Geometric Methods for Protein Structure Comparison

Carlo Ferrari and Concettina Guerra

Department of Information Engineering, University of Padova,
Via Gradenigo 6a, 35131 Padova, Italy
tel: 039-049-8277821 fax: 039-049-8277826
{*carlo,guerra*}*@dei.unipd.it*

1 Introduction

Protein structural comparison is an important operation in molecular biology and bionformatics. It plays a central role in protein analysis and design. As proteins fold in three dimensional space, assuming a variety of shapes, a careful characterization of their geometry is needed to study their function which is known to be related to the shape. Moreover, the comparison of protein structures is essential to infer evolutionary information.

The problem of comparing three-dimensional structures has been widely studied in other disciplines such as computer vision and image processing, robotics, astronomy and some core methods have migrated from these disciplines to bionformatics.

There are many instances of the protein comparison problem that have been addressed; they include: 1) protein pairwise comparison, 2) protein classification, to organize all known structures in a biologically relevant groups, 3) searching for common folding patterns and three-dimensional motifs, 4) studying of protein interaction to identify binding sites for drug design.

From the application point of view, it is important to mention how the growth of the Protein Data Bank (PDB) asks for effective automatic procedures for classification and search of the database elements. Currently the PDB contains more than 17,000 structures and this number is rapidly growing.

The protein comparison may involve different levels of representations of the three dimensional protein structures, from the atomic level to the level of secondary structures. Most methods presented in the literature deal with a protein representation in terms of atomic coordinates

and therefore with a matching problem that uses as basic elements sets of points (atoms).

Other approaches are based on secondary structures, i.e. α helices and β strands, that play an important role in the functional behavior of a protein. The secondary structure elements are represented as vectors in 3D space.

An alignment of α helices and β strands may be used for fast retrieval of folds or motifs from the PDB. On the other hand, the comparison of secondary structures can be used as the first step in a two step comparison procedure that first identifies possible candidate solutions in a fast way and then refines the solutions taking into consideration the atomic descriptions of proteins. Another advantage of a structural comparison of secondary structures is that it allows to study the folding process by tracing the evolution of the fold from the molden state.

Most of existing approaches allow to detect global similarity between entire proteins as well as local similarity ([1], [20], [21], [23], [26], [27], [41], [46], [51]).

The integration of strategies operating at different levels of representations appears very promising to achieve robustness and efficiency. Extensive surveys on the subject of protein comparison exist enphasizing different aspects of the general problem [6], [37], [50].

In this paper, we review some of the theoretical results on the computational complexity of the algorithms designed to obtain optimal solutions to the problem of matching sets of points using specific metrics. From a theoretical point of view, the problem has been extensively studied in the area of computational geometry, where it is often formulated as the problem of finding correspondences between sets of geometric features (for instance, points or segments). From these studies it appears that, in most practical cases, exact algorithms are too time consuming to be useful. Thus, approximate algorithms are considered that are computationally practical and at the same time are guaranteed to produce solutions that are within a certain bound from optimal.

Furthermore, we discuss methods for the estimation of rigid transformations under different metrics such as the Root Mean Square Deviation (RMSD) and the Hausdorff distance. Geometric indexing techniques prove their effectiveness in searching large protein databases and they are presented in details. Finally graph-theoretic protein modeling is reviewed as it is useful in designing algorithms for substructure identification and comparison.

Throughout the paper, we will use pure geometric information, ignoring other properties associated with atoms. Chemical properties, such as hydrophobicity, charge, etc. may be important in protein comparison and often they can be easily incorporated in a matching procedure. The use of such properties may help reduce the computation time by allowing pruning of the possible associations at early stages of the processing. However, we will not be consider these other properties in this chapter. Applications of protein comparison are an important subject; since they discussed in the two previous chapters of this volume, they are not considered here.

2 Protein Description

A protein is a sequence of aminoacids linked by peptide bonds. An aminoacid consists of a carbon atom C_α to which are attached a hydrogen atom, an amino group, and a carboxyl group. The 20 aminoacids differ in the side chain or *residue* attached to the C_α atom. The peptide bonds between the aminoacids in the chain join the carboxyl group of one aminoacid with the amino group of the next eliminating water in the linking process. The sequence of aminoacids is generally referred to as the *primary structure* of a protein. Its length varies from a few tens to few thousands aminoacids. A different level of protein representation, known as *secondary structure*, describes a protein in terms of recurrent regular substructures, such as the α *helices* and the β *strands*. The *tertiary structure* is the packing of the structural elements into the 3D shape. The protein may contain several chains forming its *quaternary structure*. For a survey of the protein architecture see [5], [38].

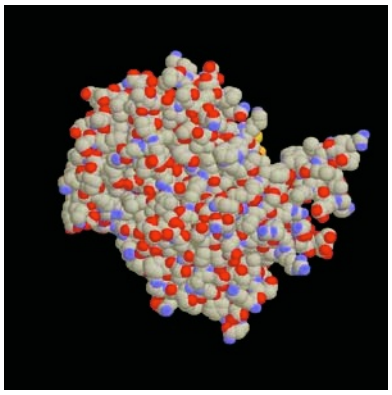

Fig. 1. The volumetric representation of protein 1rpa

The volumetric representation of a protein is displayed in figure 1 where all atoms are shown as balls; the secondary structure elements of the same protein are displayed as ribbons in figure 2.

Arrangements of the secondary structures α helices and β strands are the basis for the protein structural classification of SCOP [44]. In the SCOP classification hierarchy, the fold level corresponds to the last level of the hierarchy, the other two being family and superfamily. Proteins sharing a fold have the same major secondary structures but do not necessarily have a common evolutionary relationship, unlike proteins clustered into families and superfamilies. The similarity in the arrangements of secondary structures in a fold may be due to the physical and chemical properties of the packing of the proteins.

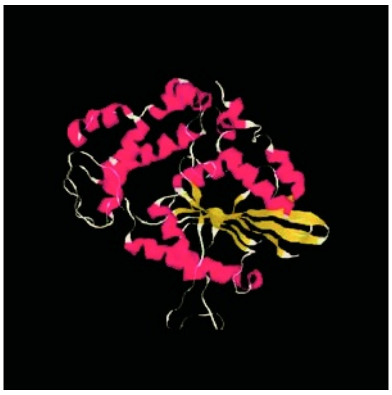

Fig. 2. The secondary structure representation of protein 1rpa

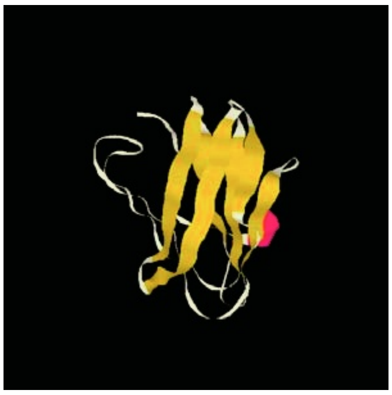

Fig. 3. The ribbon representation of protein 1bxa showing a β sandwich

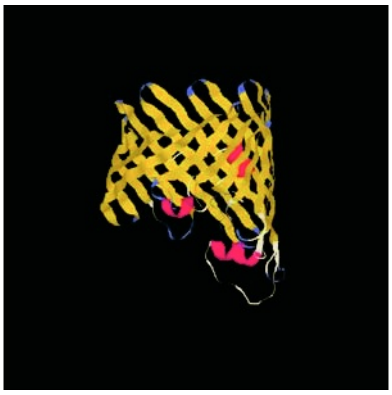

Fig. 4. The ribbon representation of protein 3por forming a β barrel

Common structural arrangements of secondary structures or *motifs* have been identified within the folds and they include, among others, the *β-sandwich* and the *β-barrel*, as seen in figure 3 and 4, respectively.

