Efficient Unbound Docking of Rigid Molecules

Dina Duhovny[1,⋆], Ruth Nussinov[2,3,⋆⋆], and Haim J. Wolfson[1]

[1] School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel
{duhovka,wolfson}@post.tau.ac.il
[2] Sackler Inst. of Molecular Medicine, Sackler Faculty of Medicine
Tel Aviv University
[3] IRSP - SAIC, Lab. of Experimental and Computational Biology, NCI - FCRDC
Bldg 469, Rm 151, Frederick, MD 21702, USA
ruthn@fconyx.ncifcrf.gov

Abstract. We present a new algorithm for unbound (real life) docking of molecules, whether protein–protein or protein–drug. The algorithm carries out rigid docking, with surface variability/flexibility implicitly addressed through liberal intermolecular penetration. The high efficiency of the algorithm is the outcome of several factors: (*i*) focusing initial molecular surface fitting on localized, curvature based surface patches; (*ii*) use of Geometric Hashing and Pose Clustering for initial transformation detection; (*iii*) accurate computation of shape complementarity utilizing the *Distance Transform*; (*iv*) efficient steric clash detection and geometric fit scoring based on a multi-resolution shape representation; and (*v*) utilization of biological information by focusing on *hot spot* rich surface patches. The algorithm has been implemented and applied to a large number of cases.

1 Introduction

Receptor-ligand interactions play a major role in all biological processes. Knowledge of the molecular associations aids in understanding a variety of pathways taking place in the living cell. Docking is also an important tool in computer assisted drug design. A new drug should fit the active site of a specific receptor. Although electrostatic, hydrophobic and van der Waals interactions affect greatly the binding affinity of the molecules, shape complementarity is a necessary condition. The docking problem is considered difficult and interesting for a number of reasons. The combinatorial complexity of the possible ways to fit the surfaces is extremely high. The structures of the molecules are not exact, containing experimental errors. In addition, molecules usually undergo conformational changes upon association, known as induced fit. Docking algorithms must be tolerant to those difficulties, making the docking task one of the most challenging problems in structural bioinformatics.

⋆ To whom correspondence should be addressed

⋆⋆ The publisher or recipient acknowledges right of the U.S. Government to retain a nonexclusive, royalty-free license in and to any copyright covering the article.

R. Guigó and D. Gusfield (Eds.): WABI 2002, LNCS 2452, pp. 185–200, 2002.

There are two instances in the docking task - 'bound' and 'unbound'. In the 'bound' case we are given the co-crystallized complex of two molecules. We separate them artificially and the goal is to reconstruct the original complex. No conformational changes are involved. Successful application of an algorithm to 'bound' docking cases is necessary to test its validity, yet it does not ensure success in the real-life 'unbound' docking prediction, where we are given two molecules in their native conformations. In this case the docking algorithm should consider possible conformational changes upon association. Most of the docking algorithms encounter difficulties with this case, since shape complementarity is affected [14].

The goal of docking algorithms is to detect a *transformation* of one of the molecules which brings it to optimal fit with the other molecule without causing steric clash. Naturally, optimality here depends not only on geometric fit, but also on biological criteria representing the resulting complex stability. Molecular docking algorithms may be classified into two broad categories: (*i*) brute force enumeration of the transformation space; (*ii*) local shape feature matching.

Brute force algorithms search the entire 6-dimensional transformation space of the ligand. Most of these methods [33,30,31,32,11,2,3] use brute force search for the 3 rotational parameters and the FFT (Fast Fourier Transform, [19]) for fast enumeration of the translations. The running times of those algorithms may reach days of CPU time. Another brute force algorithm is the 'soft docking' method [17] that matches surface cubes. There are also non-deterministic methods that use genetic algorithms [18,12].

Local shape feature matching algorithms have been pioneered by Kuntz [20]. In 1986 Connolly [6] described a method to match local curvature maxima and minima points. This technique was improved further in [23,24] and was also applied to unbound docking [25]. Additional algorithms that employ shape complementarity constraints, when searching for the correct association of molecules were also developed [21,13,10]. Some algorithms are designed to handle flexible molecules [28,27,9].

