every node is connected to every other node by an edge.] For example, in Figure 2a, there is

* Corresponding author. E-mail: jason@cis.njit.edu. Phone: (973) 596-3396. Fax: (973) 596-5777. Part of the work of this author was done while visiting Courant Institute of Mathematical Sciences, New York University.

10.1021/ci990081m CCC: $19.00
Published on Web 02/09/2000

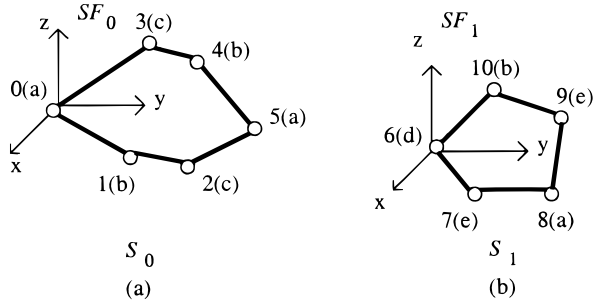

Figure 2. The rigid substructures of the molecule in Figure 1.

no bond connecting the atom numbered 1 and the atom numbered 3.

We attach a local coordinate frame to each substructure. For instance, let us focus on the substructure S_0 in Figure 2. We attach a local coordinate frame to S_0 whose origin is the atom numbered 0. This local coordinate frame is represented by three basis points P_{b_1}, P_{b_2}, and P_{b_3}, with coordinates P_{b_1}-(x_0,y_0,z_0), $P_{b_2}(x_0+1,y_0,z_0)$, and $P_{b_3}(x_0,y_0+1,z_0)$, respectively. The origin is P_{b_1} and the three basis vectors are $\vec{V}_{b_1,b_2}$, $\vec{V}_{b_1,b_3}$, and $\vec{V}_{b_1,b_2} \times \vec{V}_{b_1,b_3}$. Here, $\vec{V}_{b_1,b_2}$ represents the vector starting at point P_{b_1} and ending at point P_{b_2}. $\vec{V}_{b_1,b_2} \times \vec{V}_{b_1,b_3}$ stands for the cross product of the two corresponding vectors. We refer to this coordinate frame as substructure frame 0, denoted SF_0. [For space saving reasons, we use the cross product, rather than an additional basis point $P_{b_4}(x_0,y_0,z_0+1)$, to define the third basis vector of SF_0. As it will become clear, we store the coordinates of the basis points of SF_0 in a hash table. Storing the coordinate of the additional basis point P_{b_4} would incur extra storage overhead, and thus is avoided.] Note that the basis vectors of SF_0 are orthonormal. That is, the length of each vector is 1 and the angle between any two basis vectors has 90°. Also note that, for any atom numbered i in the substructure S_0 with global coordinate P_i-(x_i,y_i,z_i), we can find a local coordinate of the atom i with respect to SF_0, denoted P_i', where

$$P_i' = \vec{V}_{b_1,i} = (x_i - x_0, y_i - y_0, z_i - z_0)$$

1.1. Similarity Search and Related Queries. We use the edit distance to measure the similarity of two molecules. There are three types of edit operations: relabeling an atom, deleting an atom, and inserting an atom. Relabeling an atom v means to change the name of v to any valid name that differs from its original name. Deleting an atom v from a molecule means to remove v from the 3D Euclidean space and make the bonds touching v connect with one of its neighbors v'. Inserting an atom v into a molecule means to add v to the 3D Euclidean space and make a node v' and a subset of its neighbors become the neighbors of v. Notice that when an atom v is inserted or deleted, the atoms surrounding v do not move; i.e., their coordinates remain the same. Figure 3 illustrates the edit operations, where molecule O_2 results from the application of an edit operation to molecule O_1.

Our definition of edit operations is really a shorthand for the specification. Here is the specification in full detail. Consider a single edit operation, e.g., one that transforms O_1 to O_2 in Figure 3. If it is a relabeling operation, we specify the atom to be relabeled in O_1. If it is an insert operation,

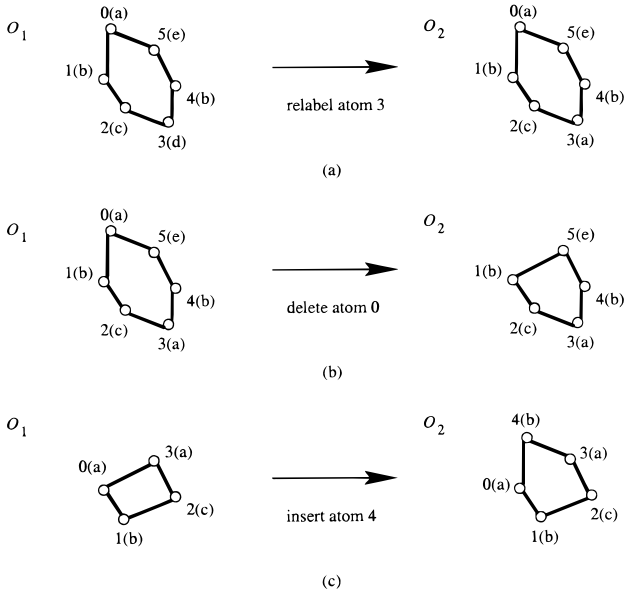

Figure 3. Illustration of the edit operations: (a) relabeling the atom numbered 3, to change its name "d" to the name "a"; (b) deleting the atom numbered 0; (c) inserting the atom numbered 4.

we must specify the atom v' that is the neighbor of the atom v to be inserted, and which subset of the neighbors of v' will be the neighbors of v. The same holds for a delete operation. We say that the edit distance, or simply the *distance* when the context is clear, between molecule O and molecule O' is n, or O approximately matches O' with distance n, if by applying an arbitrary number of rotations and translations as well as n nonredundant atom insert, delete, or relabeling operations one can transform O to O'. [Redundant edit operations refer to edit operations that have a reverse effect, e.g., inserting an atom v and then deleting the same atom v.] In practice, there may exist several different sets of n nonredundant edit operations for transforming O to O'. Our algorithms find one such set of n nonredundant edit operations to transform O to O' (or superimpose O on O').

The queries we are concerned with are categorized as follows: Given a target molecule Q and a database D of 3D molecules,

(similarity search or good-match retrieval[29]) find the molecules in D that approximately match Q, i.e., those that are within some distance, say ϵ, of Q

(k-closest retrieval) find the k molecules, for some user-specified k, in D that are closest to Q

(best-match retrieval[29]) find the closest (i.e., most similar) molecule of Q in D [This query is a special case for the k-closest retrieval where $k = 1$. The latter retrieves not only the closest molecule, but the ith, $i = 2, ..., k$, closest molecule of Q in D.]

(bad-match retrieval) find the molecules in D that are sufficiently dissimilar to Q, i.e., those that are beyond distance ϵ of Q

(k-farthest retrieval) find the k molecules in D that are farthest from Q

(worst-match retrieval[22]) find the farthest (i.e., most dissimilar) molecule of Q in D

2. PRELIMINARIES

Our approach to processing the above queries is composed of two phases. In the preprocessing phase, molecules in the database are divided into rigid substructures. These substructures are hashed into a three-dimensional disk-based hash table. In the on-line searching phase, we divide the target molecule into rigid substructures and hash the substructures using the same hash function as used in the preprocessing phase. We then locate the substructures of the data molecules that match with the substructures of the target molecule. The matched substructures are then augmented, wherever appropriate, to form larger matches. To facilitate augmentation, we maintain a common bond table, which lists pairs of substructures that are connected by a common bond in a molecule in the database.