Approaches to protein comparison use different protein structural descriptions. A complete structural description is given by the 3D coordinates (x, y, z) of the individual atoms of a protein. Often only the C_α atoms of the aminoacids, that form the so-called *backbone* of a protein, are considered for comparison.

A more compact description is in terms of the linear vectors associated to the structural elements helices and β strands. While most of the comparison approaches are based on the atomic description of a protein, the secondary structure description may provide a fast method to retrieve substructures or motifs from large protein databases. Furthermore, it is often used as a first step when searching in a database for the most similar protein with respect to a target protein. In fact, hypotheses of similarity for the target protein are generated in a fast and efficient way based on secondary structures only; such hypotheses are further verified by a more refined and costly process that is only applied to those hypothesized proteins. This two-step procedure may considerably speed up the protein comparison when large databases are involved.

For most proteins in the PDB, secondary structures are annotated by the original depositor who provides the starting and ending residue numbers of all secondary structures. However programs have been designed for the assignment of the secondary structures from the PDB files and for the analysis of the overall and residue-by-residue geometry of a protein [10], [34], [35].

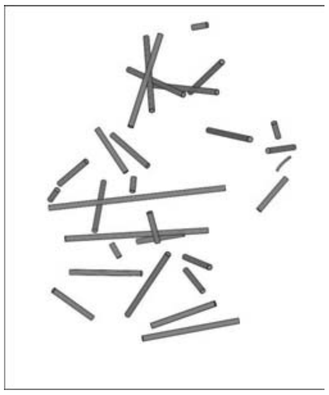

Fig. 5. The vectorial representation of protein kinase CK2

Several programs have also been developed to yield the vectorial representation of a protein [17], [41]. Singular-value decomposition (SVD) is a standard routine [4], [19] to find the axes of α helices and the best fit segments for the β strands. In this routine typically only C_α atoms are used. Other simpler methods derive the vector associated to a β strand either by directly connecting the starting and ending residues of the β strand assignments or by connecting two points that are computed as the average points of few of the extreme residues on both sides of the strand [51]. This second approach is less sensitive to curved or kinked structures. Figure 5 shows the vectorial representation of protein kinase CK2, where each segment is displayed as a cylinder of fixed radius.

3 Structural Comparison: Problem Formulation

The general matching problem can be informally defined as follows: Given two sets of geometric features, either points or line segments, determine the largest common subsets, i.e. the subsets of maximum size, that are geometrically similar. In the case of proteins, points correspond to atoms and segments are the axes of the secondary structures.

There are many variants of the matching problem that have been considered in many different contexts. We now give more formal definitions of the problem with varying degrees of computational complexity.

Problem 1. Consider two sets of geometric elements $A = \{a_1, a_2, \cdots, a_n\}$ and $B = \{b_1, b_2, \cdots, b_m\}$ in three-dimensional space and assume that they have the same cardinality, i.e. $n = m$, and that the element a_i corresponds to the element b_i. Find the transformation g between the two sets that minimizes a given distance metric D over all rigid body transformations T , i.e.

$$min_T D(T(A), B)$$

Problem 2. Consider two sets of geometric elements $A = \{a_1, a_2, \cdots, a_n\}$ and $B = \{b_1, b_2, \cdots, b_m\}$ in three-dimensional space. Find the transformation g between the two sets that minimizes a given distance metric D over all rigid body transformations T , i.e.

$$min_T D(T(A), B)$$

This problem differs from the previous one because no correspondence is known a priori between the elements of the two sets.

Problem 3. Given two sets $A = \{a_1, a_2, \cdots, a_n\}$ and $B = \{b_1, b_2, \cdots, b_m\}$ in three-dimensional space and a real number $\delta > 0$, find a maximum-cardinality set of pairs of elements, one element in A and the second one in B, such that the distance d between each pair of elements is at most δ.
In this problem we are interested in the largest subset of corresponding elements of A and B (generally, in practical applications the sets are required to be above a certain size).

Different metrics have been used in the literature to determine the structure similarity between geometric objects. The most common metric is the RMSD (Root Mean Square Deviation) defined for point sets as follows:

$$D(A,B) = RMSD(A,B) = (\sum_{i=1,n} d(a_i,b_i)^2)^{1/2}$$

where d is the Euclidean distance between two points, and assuming that the sets have the same cardinality n, a_i corresponds to b_i.

The RMSD distance is useful when comparing very similar structures, as those produced during the christallographic analysis or NMR at different stages of the process. Its disadvantage becomes apparent in the presence of *outliers* when the proteins are not structurally close. The existence of even few outliers may significantly alter the value of the distance and therefore the determination of the optimal superposition of the two structures.

A second important definition between two point sets is based on the use of contact maps [33]. The chapter by G. Lancia and Sorin Istrail in this book deals extensively with contact maps and they are not further discussed here.

Another definition is the Hausdorff metric widely used in the area of computer vision and image processing, in astronomy and extensively studied in the field of computational geometry. The Hausdorff distance $H(A,B)$ between A and B is:

$$H(A,B) = \max(h(A,B), h(B,A)))$$

where $h(A,B)$ is the *one-way Hausdorff distance from A to B* given by:

$$h(A,B) = \max_{a_i \in A} \left(\min_{b_j \in B} d(a_i,b_j) \right)$$

In the following we discuss different approaches to solve the above three problems with different metrics. Problem 1 and its solutions are presented in section 5. Problem 2 with the Hausdorff distance as metric is considered in section 6.Problems 3 is reviewed in sections 8.

4 Representation of Rigid Transformations

A large number of methods have been proposed in the literature to compute the rigid body transformation between two sets of 3D points. They differ with respect to the transformation representation, and the minimization procedure. A survey by Sabata and Aggarwal [49] lists several representation of transformations and approaches to solve this problem using both closed form solutions and iterative solutions. The book [16] gives a clear description of many issues related to rigid transformations with enphasis on visualization aspects.

Here we describe several representations of rigid transformations while the next section is devoted to review methods to compute them.

A rigid motion of an object is a motion that preserves the distances between object points. The net movement of a rigid body from one configuration to another configuration (via a rigid motion) is called a *rigid displacement.* A rigid transformation applied to an object model represents a rigid displacement of the object itself. Rigid transformations form the theoretical infrastructure both for studying actual objects motions and for predicting possible (or hypothetical) motions. In the early 1800s Chasles and Poinsot proved that every rigid body displacement can be realized by a rotation about an axis combined with a translation parallel to that axis. This motion is what it is usually referred as a *screw motion.* Different motion representations are presently used (mainly in computer vision, computer graphics and robotics), that can be roughly classified as local or global representations [43].

Let us start by formally introducing the definition of *rigid body transformation.* A rigid body transformation is a mapping $g : \mathbb{R}^3 \rightarrow \mathbb{R}^3$, that must satisfy the properties:

$\parallel g(q) - g(p) \parallel = \parallel q - p \parallel$ for all points $p, q \in \mathbb{R}^3$
$g(v \times w) = g(v) \times g(w)$ for all vectors $v, w \in \mathbb{R}^3$

The former condition says that lengths are preserved and the latter condition says that internal reflection is not allowed. As a consequence of the above definitions, rigid body transformations also preserve the inner product, in particular, orthogonal vectors are transformed to orthogonal vectors. In general, a rigid body transformation takes right-handed orthonormal coordinate frames to right-handed orthonormal coordinate frames. It is important to point out that even if the distance between points and the cross product between vectors are fixed, particles in a rigid body can move related to each others, because they can rotate (but not translate) with respect to each other. Then the motion of a body can be described by the motion of any one point and the rotation of the body around this point. Hence a right-handed Cartesian coordinate frame can be attached to some point of the body and the motion of individual points can be traced from the motion of the body frame and the motion of the frame attachment point. Due to its importance we first consider pure rotational motion.