Our method is based on local shape feature matching. To reduce complexity we, first, try to detect those molecular surface areas which have a high probability to belong to the binding site. This reduces the number of potential docking solutions, while still retaining the correct conformation. The algorithm can treat receptors and ligands of variable sizes. It succeeds in docking of large proteins (antibody with antigen) and small drug molecules. The running times of the algorithm are of the order of seconds for small drug molecules and several minutes for large proteins. In addition, we improve the shape complementarity measure, making the function more precise and reducing the complexity of its computation.

2 Methods

Our docking algorithm is inspired by object recognition and image segmentation techniques used in computer vision. We can compare docking to assembling a

jigsaw puzzle. When solving the puzzle we try to match two pieces by picking one piece and searching for the complementary one. We concentrate on the patterns that are unique for the puzzle element and look for the matching patterns in the rest of the pieces. Our algorithm employs a similar technique. Given two molecules, we divide their surfaces into patches according to the surface shape. These patches correspond to patterns that visually distinguish between puzzle pieces. Once the patches are identified, they can be superimposed using shape matching algorithms. The algorithm has three major stages:

1. **Molecular Shape Representation** - in this step we compute the molecular surface of the molecule. Next, we apply a segmentation algorithm for detection of geometric patches (concave, convex and flat surface pieces). The patches are filtered, so that only patches with 'hot spot' residues are retained [15].
2. **Surface Patch Matching** - we apply a hybrid of the Geometric Hashing [34] and Pose-Clustering [29] matching techniques to match the patches detected in the previous step. Concave patches are matched with convex and flat patches with any type of patches.
3. **Filtering and Scoring** - the candidate complexes from the previous step are examined. We discard all complexes with unacceptable penetrations of the atoms of the receptor to the atoms of the ligand. Finally, the remaining candidates are ranked according to a geometric shape complementarity score.

2.1 Molecular Shape Representation

Molecular Surface Calculation. The first stage of the algorithm computes two types of surfaces for each molecule. A high density Connolly surface is generated by the MS program [5,4]. The calculated surface is preprocessed into two data structures: distance transform grid and multi-resolution surface (see Appendix). Those data structures are used in the scoring routines. In addition, the distance transform grid is used further in the shape representation stage of the algorithm. Next, a sparse surface representation [22] is computed. It is used in the segmentation of the surface into geometric patches.

The sparse surface consists of points nicknamed 'caps', 'pits' and 'belts', where each cap point belongs to one atom, a belt to two atoms and a pit to three atoms. These correspond to the face centers of the convex, concave and saddle areas of the molecular surface [22]. The gravity center of each face is computed as a centroid and projected to the surface in the normal direction.

Detection of Geometric Patches. The input to this step is the sparse set of critical points. The goal is to divide the surface to patches of almost equal area of three types according to their shape geometry: concavities, convexities and flats. We construct a graph induced by the points of the sparse surface. Each node is labeled as a 'knob', 'flat' or a 'hole' according to their curvature (see below). We compute connected components of knobs, flats and holes. Then we apply the *split* and *merge* routines to improve the component partitioning of the surface to patches of almost equal area.

Surface Topology Graph. Based on the set of sparse critical points, the graph $G_{top} = (V_{top}, E_{top})$ representing the surface topology is constructed in the following way:

$$V_{top} = \{Sparse\ Critical\ Points\}$$
$$E_{top} = \{(u, v) \mid if\ u\ and\ v\ belong\ to\ the\ same\ atom\}$$

The number of edges in the graph is linear, since each pit point can be connected by an edge to at most three caps and three belts. Each belt point is connected to two corresponding caps (see figure 2 (a)).

Shape Function Calculation. In order to group the points into local curvature based patches we use the shape function defined in [6]. A sphere of radius R is placed at a surface point. The fraction of the sphere inside the solvent-excluded volume of the protein is the *shape function* at this point. The shape function of every node in the graph is calculated. The radius of the shape function sphere is selected according to the molecule size. We use 6Å for proteins and 3Å for small ligands. As a result every node is assigned a value between 0 and 1. In previous techniques [23] points with shape function value less than $\frac{1}{3}$, named 'knobs' and points with value greater than $\frac{2}{3}$, named 'holes', were selected as critical points. All 'flat' points with function value between those two were ignored. As can be seen from the histogram of the shape function values of the trypsin molecule (PDB code 2ptn) in figure 1, a large number of points are 'flats'. In fact about 70% of the points are flats and about 30% are knobs or holes. (These statistics are typical for other molecules as well.) Consequently, the problem was, that the number of matching sets of quadraples/triples/pairs of knobs versus holes was very low [6]. Here we sort the shape function values and find two cut-off values that split the nodes to three equal sized sets of knobs, flats and holes.