2.1. The Common Bond Table. When a molecule is large, processing it in its entirety would be costly in both time and space. Our strategy is to decompose the molecule into rigid substructures, where the substructures are rotatable with respect to each other around a common bond. Specifically, we break a molecule O into maximally sized rigid substructures (recall that a rigid substructure is a subgraph in which no rotation is possible if its component atoms are spatially fixed with respect to one another). We use an approach similar to ref 21 that employs a depth-first search algorithm to find blocks in molecules. Each block is a rigid substructure. We then merge two rigid substructures B_1 and B_2 if they are not rotatable with respect to each other; that is, the relative position of an atom $n_1 \in B_1$ and an atom $n_2 \in B_2$ is fixed. The algorithm runs in time linearly proportional to the number of bonds in the molecule O. The result is a collection C of rigid substructures where any two substructures in C are connected by at most one common bond. In each substructure, one atom o_r is distinguished and used as the origin of the local coordinate frame attached to the substructure (cf. Figure 2). The atom o_r is chosen randomly, and designating any atom as the origin will not affect the result of the proposed approach.

We maintain a table of common bonds for the molecules in the database. Each tuple in the table has the form

$$(O{\cdot}id, S_x, S_y, S_x{\cdot}P_{b_1}, S_y{\cdot}P_{b_1}, S_x{\cdot}EP_1, S_y{\cdot}EP_2)$$

where $O{\cdot}id$ is the identification number for the molecule O, S_x, and S_y are two rigid substructures in O, $S_x{\cdot}P_{b_1}$ and $S_y{\cdot}P_{b_1}$ are the atom numbers of the origins of the local coordinate frames attached to the two substructures respectively, and $S_x{\cdot}EP_1$ and $S_y{\cdot}EP_2$ are the identification numbers of the end atoms of the common bond between S_x and S_y. For example, consider again the molecule in Figure 1 and its rigid substructures in Figure 2. Suppose the identification number of the molecule is 12 and the atoms numbered 0 and 6 are chosen as the origins of the local coordinate frames attached to S_0 and S_1, respectively. Then there is a tuple (12, S_0, S_1, 0, 6, 5, 6) in the common bond table, indicating the fact that S_0 and S_1 are connected via the common bond {5, 6}.

2.2. Encoding Atom and Name Triplets. In processing a rigid substructure of a 3D molecule, we choose all three-atom combinations, referred to as atom triplets, in the substructure and hash the atom triplets. (The names of the three atoms in an atom triplet form a name triplet.) We hash three-atom combinations, because to fix a rigid substructure in the 3D Euclidean space one needs at least three atoms from the substructure and three atoms are sufficient provided they are not collinear. Notice that the proper order of choosing the atoms v_1, v_2, v_3 in a triplet is significant. We determine the order of the three atoms by considering the triangle formed by them. The first atom chosen always opposes the longest bond of the triangle, and the third atom chosen opposes the shortest bond. Thus, the order is unique if the triangle is not isosceles or equilateral, which usually holds when the coordinates are floating point numbers. In other cases, we store all configurations obeying the longest−shortest rule described above.

Suppose the three atoms chosen are v_1, v_2, v_3, in that order. We encode this atom triplet and the corresponding name triplet as follows. The code for the atom triplet is an unsigned long integer, defined as $((N_1 \times 1000 + N_2) \times 1000) + N_3$, where N_1, N_2, and N_3 are the identification numbers of v_1, v_2, and v_3, respectively. Here 1000 is an adjustable parameter value. As long as the number of atoms in a molecule is less than 1000, the code of an atom triplet is unique.

We maintain all atom names in an array A. The code for the corresponding name triplet is also an unsigned long integer, defined as $((L_1 \times 1000 + L_2) \times 1000) + L_3$, where L_1, L_2, and L_3 are the indices for the atom names of v_1, v_2, and v_3, respectively, in the array A. Thus, if the size of the atom name alphabet is less than 1000, the code of a name triplet is unique.

As an example, consider again the molecule O in Figure 1. Suppose the array A has the following entries:

index	0	1	2	3	4
atom name	a	b	c	d	e

Consider, for example, the atoms with identification numbers 1, 2, and 3 in O. The code for this atom triplet is $((1 \times 1000 + 2) \times 1000) + 3 = 1\,002\,003$. The names of the three atoms are "b", "c", and "c", respectively. Referring to the array A above, the indices for these atom names are 1, 2, and 2, respectively. Thus, the code for the corresponding name triplet is $((1 \times 1000 + 2) \times 1000) + 2 = 1\,002\,002$.

3. OUR APPROACH

After explaining the basic concepts, we now turn to the description of the proposed approach. Our approach is composed of two phases: the preprocessing phase and the on-line searching phase. We first present the algorithm used in the preprocessing phase. Then we discuss the on-line phase, followed by the algorithm used to augment substructure matches. Finally we describe the algorithms for similarity search and related queries.

3.1. Preprocessing Phase. We choose all atom triplets in each rigid substructure of the molecules in the database, and hash them into a 3D disk-based hash table. For example, consider again the substructure S_0 in Figure 2a. We choose all atom triplets in the substructure and calculate their three-dimensional hash function values as follows. Suppose the chosen atoms are numbered i, j, k in that order and have global coordinates $P_i(x_i,y_i,z_i)$, $P_j(x_j,y_j,z_j)$, and $P_k(x_k,y_k,z_k)$, respectively. Calculate l_1, l_2, l_3 where

$$l_1 = \text{Round}(((x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2) \times 100)$$

$$l_2 = \text{Round}(((x_i - x_k)^2 + (y_i - y_k)^2 + (z_i - z_k)^2) \times 100)$$

$$l_3 = \text{Round}(((x_k - x_j)^2 + (y_k - y_j)^2 + (z_k - z_j)^2) \times 100)$$

Here, we use a multiplier 100; we multiply the numbers by 100 and round the floating point numbers to integers. The reason for using the multiplier is that we want some digits following the decimal point to contribute to the distribution of the hash function values. We ignore the digits after the second position because they are inaccurate. The multiplier is a parameter whose value is determined in experiments and is adjustable for different data. Let

$$d_1 = (l_1 + l_2) \bmod Prime_1 \bmod Nrow$$

$$d_2 = (l_2 + l_3) \bmod Prime_2 \bmod Nrow$$

$$d_3 = (l_3 + l_1) \bmod Prime_3 \bmod Nrow$$

$Prime_1$, $Prime_2$, and $Prime_3$ are three prime numbers and *Nrow* is the cardinality of the hash table in each dimension. The atom triplet [*i*, *j*, *k*] will be hashed to the three-dimensional bucket $h[d_1][d_2][d_3]$. Intuitively we use the squares of the lengths of the three bonds connecting the three chosen atoms to determine the bucket address.