Pure rotational motion in $\mathbb{R}^3$ can usually be described by a proper 3x3 matrix, that can be defined by stacking next to each other, the coordinates of the principal axes of a coordinate frame B (the body frame) relative to a coordinate frame A (the inertial frame). Such a matrix is called a *rotational matrix*: its columns are mutually orthonormal and its determinant is +1. The set of all 3x3 matrices that satisfy these two conditions is denoted by $SO(3)$, where *SO* stands for *Special Orthogonal.* $SO(3)$ is a *group* under the matrix multiplications, with the identity matrix I as the identity element. $SO(3)$ is the *rotation group* of $\mathbb{R}^3$. A 3x3 rotation matrix can be seen as

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

with nine different parameters, the orthonormality conditions (for the matrix columns) add three more constraints, while the sign of the cross product adds other three constraints. These six constraint equations, reduce the degree of freedom to three, that is, only three parameters are needed to completely represent a pure rotation in $\mathbb{R}^3$. The matrix coefficients r_{ij} can be expressed in term of these three parameters.

A rotation matrix $R \in SO(3)$ represents a rigid body transformation. In fact it can be proved that it preserves distances and orientations, that is:

- $\| Rq - Rp \| = \| q - p \|$ for all points $p, q \in \mathbb{R}^3$
- $R(v \times w) = Rv \times Rw$ for all $v, w \in \mathbb{R}^3$

Moreover a rotation matrix can be seen as an operator that takes the coordinates of a point (or vector) from a frame to another. Let p_b the coordinate of a point P with respect to the frame B, and R_{ab} the rotation matrix, the coordinates of P with respect to the frame A are given as:

$$p_a = R_{ab} p_b$$

The pure rotation operator is a linear operator (with the additional constraint that it is orthonormal). A sequence of two (or more) rotations will result in a single combined rotation and conversely a given rotation can be decomposed using two or more rotations. Rotation matrices can be combined to form new rotation matrices using matrix multiplication. If a frame C has orientation R_{bc} relative to frame B and B has orientation R_{ab} from frame A, then the orientation of C with respect to A is given by:

$$R_{ac} = R_{ab} R_{bc}$$

In particular, as we could expect, $R_{ab} R_{ba} = I$ and $R_{ba} = R_{ab}^{-1} = R_{ab}^{T}$.

An important result about rotations is the Euler Theorem that establishes that any rotation $R \in SO(3)$ is equivalent to a rotation about a given axis $\omega \in \mathbb{R}^3$ ($\| \omega \| = 1$), by an angle $\theta \in [0, 2\pi)$. In fact, it is possible to represent the motion of a single point p rotating about ω at a constant unit velocity, with the following differential equation:

$$\dot{p}(t) = \omega \times p(t) = \hat{\omega} p(t)$$

Solving this equation gives the expression for a single rotation about ω by θ, that is $R(\omega, \theta) = e^{\hat{\omega}\theta}$. The matrix $\hat{\omega}$ is defined as follows:

$$\begin{bmatrix} 0 & \omega_3 & -\omega_2 \\ -\omega_3 & 0 & \omega_1 \\ \omega_2 & -\omega_1 & 0 \end{bmatrix}$$

where $\omega^T = [\omega_1, \omega_2, \omega_3]$ and it has the property that $\hat{\omega}^T = -\hat{\omega}$. Such a matrix is a skew-symmetric matrix. If $\| \omega \| = 1$, $\hat{\omega}$ is a unit skew-symmetric matrix. It can be proved that the exponential $e^{\hat{\omega}\theta}$ can be rewritten in term of the skew-symmetric matrix, resulting in the so-called *Rodriguez's formula*:

$$e^{\hat{\omega}\theta} = I + \hat{\omega}\sin\theta + \hat{\omega}^2(1-\cos\theta)$$

This method represents rotations through the *equivalent axis representation* . This is perhaps the most intuitive way of representing rotations. However, it has some disadvantages. The representation is not unique: in fact choosing $\omega^{'} = -\omega$ and $\theta^{'} = 2\pi - \theta$ gives the same rotation as ω and θ, being the exponential map many-to-one. Moreover, singularities occur when $\theta = 0$ and the exponential equals I, in such a case ω cannot be determined. These singularities create problems when computing the rotations. Finally the transformations resulting from the composition of multiple rotations cannot be easily computed.

Besides the canonical coordinates, there exist different coordinate systems for representing the rotation group, mainly used in robotics systems. The following method of describing the orientation of the coordinate frame B relative to A uses the *Euler angles.* At the beginning frames A and B are coincident. First, the frame B is rotated about the z-axis by an angle α, then it is rotated about the (new) y-axis by an angle β and finally B is rotated about the z-axis by an angle γ. The triple of angles (α, β, γ) represents the overall rotation, and the angles α, β, γ are called the *ZYZ Euler angles.*

These three rotations occurs at principal axes and the global rotation matrix can be computed from the three rotation matrices related to the three elementary rotations, that is, giving the specific values for α, β, γ it is easy to compute both R_{ab} and R_{ba}. The converse question of whether the map from SO(3) to α, β, γ is surjective is important. It can be proved that for any $R \in SO(3)$ it is possible to determine the Euler angles. This representation suffers from the problem of singularity at R = I.

In order to solve this problem new methods should be studied. Rotations in a 2D space can be represented by complex numbers on the unit circle. When moving to a 3D space, it is possible to generalize this idea, by introducing *quaternions* . Formally a quaternion Q is a 4-tuple of the form $< q_0, q_1, q_2, q_3 >$: where q_0 is the *scalar* component of Q and $\overrightarrow{q} = (q_1, q_2, q_3)$ is the *vector* component of Q. Hence, $Q = (q_0, \overrightarrow{q})$ with $q_0 \in \mathbb{R}$ and $\overrightarrow{q} \in \mathbb{R}^3$. The set of quaternions is a 4D vector space over the reals and it forms a group with respect to quaternion multiplication (denoted "·"). Quaternions multiplication is defined as follows:

$$Q \cdot P = (q_0 p_0 - \overrightarrow{q} \cdot \overrightarrow{p},\ q_0 \overrightarrow{p} + p_0 \overrightarrow{q} + \overrightarrow{q} \times \overrightarrow{p})$$

The *unit* quaternions are the subset of all quaternions Q such that $\parallel Q \parallel = 1, where \parallel Q \parallel^2 = q_0^2 + q_1^2 + q_2^2 + q_3^2$

Each rotation matrix $R = e^{\hat{\omega}\theta}$ correspond to a unit quaternion defined as $Q = (\cos(\theta/2), \omega \sin(\theta/2))$. It can be proved that if Q_{ab} correspond to a rotation of frame A to B and Q_{bc} correspond to a rotation of frame B to C, then the rotation between frame A to C is given by the quaternion $Q_{ac} = Q_{ab} \cdot Q_{bc}$. An alternative representation of rotations, often used in computer vision, is the *unit quaternion*, Given a unit quaternion $Q = (q_0, \overrightarrow{q})$ the corresponding rotation is given by $\theta = 2\cos^{-1}(q_0)$ and $\omega = k\overrightarrow{q}$ with $k = 1/\sin(\theta/2)$, if $\theta \neq 0, \omega = 0$ otherwise.

The 3x3 rotation matrix R in terms of the unit quaternion is directly given by:

$$R = \begin{bmatrix} {q_0}^2{q_1}^2 - {q_2}^2 - {q_3}^2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Quaternions give a global parametrization of SO(3), at the cost of using four numbers instead of three to represent a rotation. Since the quaternions space has a group structure that directly corresponds to that of rotations, they provide an efficient representation without suffering from singularities.

Rigid body displacements usually are not limited to pure rotations (even if pure rotations represent an important subset), but they generally consist of rotations and translations. Pure translations have a very simple representation: given two co-oriented frames A and B, with p_{ab} the representation in A of the origin of B, for any point $q \in \mathbb{R}^3$, $q_a = p_{ab} + q_b$. Pure translations can be represented by 3D vectors.