Simultaneously with the shape function calculation, the *volume normal* orientation is computed using the same probe sphere. The solvent-accessible part is defined as the complement of the solvent-excluded part of the probe sphere. Given a probe sphere we define the *volume normal* to be the unit vector at the surface point in the direction of the gravity center of solvent-accessible part.

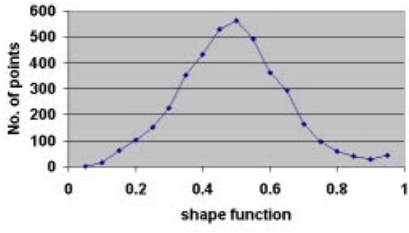


Fig. 1. Histogram of shape function values for the trypsin molecule (PDB code 2ptn).

Patch Detection. The idea is to divide the surface of the molecule into non-intersecting patches of critical points. The geometric patch is defined as a connected set of critical points of the same type (knobs, holes or flats). By 'connected' we mean that points of the patch should correspond to a connected subgraph of G_{top}. To assure better matching of patches, they must be of almost equal sizes. For each type of points (knobs, flats, holes) the graphs G_{knob}, G_{flat} and G_{hole} are constructed as subgraphs of G_{top} induced on the corresponding set of nodes.

The algorithm for finding connected components in a graph is applied to each of the three graphs. The output components are of different sizes, so *split* and *merge* routines are applied to achieve a better partition to patches. We present some definitions that are used below:

1. The **geodesic distance** between two nodes of the connected component is a weighted shortest path between them, where the weight of every edge is the Euclidean distance between the corresponding surface points.
2. The **diameter of the component** is defined as the largest geodesic distance between the nodes of the component. Nodes s and t that give the diameter are called **diameter nodes**. In case the diameter nodes are not unique we arbitrarily choose one of the pairs as the single diameter.

For each connected component we compute the diameter and it's nodes by running the APSP (*All Pairs Shortest Paths* [7]) algorithm on the component. We use two thresholds for the diameter of the components as follows:

- If the diameter of the connected component is more than *low_patch_thr* and less than *high_patch_thr*, the component represents a valid geometric patch.
- If the diameter of the connected component is larger than *high_patch_thr* the *split* routine is applied to the component.
- If the diameter of the connected component is less than *low_patch_thr* the points of this component are merged with closest components.

Note that a connected component is not always a patch, since the component may be split to a number of patches or merged with other patches.

Split routine. Given a component C, the distance matrix from the APSP algorithm and it's diameter nodes s, t we split it into two new components S and T that correspond to the Voronoi cells [8] of the points s, t. If the diameter of each new component is within the defined thresholds, the component is added to the list of valid patches, otherwise it is split again.

Merge routine. The goal is to merge points of small components with 'closest' big patches. Those points correspond to the components that are usually located between hole and knob patches, where surface topology changes quickly from concave to convex. For each point of small components we compute the geodesic distance to every valid patch using the Dijkstra [7] algorithm. The point is added to the closest patch.

At this stage most of the surface is represented by three types of patches of almost equal areas (see figure 2(b)). Now, we attempt to detect those patches, which are most likely to appear in the binding sites of the molecules.

Detection of Active Sites. The success of the docking can be significantly improved by determining the active site of the molecules. Knowledge of the binding site of at least one molecule greatly reduces the space of possible docking interactions. There are major differences in the interactions of different types of molecules (e.g. enzyme–inhibitor, antibody–antigen). We have developed filters for every type of interaction and focus only on the patches that were selected by the appropriate filter.