We use three different prime numbers $Prime_1$, $Prime_2$, $Prime_3$ here in the hope that the distribution of the hash function values is not skewed even if pairs of l_1, l_2, l_3 are correlated. In general, these prime numbers have to be chosen properly based on the range of the coordinates of the atoms in the molecules. If they are too large relative to the sums of $(l_i + l_j)$, they serve no purpose at all since if $M > N$, then $(N \bmod M) = N$. On the other hand, if they are too small, a lot of collisions and overflows would occur in the hash table.[17]

We store several items associated with the atom triplet [*i*, *j*, *k*] in the bucket $h[d_1][d_2][d_3]$: the identification number of its molecule, the identification number of its substructure, the code for the atom triplet, and the code for the name triplet. In addition, we store the coordinates of the basis points P_{b_1}, P_{b_2}, P_{b_3} of substructure frame 0 (SF_0) with respect to the three chosen atoms. Specifically, suppose the chosen atoms *i*, *j*, *k* are not collinear. We can construct another local coordinate frame, denoted *LF*[*i*, *j*, *k*], using $\vec{V}_{i,j}$, $\vec{V}_{i,k}$, and $\vec{V}_{i,j} \times \vec{V}_{i,k}$ as basis vectors. The coordinates of P_{b_1}, P_{b_2}, P_{b_3} with respect to the local coordinate frame *LF*[*i*, *j*, *k*], denoted SF_0-[*i*, *j*, *k*], form a 3×3 matrix, which is calculated as follows:

$$SF_0[i, j, k] = \begin{pmatrix} \vec{V}_{i,b_1} \\ \vec{V}_{i,b_2} \\ \vec{V}_{i,b_3} \end{pmatrix} \times A^{-1}$$

where

$$A = \begin{pmatrix} \vec{V}_{i,j} \\ \vec{V}_{i,k} \\ \vec{V}_{i,j} \times \vec{V}_{i,k} \end{pmatrix}$$

Thus, for example, the hash table entry for the three chosen atoms *i*, *j*, *k* from the substructure S_0 in Figure 2a is (12, 0, *Ncode*, *Lcode*, SF_0[*i*, *j*, *k*]), where *Ncode* is the code for the atom triplet and *Lcode* is the code for the name triplet. Since there are six atoms in S_0, we have $\binom{6}{3} = 20$ possible atom triplets in S_0 and therefore 20 entries in the hash table for this substructure.

$$P_0'(0.0000, 0.0000, 0.0000)$$
$$P_1'(0.1843, 1.0362, -0.5081)$$
$$P_2'(0.3782, 1.9816, -0.5095)$$
$$P_3'(-0.3052, 1.0442, 0.6820)$$
$$P_4'(-0.2568, 1.7077, 0.5023)$$

Figure 4. The local coordinates, with respect to SF_0, of the atoms numbered 0, 1, 2, 3, 4 in the substructure S_0 in Figure 2a.

To illustrate the hashing process, consider the coordinates of the atoms of S_0 in Table 1. The basis points P_{b_1}, P_{b_2}, P_{b_3} of SF_0 have global coordinates

$$P_{b_1}(1.0178,1.0048,2.5101)$$

$$P_{b_2}(2.0178,1.0048,2.5101)$$

$$P_{b_3}(1.0178,2.0048,2.5101)$$

Thus, for example, Figure 4 shows the local coordinates, with respect to SF_0, of the atoms numbered 0, 1, 2, 3, and 4 in the substructure S_0. Now let $Prime_1$, $Prime_2$, and $Prime_3$ be 1009, 1033, and 1057, respectively, and let *Nrow* be 31. Thus, for example, for the atoms numbered 1, 2, and 3 in S_0, the bucket address is h[10][7][7] and

$$SF_0[1,2,3] = \begin{pmatrix} -1.0567 & 0.3578 & 0.1739 \\ -0.8758 & 0.0719 & 0.9072 \\ -0.0359 & 0.4175 & 0.0240 \end{pmatrix}$$

As another example, for the atoms numbered 1, 4, and 2 in S_0, the bucket address is h[26][6][6] and

$$SF_0[1,4,2] = \begin{pmatrix} 0.3694 & -1.3082 & -0.2429 \\ -0.0435 & -0.8571 & -1.0067 \\ 0.4552 & -0.3437 & -0.0869 \end{pmatrix}$$

Similarly, for the substructure S_1, we attach a local coordinate frame SF_1 to the atom numbered 6 as shown in Figure 2b. There are 20 hash table entries for the substructure S_1, each having the form (12, 1, *Ncode*, *Lcode*, SF_1[*l*,*m*,*n*]) where *l*, *m*, *n* are any three atoms in S_1.

3.2. On-Line Phase. To facilitate detecting substructure matches, we associate an atom_match_list and a relabeling_counter with each rigid substructure of the molecules in the database. Given a target molecule *Q*, we divide *Q* into rigid substructures and hash the substructures using the same hash function as in the preprocessing phase. Then we update the atom_match_list and relabeling_counter as illustrated below.

Let us focus on the substructure S_0 of the molecule with identification number 12 shown in Figure 2a. Suppose *i*, *j*, *k* are three atoms in the substructure S_0. Then its entry in the hash table is (12, 0, *Ncode*, *Lcode*, SF_0[*i*,*j*,*k*]). Let *u*, *v*, *w* be three atoms in the target molecule *Q* that have the same bucket address as *i*, *j*, *k* (i.e., the atom triplet [*u*, *v*, *w*] hits

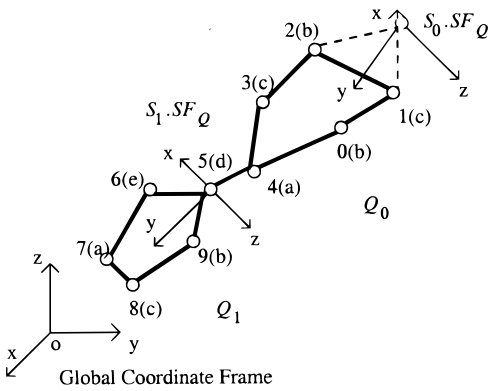

Figure 5. An example target molecule.

the substructure S_0). We decode *Ncode* to get *i, j, k* and add them into the atom_match_list of S_0. This records that *u* geometrically matches *i* (i.e., they have the same 3D coordinate), *v* geometrically matches *j*, and *w* geometrically matches *k*. We also decode *Lcode* and determine whether two geometrically matching atoms have the same name. If not, the relabeling_counter is updated to reflect the fact that there is a relabeling between the two geometrically matching atoms.

In addition, we calculate $S_0 \cdot SF_Q$ where

$$S_0 \cdot SF_Q = SF_0[i,j,k] \times \begin{pmatrix} \vec{V}_{u,v} \\ \vec{V}_{u,w} \\ \vec{V}_{u,v} \times \vec{V}_{u,w} \end{pmatrix} + \begin{pmatrix} P_u \\ P_u \\ P_u \end{pmatrix}$$

The matrix $S_0 \cdot SF_Q$ contains the coordinates of the three basis points of the substructure frame 0 (SF_0) with respect to the global coordinate frame in which the target molecule *Q* is given. In general, there may be several atom triplets of *Q* that hit S_0. We update the atom_match_list and relabeling_counter of S_0 for these matching atom triplets only if they yield the same $S_0 \cdot SF_Q$.