A rigid body motion (a rigid body displacement) then can be represented by $p_{ab} \in \mathbb{R}^3$ and $R \in SO(3)$. The Cartesian product of $\mathbb{R}^3$ with SO(3), represents all the rigid body motions and it is denoted as SE(3) (that stands for *special Euclidean group*):

$$SE(3) = \{(p, R) : p \in \mathbb{R}^3, R \in SO(3)\}$$

Each element of SE(3) serves both as a specification of a rigid body placement (with respect to a fixed environment frame) and as a transformation taking the coordinates of a point from one frame to another. If a is a 3D point and a' is its corresponding point after a rigid transformation is applied, then the following relation holds:

$$a' = R_{ab}a + p_{ab}$$

The above can be expanded into:

$$\begin{bmatrix} a'_x \\ a'_y \\ a'_z \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} + \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

The transformation of points can be usefully represented using the *homogeneous representation* . The homogeneous representation maps points and vectors in a 4D space, by adding a forth coordinate. The homogeneous coordinates of a point $q = (q_1, q_2, q_3)$ are $\bar{q} = (q_1, q_2, q_3, 1)$, while the homogeneous coordinates of a vector $v = (v_1, v_2, v_3)$ are $\bar{v} = (v_1, v_2, v_3, 0)$. Then a rigid transformation becomes:

$$\bar{a'} = \begin{bmatrix} a' \\ 1 \end{bmatrix} = \begin{bmatrix} R_{ab} & p_{ab} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ 1 \end{bmatrix} = \overline{g}_{ab}\bar{a}$$

The 4×4 matrix '$\overline{g}_{ab}$ is called homogeneous representation of $g = (p_{ab}, R_{ab}) \in SE(3)$. Within homogeneous representation we obtain a linear representation of rigid body motions, and the standard matrix multiplication is the composition rule for rigid body motion.

Homogeneous coordinates are useful for representing *twists* . A twist is a couple $(v, \hat{\omega})$, with $v \in \mathbb{R}^3$ and $\hat{\omega}$ a skew-symmetric matrix. A twist can be written as:

$$\hat{\xi} = \begin{bmatrix} \hat{\omega} \; v \\ 0 \; 0 \end{bmatrix}$$

Of course $\hat{\xi} \in \mathbb{R}^{4x4}$, while the twist coordinates $\xi = (v, \omega) \in \mathbb{R}^6$. It can be proved that the exponential of a twist multiplied by a scalar, is an element of SE(3), that is, it represents a rigid transformation. Conversely it can be proved that every rigid transformation can be written as the exponential of some twist, that is, given a $g \in SE(3)$, there exists a twist $\hat{\xi}$ and $\theta \in \mathbb{R}$ such that $g = \exp(\hat{\xi}\theta)$. It is important to remind the reader that the exponential map is many-to-one and hence the choice of ω and θ may not be unique for solving the rotational component of the motion.

The concept of twist helps us to show that every motion is a *screw motion.* A screw motion is a motion which consists of a rotation about an axis in space by an angle of θ, followed by a translation along the same axis by an amount d. This motion is called a screw motion since it remind us to the actual motion of a screw that rotate and translates about the same axis. A screw consists of an axis l, a pitch h and a magnitude M, while a screw motion represents rotation by an amount $\theta = M$, about the axis l followed by a translation parallel to the axis l by an amount $h\theta$. The overall rigid displacement can be computed and it results as:

$$g = \begin{bmatrix} e^{\hat{\omega}\theta} \; (I - e^{\hat{\omega}\theta})q + h\theta\omega \\ 0 \; 1 \end{bmatrix}$$

If we choose $v = -\omega \times q + h\omega$, the twist coordinates $\xi = (v, \omega)$ generate the given screw motion. Going one step further it can be proved that it is possible to define a screw associated with every twist. Finally it is possible to conclude that every rigid body motion can be realized by a rotation about an axis combined with a translation parallel to that axis.

5 Determination of 3D Rigid Transformations

In this section we review solutions for problem 1, as formulated in section 3. Consider two sets of points $A = \{a_1, a_2, \cdots, a_n\}$ and $B = \{b_1, b_2, \cdots, b_m\}$ in 3D space with the same cardinality. Assume that point correspondences are known, i.e. that point a_i corresponds to the point b_i. The problem is to derive the rigid body transformation that optimally maps A into B. This problem is known as the *absolute orientation problem.* Points measurements are affected by some noise and errors, due both to the estimation of the point coordinates and to the determination of the point correspondences. Formally speaking this problem can be stated as a minimization problem, according to the least square error criterion.

In the literature there exist a large numbers of algorithms that compute 3-D rigid transformation between two sets of geometrical features. For our purposes, it is interesting to review some of the most popular closed-form solution using correspondent points [39]. The rotational and translational components are computed as solutions to a least square formulation of the problem. The proposed approaches differ mainly in the transformation representation. The first was developed by Arun, Huang and Blostein [4] and it is based on computing the singular value decomposition (SVD) of a matrix.

The problem has been formalized as a particular version of the well-studied *orthogonal Procrustes problem* , that can be stated as it follows:

$$\text{minimize} \parallel A - BR \parallel \text{ with respect to R, subject to } R^T R = I$$

where $A, B \in \mathbb{R}^{m \times 3}$ are given by the set of 3-D vectors a_i, and the set of 3-D vectors b_i, respectively, and $R \in \mathbb{R}^{3 \times 3}$, is orthogonal.

This problem is equivalent to the following one:

$$\text{maximize } trace(R^T B^T A) \text{ with respect to R, subject to } R^T R = I$$

In fact

$$(A - BR)(A - BR)^T = trace(A^T A) + trace(B^T B) - 2trace(R^T B^T A)$$

This problem can be approached using “The Singular Value Decomposition (SVD) Theorem”, that can be stated as follows: **Theorem.** *If A is a real m-by-n matrix then there exist two orthogonal matrices* $U \in \mathbb{R}^{m \times m}$ *and* $V \in \mathbb{R}^{n \times n}$ *such that* $U^T AV = diag(\sigma_1, \ldots, \sigma_p), p = min\{m, n\}$ *where* $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$. The problem of maximizing $trace(R^T B^T A)$ can be solved through the computation of the SVD of $B^T A$. In fact let Σ be the SVD of $B^T A$, that is

$$\Sigma = U^T (B^T A) V = diag(\sigma_1, \sigma_2, \sigma_3)$$

and we define a new orthogonal matrix $Z = V^T R^T U$. Then we obtain: $trace(R^T B^T A) = trace(R^T U U^T (B^T A) V V^T = trace(R^T U \Sigma V^T)$ then

$$trace(R^T U \Sigma V^T) = trace(V^T R^T U \Sigma V^T V = trace(Z\Sigma)$$

The previous expression can be rewritten as:

$$trace(R^T U \Sigma V^T) = trace(Z\Sigma) = \sum_{i=1}^{3} z_{ii} \sigma_i$$

as Z is orthogonal the best choice for R is obtained when Z becomes the identity matrix. Hence $R = UV^T$

This general method proves itself useful for computing 3-D rigid transformation estimation between two sets of corresponding points. The algorithm can be sketched as a two step algorithm. The first step computes the optimal rotation matrix R using the 3x3 correlation matrix $H = \sum_{i=1}^{N} a_i b_i^T$ through its singular value decomposition ($H = Udiag(\sigma_1, \sigma_2, \sigma_3)V^T$) obtaining $R = VU^T$. The second step computes the optimal translation vector as $P = a - Rb$.

A similar approach computes the eigenvalues of a proper derived matrix (the *orthonormal matrices*) instead. It was proposed by Horn, Hilden and Negahdaripour [25]. With this method the correlation matrix H is firstly computed. However, rather than computing its SVD, a *polar decomposition* is used, such that $H = RS$, where $S = (HH^T)^{1/2}$. The optimal rotation is given by:

$$R = H^T \left(\frac{1}{\sqrt{\lambda_1}} u_1 u_1^T + \frac{1}{\sqrt{\lambda_2}} u_2 u_2^T + \frac{1}{\sqrt{\lambda_3}} u_3 u_3^T \right)$$

where $\{\lambda_i\}$ and $\{u_i\}$ are the eigenvalus and eigenvectors of the matrix HH^T.