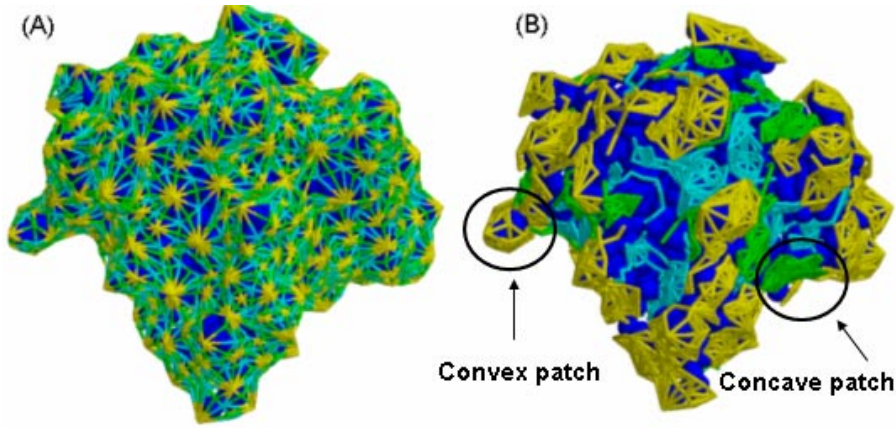


Fig. 2. (a) Surface topology graphs for trypsin inhibitor (PDB code 1ba7). The caps, belts and pits are connected with edges. (b) Geometric patches: the patches are in light colors and the protein is dark.

Hot Spot Filtering. A number of studies have shown that protein-protein interfaces have conserved polar and aromatic 'hot-spot' residues. The goal is to use the statistics collected [15,16] in order to design a patch filter. The filter is supposed to select patches that have high probability of being inside the active site. Residue hot spots have experimentally been shown (via alanine scanning mutagenesis) to contribute more than 2 Kcal/mol to the binding energetics. In order to measure it, we compute the *propensity of a residue in a patch* $Pr(res_i, patch)$ as the fraction of the residue frequency in the patch compared to the residue frequency on the molecular surface. Subsequently, we choose patches with the high propensities of hot spot residues which are: (*i*) Tyr, Asp, Asn, Glu, Ser and Trp for antibody; (*ii*) Arg, Lys, Asn and Asp for antigen; (*iii*) Ser, Gly, Asp and His for protease; and (*iv*) Arg, Lys, Leu, Cys and Pro for protease inhibitor.

Antibody-Antigen interactions: Detection of CDRs. It is well known that antibodies bind to antigens through their hypervariable (HV) regions, also called complementarity-determining regions (CDRs) [1]. The three heavy-chain and three light-chain CDR regions are located on the loops that connect the β strands of the variable domains. We detect the CDRs by aligning the sequence of the given antibody to a consensus sequence of a library of antibodies. The docking algorithm is then restricted to patches which intersect the CDR regions.

2.2 Surface Patch Matching

Given the patches of a receptor and a ligand, we would like to compute hypothetical docking transformations based on local geometric complementarity. The idea is that knob patches should match hole patches and flat patches can match any patch. We use two techniques for matching:

1. *Single Patch Matching:* one patch from the receptor is matched with one patch from the ligand. This type of matching is used for docking of small ligands, like drugs or peptides.
2. *Patch-Pair Matching:* two patches from the receptor are matched with two patches from the ligand. We use this type of matching for protein-protein docking. The motivation in patch-pair matching is that molecules interacting with big enough contact area must have more than one patch in their interface. Therefore matching two patches simultaneously will result in numerically more stable transformations. For this purpose we develop a concept of neighboring patches: two patches are considered as neighbors if there is at least one edge in G_{top} that connects the patches.

Both utilize the Computer Vision motivated Geometric Hashing [34] and Pose Clustering [29] techniques for matching of critical points within the patches. At first local features of the objects are matched to compute the candidate transformations that superimpose them. After matching of the local features a clustering step is applied to select poses (hypothetical transformations) with strong evidence, i.e. transformations consistent with a large enough number of matching local features.

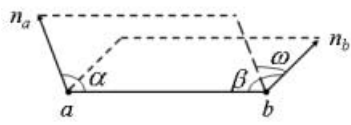


Fig. 3. Base formed by two points and their normals.

Generation of Poses. We implement the matching step using Geometric Hashing [34]. There are two stages in the matching algorithm: preprocessing and recognition. We denote by a **base** a minimal set of critical features which uniquely defines a rigid transformation. A transformation invariant shape signature is calculated for each base. In the preprocessing stage, each base is stored in a hash-table according to its signature. In the recognition stage, the bases of the receptor are computed. The base signature is used as a key to the hash-table. For each ligand base found in the hash-table a candidate transformation that superimposes local features of the receptor and the ligand is computed.

We use two points (a, b) and their volume normals (n_a, n_b) as the base for transformation calculation [23]. The base signature $(dE, dG, \alpha, \beta, \omega)$ is defined as follows (see figure 3):

- The Euclidean and geodesic distance between the two points: dE and dG.
- The angles α, β formed between the line segment ab and each of the normals n_a, n_b.
- The torsion angle ω formed between the plane of a, b, n_a and the plane of a, b, n_b.