It is likely that an atom triplet [*u, v, w*] of *Q* has the same bucket address as an atom triplet [*i, j, k*] of the substructure S_0, though they do not match geometrically. This is referred to as a *false match*. Let P_{c_1}, P_{c_2}, and P_{c_3} be the three basis points forming $S_0 \cdot SF_Q$ where P_{c_1} is the origin. It can be shown that $\vec{V}_{c_1,c_2}$, $\vec{V}_{c_1,c_3}$, and $\vec{V}_{c_1,c_2} \times \vec{V}_{c_1,c_3}$ are orthonormal vectors if and only if the atoms *u, v,* and *w* geometrically match the atoms *i, j,* and *k*, respectively. This is a theoretical criterion based on which one can detect and elimninate a false match. In practice, let the *base matrix* $S_0 \cdot E$ for $S_0 \cdot SF_Q$ be

$$S_0 \cdot E = \begin{pmatrix} \vec{V}_{c_1,c_2} \\ \vec{V}_{c_1,c_3} \\ \vec{V}_{c_1,c_2} \times \vec{V}_{c_1,c_3} \end{pmatrix}$$

We note that if $\vec{V}_{c_1,c_2}$, $\vec{V}_{c_1,c_3}$, and $\vec{V}_{c_1,c_2} \times \vec{V}_{c_1,c_3}$ are orthonormal vectors, then $|S_0 \cdot E| = 1$. Thus a practically useful criterion for detecting and eliminating false matches is to check whether or not $|S_0 \cdot E| = 1$. If $|S_0 \cdot E| \neq 1$, then $\vec{V}_{c_1,c_2}$, $\vec{V}_{c_1,c_3}$, and $\vec{V}_{c_1,c_2} \times \vec{V}_{c_1,c_3}$ are not orthonormal vectors, and therefore the atoms *u, v,* and *w* do not match the atoms *i, j,* and *k* geometrically.

To illustrate the on-line process, consider the target molecule *Q* in Figure 5. *Q* contains two rigid substructures Q_0 and Q_1. Table 2 lists the identification numbers, names, and global coordinates of the atoms in *Q*. In Q_0, the atoms numbered 0, 1, 2, 3, 4 match, after rotation, the atoms numbered 4, 3, 1, 2, 5 in the substructure S_0 in Figure 2a. The atom numbered 0 in S_0 does not appear in Q_0 (i.e., it is to be deleted). Thus, for example, for the atoms numbered 2, 3, and 1 in Q_0, the bucket address in the three-dimensional hash table is *h*[10][7][7], which is the same as the bucket address for the atom triplet [1, 2, 3] in S_0, and

Table 2. Identification Numbers, Names, and Global Coordinates of the Atoms of the Molecule in Figure 5

atom no.	atom name	global coordinates
0	b	(−0.269 000, 4.153 153, 2.911 494)
1	c	(−0.317 400, 4.749 386, 3.253 592)
2	b	(0.172 100, 3.913 515, 4.100 777)
3	c	(0.366 000, 3.244 026, 3.433 268)
4	a	(−0.020 300, 2.964 012, 2.777 921)
5	d	(0.102 900, 2.316 302, 2.220 155)
6	e	(0.500 900, 1.477 885, 2.065 228)
7	a	(0.422 900, 0.737 686, 1.534 191)
8	c	(0.005 600, 1.309 948, 1.101 230)
9	b	(−0.294 100, 2.264 259, 1.484 623)

$$S_0 \cdot SF_Q = \begin{pmatrix} -0.012200 & 5.005500 & 4.474200 \\ 0.987800 & 5.005500 & 4.474200 \\ -0.012200 & 4.298393 & 3.767093 \end{pmatrix}$$

For the atoms numbered 2, 0, and 3 in Q_0, the bucket address is *h*[26][6][6], which is the same as the bucket address for the atom triplet [1, 4, 2] in S_0, and

$$S_0 \cdot SF_Q = \begin{pmatrix} -0.012200 & 5.005500 & 4.474200 \\ 0.987800 & 5.005500 & 4.474200 \\ -0.012200 & 4.298393 & 3.767093 \end{pmatrix}$$

These two matches (hits) have the same $S_0 \cdot SF_Q$, and therefore the atom_match_list for the substructure S_0 includes the atoms numbered 1, 2, 3, and 4. After hashing all atom triplets of Q_0, the atom_match_list of S_0 will include the atoms numbered 1, 2, 3, 4, and 5. Since the corresponding names of the matching atoms are the same, the relabeling_counter of S_0 is 0.

Note that, for any atom *i* in the substructure Q_0 with global coordinate $P_i(x_i,y_i,z_i)$, it has a local coordinate with respect to $S_0 \cdot SF_Q$, denoted P_i', where

$$P_i' = \vec{V}_{c_1,i} \times S_0 \cdot E$$

Here P_{c_1} is the origin of $S_0 \cdot SF_Q$; $S_0 \cdot E$ is the base matrix of $S_0 \cdot SF_Q$. Thus, for example, the local coordinates, with respect to $S_0 \cdot SF_Q$, of the atoms numbered 2, 3, and 1 in Q_0 are

$$P_2'(0.184300, 1.036200, -0.508100)$$

$$P_3'(0.378200, 1.981600, -0.509500)$$

$$P_1'(-0.305200, 1.044200, 0.682000)$$

They match the local coordinates, with respect to SF_0, of the atoms numbered 1, 2, and 3 in the substructure S_0 (cf. Figure 4). Likewise, the local coordinate, with respect to $S_0 \cdot SF_Q$, of atom 0 in Q_0 is

$$P_0'(-0.256800, 1.707700, 0.502300)$$

which matches the local coordinate, with respect to SF_0, of atom 4 in the substructure S_0 (cf. Figure 4).

Similarly, in Q_1, the atoms numbered 5, 6, 7, 8, 9 match, after rotation, the atoms numbered 6, 7, 8, 9, 10 in the substructure S_1 in Figure 2b. The atom_match_list of the substructure S_1 includes atoms 6, 7, 8, 9, and 10 after hashing all atom triplets in Q_1. The relabeling_counter for S_1 is 1, since the name of the atom numbered 8 in Q_1 differs from that of the atom numbered 9 in S_1.

3.3. Augmenting Substructure Matches. Substructure matches with the same molecule identification number may be augmented by utilizing the common bond table. Suppose that, in the common bond table, there is a tuple

$$(O{\cdot}id, S_x, S_y, S_x{\cdot}P_{b_1}, S_y{\cdot}P_{b_1}, S_x{\cdot}EP_1, S_y{\cdot}EP_2)$$

for two substructures S_x and S_y in a molecule O in the database. Let SF_x represent the local coordinate frame attached to S_x and SF_y represent the local coordinate frame attached to S_y. Suppose that, after hashing all atom triplets of the target molecule Q, $S_x{\cdot}SF_Q$ ($S_y{\cdot}SF_Q$, respectively) contains the coordinates of the three basis points of SF_x (SF_y, respectively) with respect to the global coordinate frame in which Q is given. The base matrix for $S_x{\cdot}SF_Q$ ($S_y{\cdot}SF_Q$, respectively) is $S_x{\cdot}E$ ($S_y{\cdot}E$, respectively). $S_x{\cdot}P_{c_1}$ ($S_y{\cdot}P_{c_1}$, respectively) is the origin of $S_x{\cdot}SF_Q$ ($S_y{\cdot}SF_Q$, respectively).