Representing rotations using unit and dual quaternions gives two more techniques that have been proposed respectively by Horn [28] and by Walker, Shao and Voltz [56]. The former method asks to rewrite the minimization problem in the quaternion framework. A new 4x4 matrix can be constructed from the correlation matrix H as:

$$K = \begin{bmatrix} H_{00} + H_{11} + H_{22} & H_{12} - H_{21} & H_{20} - H_{02} & H_{01} - H_{10} \\ H_{12} - H_{21} & H_{00} - H_{11} - H_{22} & H_{01} + H_{10} & H_{20} + H_{02} \\ H_{20} - H_{02} & H_{01} + H_{10} & H_{11} - H_{00} - H_{22} & H_{12} + H_{21} \\ H_{01} - H_{10} & H_{20} + H_{02} & H_{12} + H_{21} & H_{22} - H_{11} - H_{00} \end{bmatrix}$$

The optimal rotation is the eigenvector related to the largest positive eigenvalue of K.

The latter method is the most significantly different of the four. It was designed ot minimize the equation;

$$\Sigma^2 = \sum_{i=1}^{L} \alpha_i \parallel n_1 i - R n_2 i \parallel^2 + \sum_{i=1}^{N} \beta_i \parallel a_i - R b_i - P \parallel^2$$

where $\{n_{1i}\}$ and $\{n_{2i}\}$ are two sets of corresponding unit normal vectors, and $\{\alpha_i\}$, $\{\beta\}$ are weighting factors reflecting data reliability. Dual quaternions for representing both rotation and translation are used and again the minimization problem can be rewritten in this new framework, resulting in new equations involving the parametrization of the dual quaternions. Again optimal values for R and P can be computed.

These four algorithms can be compared with respect to their accuracy, stability and efficiency [40]. Experimentations shows that no one algorithm is superior in all case. In fact difference in accuracy (on nondegenerate 3-D point sets) is almost insignificant. Stability is more discriminant instead. The SVD and the unit quaternion method are very similar and usually the most stable. In terms of efficiency, the orthonormal matrix looks quicker with small data sets, while the dual quaternions method is superior with larger data sets (according to proper computer memory configuration). In conclusion, the SVD should provide the best stability and accuracy, even if is not as efficient as dual quaternions with large data set. Otherwise the unit quaternions can be chosen on smaller data set for slightly better speed performance.

6 Geometric Pattern Matching

In this section we focus on the Hausdorff distance, defined in section 3, and review both theoretical and practical approaches to compute it.

Much work has been done on the computation of the Hausdorff distance. In the area of computational geometry exact algorithms have been studied for the problem of deciding whether there exist a transformation that maps one set of points into another set within a given distance. Fundamental robustness issues are discussed in [3]. Chew et al. [10] have considered the problem of matching point-sets in a d-dimensional space using the Hausdorff distance under translation only. For the case $d = 3$, they provide exact solutions in $O(n^3 \log^2 n)$ time. Extensions [11] to the more general case of Euclidean motion and of sets of segments have obtained exact solutions in $O(n^6 \log^2 n)$ time. However they are limited to the case of planar sets.

Exact algorithms cannot be used in most practical applications where measurement errors and noise are present; furthermore, the high computational complexity of the exact algorithms make them impractical for use in real problems. For these reasons, approximate solutions for the case of point sets, both in 2-dimensional and in 3-dimensional space, have been considered [15].

In the field of computer vision, an efficient multi-resolution technique for comparing images using the Hausdorff distance has been presented in [29] where the space of possible transformations is limited to translations and scaling; in [48] the above technique is extended to affine transformations. Affine transformations are used in [24] for matching point sets. Other approaches to matching sets of segments in 3D space based on various techniques and metrics are given in [9], [24], [31]

The computation of the Hausdorff distance does not necessarily produce a one-to-one correspondence between the elements of the two sets; it may happen, in fact, that multiple elements in one set are associated with a single element of the other set. This is unlike most existing object recognition methods that give an explicit pairing.

Approaches have been proposed for the computation of the Hausdorff distance between sets of segments associated to secondary structures [10]. The standard Hausdorff distance provides a good metric over point sets but does not preserve the notion of relevant subsets like the segments. To keep information relative to the line segments in the definition of the distance function an alternative definition of the Hausdorff metric between sets of segments has been introduced in [23], together with efficient approximate algorithms for its computation Given two sets $A = \{a_1, a_2, \cdots, a_m\}$ and $B = \{b_1, b_2, \cdots, b_n\}$ of line segments a_i and b_j, the Segment Hausdorff distance $H_S(A, B)$ between A and B is:

$$H_S(A, B) = \max(h_S(A, B), h_S(B, A))),$$

where $h_S(A, B)$ is the one-way Segment Hausdorff distance given by:

$$h_S(A, B) = \max_{a_i \in A} \left(\min_{b_j \in B} H(\{a_i\}, \{b_j\}) \right)$$

The matching strategy is essentially an *alignment* that selects a few "representative" segments of the set A and computes a rigid transformation based on an hypothesized correspondence between the representative segments of A and a group of segments of B. It then verifies the hypothesis by computing the distance measure for such a transformation. The above steps are repeated for all possible groups of segments of A. More precisely, the algorithm looks for a rigid body transformation g (translation plus rotation) that minimizes the distance between two sets of segments, A and B and it consists of the following three main steps:

Step 1 determine a translation P;
Step 2 determine a rotation R;
STep 3 evaluate the distance between $g(A)$ and B, where g is the combined transformation.

The rigid body transformation is obtained by selecting three representatives for each of the two sets A and B that are affine independent elements.

First, a representative segment a for A is randomly chosen. This representative is paired with each element b of B. For each such pair (a, b), the translation P is defined by taking the mid-point a_m of a into the mid-point b_m of b. This choice of the translation minimizes the distance between the transformed segment $P(a)$ and b.

To define the rotation, two additional independent elements of A are needed. The second representative a' is chosen as the segment containing the point a'_f farthest from a_m. The third representative segment a'' is chosen so that it contains the point a''_d at maximum distance from the line $\overline{a_m a'_f}$. It is easy to see that the points a'_f and a''_d must each be an endpoint of some segment. The condition that is enforced is the affine independence of the three points a_m, a'_f and a''_d. These choices bind the error due to the approximation. The next step of the algorithm is to choose the segments b' and b'' of B in all the m^2 possible ways. For each b' and each endpoint of b', consider the rotation that has origin in a_m and that makes a'_f and a_m to become collinear with the endpoint of b'. Define R' as the one of the above rotations that minimizes the distance between a'_f and the endpoints of b'. Then define R'' to be the rotation about the axis $a_m a'_f$ that brings a''_d closest to an endpoint of b''. Apply the transformations $R'' R' P(A)$. Finally choose over all the triplets b, b' and b'' the transformation g that resulted in the smallest distance.

The time complexity of the overall algorithm is $O(mn^3 \log n)$. In fact, using the Hausdorff metric, the nearest neighbor query in a set of segments (to identify the segment of B "closest" to a segment of A), reduces to a nearest neighbor query among points in $\mathcal{R}^6$ that can be performed in optimal $O(\log n)$ time within a known error bound. It is shown that the error introduced with the approximation is within a bounded factor from optimal. This bound is the same as the bound obtained in [15] for the simpler case of point sets.

Experiments have been conducted on several proteins and the results were consistent with previous studies. As an example presented in [22], figure 6 shows the superposition of two sets of segments associated to proteins 1rpa and 1rpt, with very similar structures.