Two signatures, one from the receptor and the other from the ligand, are compatible if their signatures are close enough. Note that we do not demand here matching of a knob to a hole, since it is ensured by matching knob patches with hole patches.

Clustering Poses. Matching of local features may lead to multiple instances of 'almost' the same transformation (similar pose). Therefore, clustering is necessary to reduce the number of potential solutions. We apply two clustering techniques: clustering by transformation parameters and RMSD clustering. Clustering by transformation parameters is coarse but very fast, and is applied first. After the number of transformations is significantly reduced, we run RMSD clustering, which is more exact, but also much slower.

2.3 Filtering and Scoring

Since the transformation is computed by local matching of critical points within patches, it may result in unacceptable steric clashes between the receptor and ligand atoms. We should filter out all those transformations. In addition, we need to rank the rest of the solutions.

Steric Clashes Test. In this stage the distance transform grid is extensively used. For each candidate transformation we perform the steric clash test as follows. The transformation is applied on the surface points of the ligand. Next we access the distance transform grid of the receptor with the coordinates of every surface point. If the distance is less than *penetration threshold* for each surface point, the transformation is retained for the next step, otherwise the transformation is disqualified.

Geometric Scoring. The general idea is to divide the receptor into shells according to the distance from the molecular surface. For example, in [23,10] a receptor was divided into 3 shells and a grid representing them was constructed as follows: interior(I) grid voxels corresponding to interior atoms (no surface point generated for them), exterior(E) voxels for exterior atoms and surface(S) voxels for surface MS dots. The score of the transformation was a weighted function of ligand surface points in each range: *S-4E-10I*. We have generalized this approach and made it more accurate using a distance transform grid (see Appendix). Each shell is defined by a range of distances in the distance transform grid. Instead of using 3 shells we can use any number of shells and the score is a weighted function of the number of ligand surface dots in each shell. In the current algorithm implementation 5 shells are used: $[-5.0, -3.6)$, $[-3.6, -2.2)$, $[-2.2, -1.0)$, $[-1.0, 1.0)$, $[1.0-)$. In the scoring stage for each candidate transformation we count the number of surface points in each shell. The geometric score is a weighted average of all the shells, when we prefer candidate complexes with large number of points in the $[-1.0, 1.0)$ shell, and as little as possible points in the 'penetrating' shells: $[-5.0, -3.6)$, $[-3.6, -2.2)$.

Those algorithms provide very accurate filtering and scoring, but are also very slow, especially when high-density surface is used for the ligand. In order to speed-up this part of the program we use a multi-resolution molecular surface data structure (see Appendix). We construct two trees: one of high density using Connolly's MS Surface and the other of lower density, using the sparse surface [22] as the lowest level. The tree based on the sparse surface is used for the

primary scoring of the transformations. The tree based on the denser MS surface is used for penetration (steric clash) check and for fine geometric scoring.

Given a set of candidate transformations from the matching step, we first check the steric clashes. Only transformations with 'acceptable' penetrations (less than 5Å) of the atoms are retained. These transformations are scored with the low density based tree. We select 500 high scoring transformations for every patch and re-score them using the high density surface.

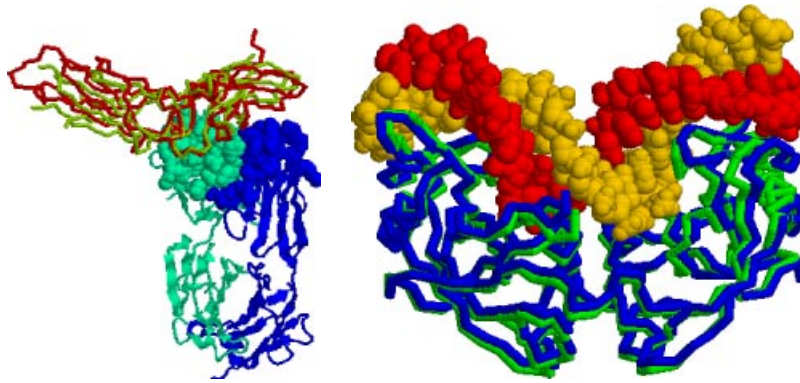
The remaining transformations can be further re-ranked according to biological criteria.