Let

$$S_x{\cdot}EP_1' = \vec{V}_{S_x{\cdot}P_{b1},S_x{\cdot}EP_1} \times S_x{\cdot}E + S_x{\cdot}P_{c_1}$$

$$S_x{\cdot}EP_2' = \vec{V}_{S_x{\cdot}P_{b1},S_y{\cdot}EP_2} \times S_x{\cdot}E + S_x{\cdot}P_{c_1}$$

$$S_y{\cdot}EP_1' = \vec{V}_{S_y{\cdot}P_{b1},S_x{\cdot}EP_1} \times S_y{\cdot}E + S_y{\cdot}P_{c_1}$$

$$S_y{\cdot}EP_2' = \vec{V}_{S_y{\cdot}P_{b1},S_y{\cdot}EP_2} \times S_y{\cdot}E + S_y{\cdot}P_{c_1}$$

$\vec{V}_{S_x{\cdot}P_{b1},S_x{\cdot}EP_1}$ ($\vec{V}_{S_x{\cdot}P_{b1},S_y{\cdot}EP_2}$, respectively) represents the coordinate of $S_x{\cdot}EP_1$ ($S_y{\cdot}EP_2$, respectively) with respect to the local coordinate frame SF_x. $\vec{V}_{S_y{\cdot}P_{b1},S_x{\cdot}EP_1}$ ($\vec{V}_{S_y{\cdot}P_{b1},S_y{\cdot}EP_2}$, respectively) represents the coordinate of $S_x{\cdot}EP_1$ ($S_y{\cdot}EP_2$, respectively) with respect to the local coordinate frame SF_y. $S_x{\cdot}EP_1'$ represents the coordinate of the first end atom of the common bond between S_x and S_y with respect to the global coordinate frame in which the target molecule Q is given when matching S_x with Q. $S_y{\cdot}EP_1'$ represents the coordinate of the first end atom of the common bond between S_x and S_y with respect to the global coordinate frame in which the target molecule Q is given when matching S_y with Q.

Suppose the substructure Q_1 of Q matches the substructure S_x of the molecule O and the substructure Q_2 of Q matches the substructure S_y of O. The two substructure matches are said to be *augmentable* if Q_1 (S_x, respectively) is connected with Q_2 (S_y, respectively) via a common bond and the two substructures are rotatable with respect to the common bond. In general, it can be shown that $S_x{\cdot}EP_1' = S_y{\cdot}EP_1'$ and $S_x{\cdot}EP_2' = S_y{\cdot}EP_2'$ if and only if the two substructure matches are augmentable. To see this, notice that when two substructures are rotated around the common bond, the relative positions of all the atoms in one substructure with respect to the other substructure are changed except the two end atoms of the common bond. If the two substructure matches are augmentable, we can fix the two substructures of the target molecule Q (the data molecule O, respectively) to form a larger substructure K (S, respectively), thus obtaining a match between K and S. The atom_match_list of S is the union of the atom_match_list of S_x and the atom_match_list of S_y. The relabeling_counter of S is the sum of the relabeling_counter of S_x and the relabeling_counter of S_y.

To illustrate this augmentation process, consider again the example in section 3.2. The atom_match_list of the substructure S_0 includes the atoms numbered 1, 2, 3, 4, and 5 after hashing all atom triplets of Q_0. The relabeling_counter of S_0 is 0. The atom_match_list of the substructure S_1 includes the atoms numbered 6, 7, 8, 9, and 10 after hashing all atom triplets of Q_1. The relabeling_counter of S_1 is 1. There is a tuple (12, S_0, S_1, 0, 6, 5, 6) in the common bond table. Therefore we calculate

$$\vec{V}_{P_0,P_5} = (1.0097, 3.6478, 2.2660) - (1.0178, 1.0048, 2.5101) = (-0.0081, 2.6430, -0.2441)$$

$$S_0{\cdot}SF_Q = \begin{pmatrix} -0.012200 & 5.005500 & 4.474200 \\ 0.987800 & 5.005500 & 4.474200 \\ -0.012200 & 4.298393 & 3.767093 \end{pmatrix}$$

$$S_0{\cdot}E = \begin{pmatrix} 1.000000 & 0.000000 & 0.000000 \\ 0.000000 & -0.707107 & -0.707107 \\ 0.000000 & 0.707107 & -0.707107 \end{pmatrix}$$

$$S_0{\cdot}P_{c_1} = (-0.012200, 5.005500, 4.474200)$$

$$S_0{\cdot}EP_1' = \vec{V}_{P_0,P_5} \times S_0{\cdot}E + S_0{\cdot}P_{c_1} = (-0.020300, 2.964012, 2.777921)$$

Similarly,

$$S_0{\cdot}EP_2' = \vec{V}_{P_0,P_6} \times S_0{\cdot}E + S_0{\cdot}P_{c_1} = (0.102900, 2.316302, 2.220155)$$

$$S_1{\cdot}EP_1' = \vec{V}_{P_6,P_5} \times S_1{\cdot}E + S_1{\cdot}P_{c_1} = (-0.020300, 2.964012, 2.777921)$$

$$S_1{\cdot}EP_2' = \vec{V}_{P_6,P_6} \times S_1{\cdot}E + S_1{\cdot}P_{c_1} = (0.102900, 2.316302, 2.220155)$$

Since $S_0{\cdot}EP_1' = S_1{\cdot}EP_1'$ and $S_0{\cdot}EP_2' = S_1{\cdot}EP_2'$, the two substructure matches are augmentable. We fix Q_0 and Q_1 to form Q in Figure 5 and fix S_0 and S_1 to form O in Figure 1. The atom_match_list of O now includes atoms 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10, meaning that these atoms match atoms in Q geometrically. The relabeling_counter of O is 1, meaning that there is a relabeling operation (i.e., changing "e" to "c") when matching O with Q.

3.4. Query Processing Algorithms. By consulting the common bond table, one can augment small substructure matches to form larger substructure matches whenever appropriate. Then we can obtain the atom_match_list and relabeling_counter of the data molecule O. The size of atom_match_list of O shows the number of atoms in O that match atoms in Q geometrically. The relabeling_counter of O shows among those geometrically matching atoms how many need to be relabeled. Thus, the atom_match_list and relabeling_counter together show the distance between O and Q.

Formally, let n be the number of atoms in the atom_match_list, m be the value of relabeling_counter, and

Table 3. Algorithms for Processing the Six Types of Queries Described in Section 1.1

query type	data molecules returned
good-match	the molecules O where $\Delta(O,Q) \leq \epsilon$
k-closest	the k molecules O with the smallest $\Delta(O,Q)$'s
best-match	the molecule O with the smallest $\Delta(O,Q)$
bad-match	the molecules O where $\Delta(O,Q) > \epsilon$
k-farthest	the k molecules O with the largest $\Delta(O,Q)$'s
worst-match	the molecule O with the largest $\Delta(O,Q)$

$|Q|$ ($|O|$, respectively) be the size of the target molecule Q (the data molecule O, respectively). Observe that in matching O with Q there are $|O| - n$ atom deletes, $|Q| - n$ atom inserts, and m atom relabelings. Therefore the distance between O and Q, denoted $\Delta(O,Q)$, is

$$\Delta(O,Q) = m + |O| + |Q| - 2n$$

Referring to section 1.1, $\Delta(O,Q)$ represents the number of nonredundant edit operations needed to transform O to Q (or superimpose O on Q).