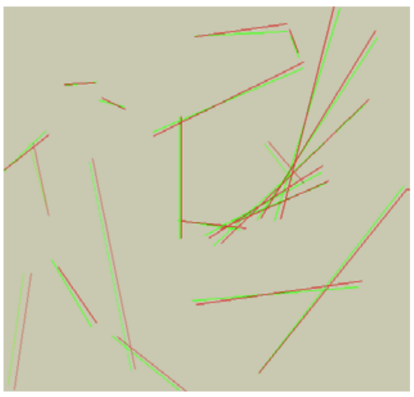

Fig. 6. The alignment of proteins 1rpa (red segments) and 1rpt (green segments)

7 Indexing Techniques

Indexing techniques, initially proposed in the field of computer vision by Wolfson et al., have found interesting applications in the area of bioinformatics. Indexing or geometric hashing provides a way to efficiently search a large database of proteins by storing redundant transformation invariant information about the proteins in a hash table, from which this information can be easily retrieved. The construction of the hash table, that constitutes the most complex part of the entire process, is done off-line at a preprocessing stage.

Indexing techniques have been applied to compare proteins at different levels of representations [2], [8], [13], [14], [32], [36], [55] (see also the chapter by H. Wolfson of this volume).

One major distinction of the comparison approaches is whether they are *order dependent* or *order independent*, in other words whether the use the order of the elements along the protein chain as a constraint in the correspondence process. Indexing techniques do not take into consideration the order of elements (either points or secondary structures) along the chain and therefore fall into the category of order-idenpendent methods.

For matching 3D point sets, quadruples of points are used to define reference frames or bases in which the coordinates of all other points remain invariant. Models are stored into the table by considering all possible combinations of quadruples of points as bases and using the invariant coordinates of the remaining points to index the table. At recognition time, if the correct quadruple of points is chosen from the image points, the candidate matches are efficiently retrieved from the corresponding entries of the hash table.

Here we concentrate on the application of hashing techniques at the secondary structure level involving transformation invariant properties of vectors associated to the secondary structures. The programs 3dSEARCH [51] and 3d-Lookup [26], based on hashing, compute geometric properties of pairs of secondary structures. They both construct the hash table by a procedure that consists of the following steps for the insertion of a protein in the database:

Step 1. For each pair of vectors of the protein, compute a reference frame or coordinate system identified by the two endpoints of one vector and by the orientation of the other vector.
Step 2. For each remaining vector in the protein, compute its coordinates in the reference frame defined in the previous step 1.
Step 3. The coordinates are quantized into fixed size interval and used to access the entry of the table corresponding to those coordinates where the following pair of information is stored: 1) name of the protein that hashed into it; 2) identifiers of the two vectors used as reference frame.

Once the hash table is built, each secondary structure vector from the given query structure is simultaneously compared to the entire library of target structures by simply indexing into this table. Thus, to compare a query protein to all target proteins the above step 1 and 2 are repeated for the query protein. Step 3 is replaced by the following:

Step 3' The coordinates are quantized into fixed size interval and used to access the entry of a 3d table corresponding to those coordinates where a vote is cast to every pair (protein name, two vector identifiers) present at that entry.

At the end of the process the proteins in the table which obtained the most votes are the candidates for matching.

A recently proposed approach [21] considers triplets of secondary structures rather than pairs to build the hash table. The three dihedral angles associated to all triplets of secondary structures are used to index a hash table. Let (s_i, s_j, s_k) be a triplet of segments, where s corresponds either to an α-helix or a β-strand. Let α_{sr} be the dihedral angle formed by two segments s and r. The dihedral angle between two segments is the angle formed by the two planes perpendicular to the straight lines containing the segments themselves and therefore is defined in the range [0, 180].

The triplet of segments (s_i, s_j, s_k) is then described by the three angles $(\alpha_{ij}, \alpha_{jk}, \alpha_{ki})$. The three angles, quantized into uniform intervals, provide three indeces for the table; a fourth index *triplet_type* is used to access the table; it depends on the types of the secondary structures in the triplets, whether all α helices, one α helix and two β strands and so forth. Thus a 4-dimensional table is constructed during the set up phase of the method. No explicit information is present in the tables about the order of the segments along the polypeptide chain.

Each entry of the table keeps a record of each triplet hashed into it. The record contains the following information: 1) the name of the protein, 2) the identifiers for the three vectors, 3) the pairwise distances between the three vectors. Such distances are used to filter incorrect hypotheses of associations in the matching process, because false results could be obtained based on angular information only. The distance is measured as the distance between the middle points of two segments.

The construction of the table is computation intensive; it requires $O(n^3)$ time, for n secondary structures. Once it is built it allows fast retrieval of candidate matches between the query protein and the proteins stored in the database. The space requirements for this approach may be high. However, the table is only partially occupied since the three angles are related by the triangular inequality.

The table can be queried to find similarities in the arrangement of the secondary structures of a query protein with the proteins stored in the database. A protein P is matched against the database of proteins by the following procedure:

Step 1 For each triplet (s_i, s_j, s_k) of secondary structures of P, compute the three angles (α, β, γ) and the three distances of the associated segments.

Step 2 Access the cell of the hash table indexed by $(\alpha, \beta, \gamma, \mathit{triplet\_type})$ and tally a vote for each entry in the cell with similar distance values.

Step 3 Formulate and rank hypotheses of matching by determining the proteins with the highest number of votes.

The verification of the hypothesized matches may be performed by a pairwise comparison between the proteins, either at the level of secondary structures [23] or by extending the matching to residue level.

Once compiled, the table can be used for different types of comparisons, for instance for all-to-all structure comparison.

Experiments have been conducted that consisted in building the hash table for all proteins in the PDB (approx. 14.000) have shown that the approach is both robust and efficient. The construction of such a table for approximately 350 representative proteins from the PDB has led to interesting observations about the distribution of the angles of the secondary structures which deviate from the distribution of randomly chosen vectors more significantly than one would have expected [47].

8 Graph-Theoretic Approaches

Here we give a brief description of the use of graphs to represent protein structures and of the techniques used to identify common substructures within proteins or to search for specific patterns in databases of molecules. Early work on protein matching based on graph theory was done by Brint and Willett [7]. To compare two or more structures, the method generates a graph of correspondences where each node represents a pair of atoms, one for each protein, and an edge connects two nodes (a, b) and (a', b') if the difference of the distances $d(a, a')$ and $d(b, b')$ is below a certain tolerance. Problem 3 defined in section 3, that is finding a maximum-cardinality set of element pairs such that the distance between each pair is at most a given value δ, can then be restated as one of determining the maximal clique of a graph. Finding a clique in a graph is a well known NP-complete problem.

In [57] graph isomorphism is used to search for pharmacophoric 3D patterns in a database. The method of representation of a 3D structure is the connection table, which contains the list of all atoms of the structure together with the bond information that describes the way in which the atoms are linked together. The connection table is basically a graph where the nodes represent the individual atoms and the edges represent interatomic distances. The graph is complete in the sense that there is an edge connecting every pair of nodes. A variant of this graph is obtained by using angular information to label the edges. The angle can be either the torsion angle or the valence angle defined as the angle between three bonded atoms. The presence of a query substructure in a database can be tested by means of a subgraph isomorphism. Subgraph isormorphism is computationally expensive and therefore cannot be used to test all entries in the database. Thus a screening strategy is suggested to reduce the overall execution time by eliminating most of the entries of the database from consideration by subgraph isomorphism. This screening is analogous of an index technique in that it provides access to a small fraction of the database by a preprocessing operation that groups all elements with similar characteristics. An extension to flexible matching is provided as well.