For example, consider again the target molecule Q in Figure 5 and the data molecule O in Figure 1. $|Q| = 10$ and $|O| = 11$. After the augmentation as explained in the example in section 3.3, the number of atoms in the atom_match_list of O is 10 and the relabeling_counter of O is 1. Thus the distance between O and Q is $\Delta(O,Q) = 1 + 11 + 10 - (2 \times 10) = 2$. Referring to Figure 1 and Figure 5, we see that in matching O with Q we delete one atom (i.e., delete the atom numbered 0 in O) and relabel another atom (i.e., change the name "e" of atom 9 in O to the name "c" of atom 8 in Q).

Thus, after hashing the target molecule Q, we check the atom_match_list and relabeling_counter for each molecule O in the database and calculate $\Delta(O,Q)$. Table 3 summarizes the algorithms for processing the six types of queries described in section 1.1.

4. EXPERIMENTS

We have implemented the proposed algorithms using the C programming language on a SunSPARC 20 workstation running Solaris version 2.4. Two files were maintained: one recording the bucket addresses and the other containing all hash table entries. We applied the algorithms to 226 3D molecular structures obtained from a database maintained in the National Cancer Institute. The number of atoms in the molecules ranged from 5 to 51. It took 9 s to hash all 226 molecules in the preprocessing phase. In order to demonstrate the advantage of the decomposition/augmentation processes, we studied two cases. In the first case, we hashed and retrieved a molecule in its entirety. In the second case, we decomposed the molecules to rigid substructures and augmented substructure matches during the retrieval as described in the paper. Figure 6 shows the results. The dashed line represents the retrieval time without the decomposition/augmentation processes. The solid line represents the retrieval time with the processes. It can be seen that the decomposition/augmentation processes speed up the retrieval by a factor of 100 when molecules have 30 atoms and 1000 when molecules have 50 atoms.

We then compared our technique with exhaustive search using more molecules taken from the NCI database. By

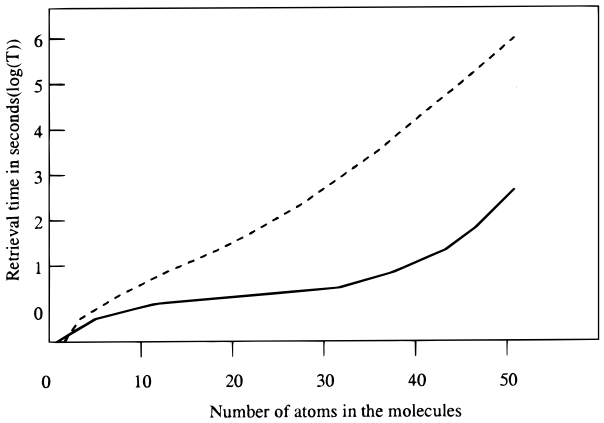

Figure 6. Retrieval times as a function of the size of molecules.

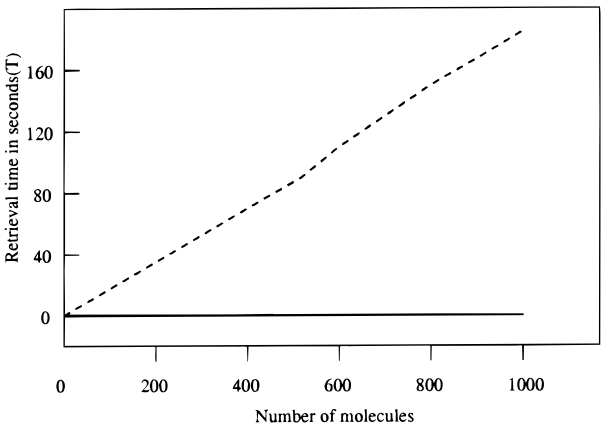

Figure 7. Retrieval times as a function of the number of molecules.

exhaustive search, we mean that in the preprocessing phase we sort and store the lengths of the three bonds of the triangle formed by every atom triplet $[i, j, k]$ in a three-dimensional array. We also store the local coordinate frame $LF[i,j,k]$. In the on-line phase, we find atom-triplet matches by searching the array. Figure 7 shows the results. The dashed line represents the exhaustive search method, and the solid line represents our technique. It can be seen that our technique is 100 times faster than the exhaustive search method when the database has over 600 molecules while achieving the same recall.

5. RELATED WORK

The geometric hashing algorithm used in the paper was originated from the work of Lamdan and Wolfson for model-based recognition in computer vision.[19] Several researchers attempted to parallelize the algorithm,[5,23,24] design delicate rehash functions to balance the distribution of hash function values,[24] and explore the uncertainty existing in the algorithm.[13,27] However, none of the work addressed the similarity search problem in structure databases. The work most closely related to ours, in the context of geometric hashing, is.[25,37] In ref 25, Rigoutsos et al. solved the substructure matching problem for 3D molecules; in ref 37, Wang et al. presented techniques for finding frequently occurring substructures in these molecules. In contrast to refs 25 and 37, we present here a framework for systematically answering a class of similarity-based queries. Furthermore, in contrast to ref 25, which employed "magic vectors" for substructure matching,

we store a coordinate frame in a hash table entry and adopt the decomposition/augmentation processes to speed up the search. There were papers about graph matching.[3,6,10,30,32,35,39,40] However, none of them considered the queries presented here using geometric hashing algorithms.

Similarity searching in 3D molecules has been studied extensively in the past.[1,2,4,7,18,20,33,34,42,43] Willett et al.[41] presented an excellent survey. Many similarity measures were defined based on 2D descriptors, 3D descriptors, or other descriptors of molecules. Filimonov et al.,[9] for example, proposed a descriptor based on multilevel neighborhoods of atoms to measure the similarity of two compounds. An atom's zero level descriptor includes only the atom itself. An atom's first level descriptor includes the atom and its neighbors' zero level descriptors. An atom's second level descriptor includes the atom and its neighbors' first level descriptors. This process of representation is repeated recursively until a desired number of levels is reached. A compound is then represented by the set of the descriptors of all its atoms. In calculating the similarity of two compounds, the occurrences of the descriptors and the frequencies of their occurrences in the two compounds are calculated and compared.

In ref 12, Ginn et al. proposed a descriptor derived solely from the vibrational frequencies of a molecule's infrared image. The authors then combined the descriptor with 2D fingerprints to measure the similarity of two compounds. Xue et al.[46] considered descriptors comprising the number of aromatic bonds and hydrogen-bonding acceptors, the fraction of rotatable bonds per molecule, and structural key-type fragments. The authors encoded the descriptors into binary bit strings and performed similarity searching by comparing those strings.

Flower[11] studied the effectiveness of bit string based similarity measures. He found that the performance of comparing binary bit strings is heavily dependent on the effectiveness of the features (e.g., descriptors) encoded in a binary string. Problems may arise if a feature is not a metric. A metric δ is a function where for any three feature values A, B, and C (i) $\delta(A,B) > 0$, if $A \neq B$; (ii) $\delta(A,A) = \delta(B,B) = 0$; (iii) $\delta(A,B) = \delta(B,A)$; and (iv) (triangular inequality) $\delta(A,B) \leq \delta(A,C) + \delta(C,B)$. As a nonexample, the similarity measure derived from bit strings based on hashed 2D structural fingerprints does not satisfy the triangular inequality.