The approach by Escalier at al. [12] finds the largest similar subsets of atoms by recursively building subsets of increasing sizes, combining two subsets of size k to build a subset of size $k + 1$. Two subsets can be combined to form a larger one if they differ in one element only and their inter-atomic distances are all below a given threshold. Thus, this problem is equivalent to a clique finding problem. A suitable tree data structure allows an efficient implementation of the merge operation. As stated by the authors, the approach is suitable for small (less than 30 atoms) molecules. For larger structures such as proteins, a brute force application of the algorithm may lead to unreasonable execution times. Heuristics have to be introduced to reduce the amount of computation. One such heuristic is to split the problem in two parts: the first is to identify local fragments and the second is to assemble them together.

A graph-based approach was used in [20], [41] to compare secondary structure motifs in proteins. Proteins and motifs are represented as labelled graphs with the nodes corresponding to the segments associated s to secondary structures and the arcs to the angular and spatial relations between segments. Subgraph isomorphism is used to identify common structural patterns in pairs of proteins or to search for motifs in the PDB. The Ullman's algorithm for subgraph isosmorphism [54] was found to be sufficiently fast for the search of small motifs from he PDB. Chains of lines segments, corresponding to either secondary structures or to linear representation of other fragments of the backbone have also been considered [1].

9 Integration of Methods for Protein Comparison Using Different Representations

Different representations offer richer source of information that can be used in the comparison. This approach has been already investigated in the literature resulting in effective tools. Alignment of superfamily members has been obtained through conservation of structural features such as solvent accessibility, hydrogen bonding and the presence of secondary structures [42], [45], [50], [53].

An hierarchical protein structure superposition using both secondary structures and backbone atoms was recently proposed by Singh and Brutlag [51]. The local alignment of secondary structures is obtained by a variation of the Smith-Waterman dynamic programming algorithm [52]. A score function is used in the dynamic programming to measure the degree of similarity between pairs of vectors (linear segments) and is an attribute that may be either orientation independent (like the angle between two vectors within the same protein) or orientation dependent (like the angle between two vectors corresponding to two structures each belonging to one protein of the pair). Other attributes may relate distances between segments within the same or different proteins. The expression for the score function S used in this approach, similar to that used by Gerstein and Levitt [18], is given by:

$$S = \frac{2M}{1+[d/d_0]^2} - M$$

where M is a weighting factor for the attribute being measured, d the attribute value and d_0 is the value at which the score should be 0. An important choice of a dynamic programming approach is how to assign gap penalties. For secondary structure alignment it may be appropriate to decide to introduce no penalty because often the deletion of a secondary structure is due to an incorrect assignment in the PDB or to a mutation that changed a single secondary structure element, say a strand, into two structures or converted a strand into a turn.

Once an initial superposition of the secondary structures has been obtained by the dynamic programming algorithm, it is refined by iteratively minimizing the RMSD between pairs of nearest atoms from the two proteins. An interesting feature of the approach is that it does not simply rely on RMSD for judging the quality of the alignment but it takes into consideration also the number of “well” aligned atoms. Well aligned atoms define the “core” of the proteins and are selected as follows. Pairs of atoms, one atom from each protein, are selected so that each atom of the pair is the nearest atom of the other atom of the pair in the other protein. Furthermore, to be included in the core, such pairs of atoms have to satisfy the co-linearity property i.e. if (i, j) and (h, k) are two such pairs and $i < h$ then it must be $j < k$. Thus this method is order-dependent, according to one major classification of protein comparison approaches. The last step of the algorithm is to try to improve on the superposition of the core atoms even at the cost of degrading the alignment of the rest of the atoms.

The algorithm is efficient in terms of computational complexity and spends most of the execution time on the secondary and atomic alignment and a small fraction on the alignment of the core structures.

10 Conclusions

The problem of protein comparison can be successfully approached by first considering the related geometric issues. In the paper, the power and limitations of the different algorithms for protein structure comparison have been reviewed and discussed. Most of them have already proved their utility in computer vision and image processing, as well as in robotics, astronomy and physics. Their use in molecular biology and bioinformatics opens new perspectives for developing integrated methods for protein comparison, classification and engineering. Even if the different methods have been introduced to be used within different applications (characterized by different requirements), they solve particular instances of a more general matching problem, as deeply investigated in the area of computational geometry. The variety of protein representations supports the reasoning both at the level of points (i.e. atomic level) and at level of segments (or secondary structures). Estimation of rigid transformations with different metrics is an important technique within the protein structure comparison algorithms. Moreover geometric indexing techniques prove their effectiveness in searching large protein databases. Finally, graph-theoretic protein modeling helps in designing algorithms for substructure identification and comparison.

From the current research it has been recognized that the combination of different methods and different protein representations may result in new and effective algorithms with decreased computational complexity and better speed. Solvent accessibility, hydrogen bonding and the presence of secondary structures can be considered together in the alignment of superfamilies, while a hierarchical protein structure superposition can be obtained using both secondary structures and backbones atoms. For a better characterization of the proteins functions and their evolutionary information, geometric reasoning should be coupled with some proper chemical consideration involving hydrophobicity, charge, etc. The goal is to use domain-specific information for allowing a better pruning of the possible association choices at a very early stage of the matching process.

11 Acknowledgements

Support for Guerra was provided in part by the Italian Ministry of University and Research under the National Project "Bioinformatics and Genomics Research", and by the Research Program of the University of Padova.

Support for Ferrari was provided in part by the Italian Ministry of University and Research and by the Research Program of the University of Padova.

References

1. Abagyan, R.A. , Maiorov, V.N. (1989). An Automatic Search for Similar Spatial Arrangements of α helices and β-strands in globular proteins. *Journal of Molecular Structural Dynamics*, **6**, 5, 1045-1060.
2. Alesker, A., Nussinov, R., Wolfson H.J. (1996). Detection of non-topological motifs in protein structures. *Protein Engineering*, **9**, 5, 1103-1119.
3. Alter, T. D. (1992). Robust and efficient 3D recognition by alignment. Technical Report AITR-1410, Massachusetts Institute of Technology, Artificial Intelligence Laboratory.
4. Arun, K.S., Huang,T.S., Blostein, S.D. (1987). Least-square fitting of two 3-D point sets. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **9**, 5, 698-700.
5. Branden, C., Tooze, J. (1999) *Introduction to Protein Structure*, (second edition), Garland.
6. Brown, N.P., Orengo C.A., Taylor W.R. (1996). A protein structure comparison methodology. *Comp. Chem.* , **27**, 359-380.
7. Brint, A.T., Willett, P. (1987). Algorithms for the identification of three-dimensional maximal common substructures. *J. Chem. Inform. Comput. Sci.*, **27**, 152-156.
8. Califano, A., Mohan, R. (1992). Multidimensional indexing for recognizing visual shapes. *IEEE Trans. on Pattern Analysis and machine Intelligence*, **16**, 4, 373-392.
9. Chen, H. H., and Huang, T. S. (1990) Matching 3d line segments with application to multiple-objects motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**, 1002-1008.