Many molecules have more than one conformation due to torsional flexibility, which is often caused by rotations around rotatable single bonds. Given one conformation of a molecule, the problem of locating those different conformations is referred to as flexible searching. Some researchers applied genetic algorithms to approaching this problem. In refs 31 and 38, Willett and co-authors studied different types of genetic algorithms. Their algorithms worked by identifying a set of geometric transformations, including rotations, translations, and torsional rotations, that results in the maximal overlap of a database structure's molecular electrostatic potential with that of the target structure.

Handschuh et al.[15] also applied genetic algorithms to the superposition of 3D chemical structures. A compound can be superimposed on another compound if the corresponding atoms can be aligned together. The authors exploited the power of Pareto optimization and Tournament selection to improve the performance of their algorithms. In ref 8, Cramer et al. developed an approach to searching for chemical structures with similar topomer shapes. The authors concluded that topomer shape similarity searching can enhance the effectiveness of the overall drug discovery process. To compute the difference in shape between any two monovalent fragments, their algorithm first built a single characteristic "topomeric" conformation for each fragment by using rule-based adjustments of the appropriate torsion angles in the fragment and chiralities of its Concord-generated 3D structure. The algorithm then positioned two resulting conformations to superimpose the two attachment valences of the fragments.

In ref 36, Wang and Zhou combined 1D, 2D, and 3D searching in one toolkit. They first searched the database for those structures that had the same types and numbers of atoms as those in the target structure. The authors then used the generic match algorithm invented by Xu[45] to perform 2D screening. The third step conducted a 3D rigid search. In the rigid search, the relative positions of the atoms were fixed; i.e., no torsional flexibility was considered in the search. If the rigid search failed to identify enough qualified structures matching the target structure, a conformationally flexible search was activated.

In general, rotatable single bonds can be classified into two categories: (i) those that connect two rigid substructures[8] and (ii) those that are within one flexible substructure.[15] Torsion angle changes are caused by rotations around these rotatable single bonds. We consider in the paper the rotatable single bonds in the first category and approach the torsion angle change problem by decomposing a molecule into rigid substructures. This technique can be incorporated into previously published methods (see, e.g., Wang and Zhou[36]) for 3D rigid searching.

In cases where there are torsion angle changes caused by rotations around rotatable single bonds in the second category, using our searching algorithms with edit distance 0 will suffer from a low recall. In these cases, one has to conduct a similarity search by allowing a certain number of edit operations to exist when matching two molecules. In this way, a database molecule can be superimposed on the target molecule even though they do not agree in a small number of atoms, which is probably caused by some torsion angle changes. These atoms may have different names or may appear in different positions. The edit operations include relabeling an atom, inserting an atom, and deleting an atom (relabeling an atom can be considered as replacing the atom in its position by a different atom). Thus, for example, if a rotatable bound occurs in a flexible substructure, our algorithms can detect it in a similarity search with edit distance 1 or 2 (i.e., one or two atom inserts/deletes are allowed in the search); cf. the good-match query in section 1.1. This technique may be used to enhance the recall of previously published methods for flexible searching (see, e.g., Willett et al.[31,38]).

Many earlier techniques for superposition of 3D molecules attempted to minimize the root-mean-square error of the distances of corresponding atoms.[1,15,34,36,42] Clearly, these approaches do not apply to chemical structures with large torsion angle changes. The approximate matching technique proposed here can also be used, in combination with these

superposition methods, to improve the overall search performance.

6. CONCLUSIONS

In this paper we have presented a geometric hashing technique for similarity retrieval in three-dimensional structure databases. We applied the technique to processing a class of similarity-based queries. Our technique can also be extended to solve the substructure search problem.[14,16,44] Given the target molecule Q and a database D of molecules, the substructure search problem is to find the molecules O in D that approximately contain Q; i.e. there exists a subgraph O' of O such that O' approximately matches Q. Refer to section 3. Suppose the number of atoms in the atom_match_list associated with a data molecule O is n. The value of the relabeling_counter of O is m. The size of the target molecule Q is $|Q|$ and the size of the data molecule O is $|O|$; $|Q| \leq |O|$. Then we know that there exists a subgraph O' of O where O' matches Q with distance $|Q| - n + m$. Thus if the user is interested in finding molecules that approximately contain Q within distance ϵ, our programs return those data molecules O whose $|Q| - n + m$ is less than or equal to ϵ. In addition, our programs find and display the optimal alignment between O and Q.

We have made the software for processing the similarity-based queries and substructure search available on the Internet; please visit the Web site at http://www.cis.njit.edu/~discdb for details. Interested readers may also contact the authors directly to get the programs.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation under Grant IRI-9531548. We thank the anonymous reviewers for their constructive suggestions that improved both the quality and presentation of this paper. We also thank Dr. Isidore Rigoutsos (IBM), Dr. Bruce Shapiro (National Cancer Institute), and Professor Dennis Shasha (New York University) for their inspirations and very helpful discussions during the preparation of the paper.

REFERENCES AND NOTES

(1) Adamson, G. W.; Bush, J. A. A comparison of the performance of some similarity and dissimilarity measures in the automatic classification of chemical structures. *J. Chem. Inf. Comput. Sci.* **1975**, *15*, 55−58.

(2) Adamson, G. W.; Cowell, J.; Lynch, M. F.; McLure, A. H. W.; Town, W. G.; Yapp, A. M. Strategic considerations in the design of a screening system for substructure searches of chemical structure files. *J. Chem. Doc.* **1973**, *13*, 153−157.

(3) Ash, J. E.; Chubb, P. A.; Ward, S. E.; Welford, S. M.; Willett, P. *Communication, Storage and Retrieval of Chemical Information*; Ellis Harwood: Chichester, England, 1985.

(4) Basak, S. C.; Magnuson, V. R.; Niemi, G. J.; Regal, R. R. Determining structural similarity of chemicals using graph-theoretic indices. *Discrete Appl. Math.* **1988**, *19*, 17−44.

(5) Bourdon, O.; Medioni, G. Object recognition using geometric hashing on the connection machine. *Proceedings of the 10th International Conference on Pattern Recognition, 1990*; pp 596−600.

(6) Brown, R. D.; Jones, G.; Willett, P.; Glen, R. C. Matching two-dimensional chemical graphs using genetic algorithms. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 63−70.

(7) Carbo, R.; Arnau, M.; Leyda, L. How similar is a molecule to another? An electron density measure of similarity between two molecular structures. *Int. J. Quantum Chem.* **1980**, *17*, 1185−1189.

(8) Cramer, R. D.; Poss, M. A.; Hermsmeier, M. A.; Caulfield, T. J.; Kowala, M. C.; Valentine, M. T. Prospective identification of biologically active structures by topomer shape similarity searching. *J. Med. Chem.* **1999**, *42* (19), 3919−3933.

(9) Filimonov, D.; Poroikov, V.; Borodina, Y.; Gloriozova, T. Chemical similarity assessment through multilevel neighborhoods of atoms: Definition and comparison with the other descriptors. *J. Chem. Inf. Comput. Sci.* **1999**, *39* (4), 666−670.

(10) Fisanick, W.; Lipkus, A. H.; Rusinko, A. III. Similarity searching on CAS registry substances. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 130−140.

(11) Flower, D. R. On the properties of bit string-based measures of chemical similarity. *J. Chem. Inf. Comput. Sci.* **1998**, *38* (3), 379−386.

(12) Ginn, C. M. R.; Turner, D. B.; Willett, P.; Ferguson, A. M.; Heritage, T. W. Similarity searching in files of three-dimensional chemical structures: Evaluation of the EVA descriptor and combination of rankings using data fusion. *J. Chem. Inf. Comput. Sci.* **1997**, *37* (1), 23−37.

(13) Grimson, W. E. L.; Huttenlocher, D. P.; Jacobs, D. W. *Affine matching with bounded sensor error: Study of geometric hashing and alignment*; Technical Memo AIM-1250; Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1991.

(14) Hagadone, T. R. Molecular substructure similarity searching: Efficient retrieval in two-dimensional structure databases. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 515−521.

(15) Handschuh, S.; Wagener, M.; Gasteiger, J. Superposition of three-dimensional chemical structures allowing for conformational flexibility by a hybrid method. *J. Chem. Inf. Comput. Sci.* **1998**, *38* (2), 220−232.

(16) Hicks, M. G.; Jochum, C. Substructure search system. 1. Performance comparison of the MACCS, DARC, HTSS, CAS registry MVSSS, and S4 substructure search systems. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 191−199.

(17) Horowitz, E.; Sahni, S. *Fundamentals of Computer Algorithms;* Computer Science Press: Rockville, MD, 1978.

(18) Kearsley, S. K.; Sallamack, S.; Fluder, E. M.; Andose, J. D.; Mosley, R. T.; Sheridan, R. P. Chemical similarity using physiochemical property descriptors. *J. Chem. Inf. Comput. Sci.* **1996**, *36*, 118−127.

(19) Lamdan, Y.; Wolfson, H. Geometric hashing: A general and efficient model-based recognition scheme. *Proceedings of International Conference on Computer Vision, 1988*, pp 237−249.

(20) Lynch, M. F. The microstructure of chemical databases and the choice of representation for retrieval. In *Computer Representation and Manipulation of Chemical Information*; Wipke, W. T., et al., Eds.; Wiley: New York, 1974.

(21) McHugh, J. A. *Algorithmic Graph Theory;* Prentice Hall: Englewood Cliffs, NJ, 1990.

(22) Minsky, M.; Papert, S. *Perceptrons: An Introduction to Computational Geometry.* MIT Press: Cambridge, MA, 1969.

(23) Rigoutsos, I.; Hummel, R. *Scalable parallel geometric hashing for hypercube SIMD architectures*; Technical Report TR-553; Department of Computer Science, New York University, 1991.

(24) Rigoutsos, I.; Hummel, R. *On a parallel implementation of geometric hashing on the connection machine*; Technical Report TR-554; Department of Computer Science, New York University, 1991.

(25) Rigoutsos, I.; Platt, D.; Califano, A. *Flexible substructure matching in very large databases of 3D-molecular information*; Research Report; IBM T. J. Watson Research Center, 1996.

(26) Sankoff, D., Kruskal, J. B., Eds.; *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*; Addison-Wesley, Reading, MA, 1983.

(27) Sarachik, K. B. *Limitations of geometric hashing in the presence of Gaussian noise*; Technical Memo AIM-1395; Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1992.

(28) Shapiro, B. A.; Zhang, K. Comparing multiple RNA secondary structures using tree comparisons. *Comput. Appl. Biosci.* **1990**, *6* (4), 309−318.

(29) Shasha, D.; Wang, T. L. New techniques for best-match retrieval. *ACM Trans. Inf. Syst.* **1990**, *8* (2), 140−158.

(30) Takahashi, Y.; Sukekawa, M.; Sasaki, S-I. Automatic identification of molecular similarity using reduced-graph representation of chemical structure. *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 639−643.

(31) Thorner, D. A.; Wild, D. J.; Willett, P.; Wright, P. M. Similarity searching in files of three-dimensional chemical structures: Flexible field-based searching of molecular electrostatic potentials. *J. Chem. Inf. Comput. Sci.* **1996**, *36* (4), 900−908.

(32) Trinajstic, N.; Nikolic, S.; Knop, J. V.; Muller, W. R.; Szymanski, K. *Computational Chemical Graph Theory;* Ellis Harwood Limited: Chichester, England, 1991.

(33) Tversky, A. Features of similarity. *Psych. Rev.* **1977**, *84*, 327−352.

(34) Van Drie, J.; Weininger, D.; Martin, Y. ALADDIN: An integrated tool for computer-assisted molecular design and pharmacophore

recognition from geometric, steric and substructure searching of three-dimensional molecular structures. *J. Computer-Aided Mol. Des.* **1989**, *3*, 225−251.
(35) Wang, J. T. L., Shapiro, B. A., Shasha, D., Eds.; *Pattern Discovery in Biomolecular Data: Tools, Techniques and Applications*; Oxford University Press: New York, NY, 1999.
(36) Wang, T.; Zhou, J. 3DFS: A new 3D flexible searching system for use in drug design. *J. Chem. Inf. Comput. Sci.* **1998**, *38* (1), 71−77.
(37) Wang, X.; Wang, J. T. L.; Shasha, D.; Shapiro, B. A.; Dikshitulu, S.; Rigoutsos, I.; Zhang, K. Automated discovery of active motifs in three-dimensional molecules. *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, 1997*; pp 89−95.
(38) Wild, D. J.; Willett, P. Similarity searching in files of three-dimensional chemical structures. Alignment of molecular electrostatic potential fields with a genetic algorithm. *J. Chem. Inf. Comput. Sci.* **1996**, *36* (2), 159−167.
(39) Willett, P. *Similarity and Clustering Methods in Chemical Information Systems;* Research Studies Press: Letchworth, 1987.
(40) Willett, P. Algorithms for the calculation of similarity in chemical structure databases. In *Concepts and Applications of Molecular Similarity*; Johnson, M. A., Maggiora, G. M., Eds.; John Wiley & Sons, Inc.: New York, NY, 1990; pp 43−61.
(41) Willett, P.; Barnard, J. M.; Downs, G. M. Chemical similarity searching. *J. Chem. Inf. Comput. Sci.* **1998**, *38* (6), 983−996.
(42) Willett, P.; Winterman, V. A comparison of some measures for the determination of intermolecular structural similarity. *Quant. Struct.-Act. Relat.* **1986**, *5*, 18−25.
(43) Willett, P.; Winterman, V.; Bawden, D. Implementation of nearest-neighbor searching in an online chemical structure search system. *J. Chem. Inf. Comput. Sci.* **1986**, *26*, 36−41.
(44) Wipke, W. T.; Rogers, D. Rapid subgraph search using parallelism. *J. Chem. Inf. Comput. Sci.* **1984**, *24*, 255−262.
(45) Xu, J. GMA: A generic match algorithm for structural homomorphism, isomorphism, and maximal common substructure match and its applications. *J. Chem. Inf. Comput. Sci.* **1996**, *36* (1), 25−34.
(46) Xue, L.; Godden, J. W.; Bajorath, J. Database searching for compounds with similar biological activity using short binary bit string representations of molecules. *J. Chem. Inf. Comput. Sci.* **1999**, *39* (5), 881−886.
(47) Zhang, K.; Wang, J. T. L.; Shasha, D. On the editing distance between undirected acyclic graphs. *Int. J. Found. Comput. Sci.* **1996**, *7* (1), 43−57.

CI990081M