10. Chew, P.L., Dor D.,pEfrat A., Kedem, K. (1995). Geometric pattern matching in *d*-dimensional space, *Algorithms-ESA '95*, 264-279.
11. Chew, L.P., Goodrich M.T., Huttenlocher,D.P., Kedem,K., Kleinberg, J.M., Kravets, D. (1997). Geometric pattern matching under Euclidean motion, *Computational Geometry. Theory and Applications*, **7**, 113-124.
12. Escalier V., Pothier, J., Soldano, H., Viari, A. (1998). Pairwise and Multiple Identification of three-dimensional common substructures in proteins. *J. of Computational Biology*, **5**, 41-56,
13. Fischer, D., Bachar, O., Nussinov, R., and Wolfson, H. (1992). An efficient automated computer vision based technique for detection of three dimensional structural motifs in proteins. *J. Biomol. Struct. Dyn.*, **9**, 769-789.
14. Fischer, D., Tsai, C.J., Nussinov, R., and Wolfson, H. (1995). A 3D sequence-independent representation of the protein data bank. *Protein Engineering*, **8**, 981-997.
15. Goodrich, M.T., Mitchell, J.S.B., Orletsky, M.W. (1999) Practical Methods for Approximate Geometric Pattern Matching Under Rigid Motion. *IEEE Trans. Pattern Analysis and Machine Intelligence*, **21**, 4, 371-379.
16. A. Glassner, (editor), *Graphics Gems*, Academic Press, 1990.
17. Gerstein, M. (1992). A Resolution-Sensitive Procedure for Comparing Protein Surfaces and its Application to the Comparison of Antigen-Combining Sites. *Acta Cryst.*, **A8**, 271-276.
18. Gerstein, M., Levitt, M. (1998). Comprehensive Assessment of automatic structural alignment against a manual standard, the scop classification of proteins. *Protein Science*, **7**, 445-456.
19. Golub, G.H., Van Loan C.F. , (1996) *Matrix Computation*, Johns Hopkins University Press.
20. Grindley, H.M., Artymiuk, P.J., Rice, D.W., and Willett, P. (1993). Identification of tertiary structure resemblance in proteins using a maximal common subgraph isomorphism algorithm. *J. Mol. Biol.*, **229**, 707-721.
21. Guerra, C., Lonardi, S., Zanotti, G., (2002). Analysis of proteins secondary structures using indexing techniques. *IEEE Proc First Int. Symposium on 3D Data Processing Visualization and Transmission*, 812-821.
22. C. Guerra, V. Pascucci. (1999) On matching sets of 3D segments", *Proceedings of SPIE Vision Geometry VIII*, 157-167.
23. Guerra, C., Pascucci, V. (1999) 3D segment matching using the Hausdorff distance. *Proceedings of the IEEE Conference on Image Processing and its Applications, IPA99*, 18-22.
24. Hagedoorn, M., Veltkamp, R. C. (1999). Reliable and efficient pattern matching using affine invariant metric. *Int. J. of Computer Vision*, **31**, (2/3), 203-225.
25. B. Horn, H. Hilden, S. Negahdaripour (1988). Closed-form solution of absolute orientation using orthonormal matrices. *J. Opt. Soc. Am.*, **5**, 1127-1135.
26. Holm, L., Sander, C. (1995). 3D-Lookup: Fast protein structure database searches at 90% reliability. *Proc. Third Int. Conf. on Intell. Sys. for Mol. Biol.*, Menlo Park, 179-187.
27. Holm, L., Sander, C. (1996). Mapping the protein universe. Science, *Science*, **273**, 595-602.
28. B. Horn, (1987). Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am.*, **4**, 629-642.

29. Huttenlocher, D. P., Klanderman, G. A., and Rucklidge, W. J. (1993) Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**, 9, 850-863.
30. Kabsch, W., Sander, C., (1983). Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, **22**, 2577-2637.
31. Kanmgar-Parsi, B., and Kamgamr-Parsi, B. (1997) Matching sets of 3d line segments with application to polygonal arc matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**, 10, 1090-1099.
32. Y. Lamdan, J. T. Schwartz, H. J. Wolfson. (1990). Affine invariant model-based object recognition, *IEEE Trans. on Robotics and Automation*, 578-589.
33. Lancia, G., Carr, R., Walenz, B., Istrail, S. (2001). Optimal PDB structure alignments: A Branch-and-Cut algorithm for the maximum contact map overlap problem. *Proc. 5th ACM REsearch in COMputational Biology*, 193-202.
34. Laskowski R.A., MacArthur M.W., Moss D.S., Thornton J.M. (1992). Stereochemical duality of protein structure coordinates. *Proteins*, **12**, 345-364.
35. Laskowski R.A., MacArthur M.W., Moss D.S., Thornton J.M. (1993). PROCHECK: a program to check the stereochemical quality of protein structures. *J. Appl. Cryst.* , **26**, 283-291.
36. Leibowitz, N., Fligelman, Z.Y., Nussinov, R., Wolfson, H.J. (1999). Multiple structural alignment and core detection for geometric hashing. *Proc. ISMB99*, Heidelberg, Germany, 169-177.
37. Lemmen, C., Lengauer, T. (2000). Computational methods for the structural alignment of molecules. *J. of Computer-Aided Molecular Design*, **14**, 215-232.
38. Lesk, A. M. (1991).*Protein architecture: a practical approach.* Oxford Univ. Press, Oxford.
39. Lesk, A. (1994). Computational Molecular Biology. *Encyclopedia of Computer Science and Technology*, **31**, Marcel Dekker, Inc..
40. Eggert,D.W., Lorusso, A., Fisher, R.B. (1997). Estimating 3-D rigid body transformations: a comparison of four major algorithms, *Mach. Vis. and Applic.*, **9**, 272-290.
41. Mitchell, E.M., Artymiuk, P.J., Rice, D.W. Willett, P. (1989) Use of Techniques derived from graph theory to compare secondary structures motifs in proteins. *J. Molecular Biology*, **212**, 151-166.
42. Mizuguchi, K., Deane, C.M., Blundell, T.L., Johnson, M.S. and Overington,J.P. (1998). JOY: protein sequence-structure representation and analysis. *Bioinformatics*, **14**, 617-623.
43. Murray, R.M, Li, X., Sastry, S.S., (1994). *A Mathematical Introduction to Robotic Manipulation*, CRC Press.
44. Murzin, A.G., Brenner, S.E., Hubbard, T., Chothia, C. (1995) SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Molecular Biology*, **247**, 536-540.
45. J. P. Overington, Z. Y. Zhu, A. Sali, M. S. Johnson, R. Sowdhamini, G. V. Louie, and T. L. Blundell. (1993). Molecular recognition in protein families: a database of aligned three-dimensional structures of related proteins. *Biochem. Soc. Trans.*, **21** , 3, 597-604.
46. Pennec, X., Ayache, N. (1998) A geometric algorithm to find small but highly similar 3D substructures in proteins. *Bionformatics*, **14**, 6, 516-522.
47. Platt, D.E., Guerra, C., Rigoutos, I., Zanotti, G. (2002). Global secondary structure packing angle bias in proteins. Manuscript.

48. Rucklidge, W. J. (1997) Efficiently locating objects using the Hausdorff distance. *International Journal of Computer Vision*, **24**, 3, 251-270.
49. Sabata, B., Aggarwal, J.K. (1991). Estimatiom of motion from a pair of range images: a review. *Computer Vision, Graphics and Image Processing: Image Understanding*, **54**, 3, 309-324.
50. Sali A., Blundell, T.L. (1990). Definition of a General Topological Equivalence in protein structures: a procedure involving comparisons of properties and relationships through simulated annealing and dynamic programming, *Journal of Molecular Biology*, **212**, 403-428.
51. Singh,A.P., Brutlag, D.L., (1997). Hierarchical protein structure superposition using both secondary structures and atomic representations. *Proc. Fifth Int. Conf. on Intell. Sys. for Mol. Biol.*, Menlo Park, 284-293.
52. Smith, T.F., Waterman, M.S., (1981). Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195-197.
53. Sowdhamini R., Burke D.F, Huang J-F, Mizuguchi, K. Nagarajaram H.A., Srinivasan N., Steward RE. and Blundell T.L. (1998). CAMPASS: A database of structurally aligned protein superfamilies. *Structure*, **6**, 9, 1087-1094.
54. Ullman, J.R. (1995) *J. Assoc. Comp.*, **23**, 31-42.
55. Verbitsky, G. Nussinov, R., Wolfson H.J. (1998). Structural comparisons allowing hinge bendings, swiveling motions. *Proteins*, **34**, 232-254.
56. Walker M.W, Shao L., Voltz R.A., Estimating 3-D Location Parameters Using Dual Number Quaternions, *CVGIP:Image Understanding*, **54**, 3, 1991, 358-367
57. Willett, P. (1995) Searching fore pharmacophoric patterns in databases of three-dimensional chemical structures. *J. of Molecular Recognition*, **8**, 290-303.