3 Results and Discussion

It is not trivial to detect data to test unbound docking algorithms. Note that we need to know three structures. The structures of the two molecules to be docked as well as the structure of their complex, so that we could reliably test our prediction results. Thus, the maximal currently available benchmark contains only few tens of molecule pairs.

Table 1 lists the 35 protein-protein cases from our test set. Our dataset includes 22 enzyme/inhibitor cases and 13 antibody/antigen cases. In 21 cases the unbound structures of both proteins were used in docking, in the other 14 cases the unbound structure for only one molecule was available. Our method is completely automated. The same set of parameters is used for each type of interactions. In the enzyme/inhibitor docking we match 'hole' patches of the enzyme with 'knob' patches of the inhibitor, since the inhibitor usually blocks the active site cavity of the enzyme. In the antibody/antigen docking we match all patch types, since the interaction surfaces are usually flat compared to enzyme/inhibitor. In addition we restrict our search to the CDRs of the antibody.

The final results are summarized in table 2. All the runs were made on a PC workstation (Pentium©II 500 MHz processor with 512MB internal memory). First, we present the results for the bound cases. As can be seen the correct solution was found for all the cases. In 31 out of the 35 examples, the lowest RMSD achieved is below 2Å. The first ranked result is the correct one in 26 cases. In other 9 cases the correct solution is ranked among the first 30 results. Columns 4 and 5 describe the steric clashes that occur when the unbound structures are superimposed on the bound complex. It shows the extent of shape complementarity in every example. Column 4 lists the maximal penetration of the ligand and receptor surfaces into each other. In some cases it is more than 5Å. Column 5 lists the number of residues of the receptor and the ligand respectively that cause deep steric clashes (more than 2.2Å surface penetration). The number of those residues is more then 10 in 4 cases. In the unbound-bound cases the numbers are significantly lower, allowing us to reach better results. We do not list the penetration level for the bound complexes since the numbers are very small: the maximal surface penetration is below 2.5Å. In the results obtained for the unbound cases we succeeded in finding solutions with RMSD under 5Å in all but 3 cases. In those cases (PDB codes 1DFJ, 2SNI, 1DQJ) shape complementarity



(a) Antibody–antigen (b) Protein–DNA

Fig. 4. (a) Unbound docking of the Antibody Fab 5G9 with Tissue factor (PDB codes 1FGN,1BOY). The antibody is depicted in ribbons representation and the CDRs are in spacefill. The antigen and the solution obtained by our program are depicted in backbone representation (RMSD 2.27Å, total rank 8). (b) Protein-DNA docking: unbound-bound case (PDB codes 1A73,1EVX). Best RMSD obtained 0.87, rank 2. The DNA is shown in spacefill. Our solution superimposed on the native complex is shown in backbone representation.

is affected by many side-chain movements. The rank is under 350 in 17 out of 22 enzyme/inhibitor cases and is under 1000 in 11 out of 13 antibody/antigen cases. The best ranked result for antibody Fab 5G9/Tissue factor is shown in figure 4(a).

The program was tested on additional examples (Protein-DNA, Protein-Drug) which are not shown here. In figure 4(b) we show one of these results.

The rank of the correct solution depends on a number of factors:

1. **Shape complementarity** - steric clashes introduce noise to the scoring functions, reducing the score and the rank of the correct solution.
2. **Interface shape** - it is much easier to find correct association in molecules with concave/convex interface (enzyme/inhibitor cases) rather than with flat interfaces (antibody/antigen cases). The shape complementarity is much more prominent in the concave/convex interfaces and therefore easier to detect.
3. **Sizes of the molecules** - the larger the molecules the higher the number of results.

In our algorithm the division to patches and selection of energetic hot spots reduce the area of the matched surfaces, therefore the number of possible docking configurations is decreased and the ranking is improved.

Table 1. The dataset of Enzyme/Inhibitor and Antibody/Antigen test cases. We list the PDB codes of the complex and the unbound structures, protein names and the number of amino acids in every protein. The unbound-bound cases are marked with *.

Complex	Receptor	Ligand	Description	Rec. size	Lig. size
1ACB	5CHA	1CSE(I)	α-chymotrypsin/Eglin C	236	63
1AVW	2PTN	1BA7	Trypsin/Sotbean Trypsin inhibitor	223	165
1BRC	1BRA	1AAP	Trypsin/APPI	223	56
1BRS	1A2P	1A19	Barnase/Barstar	108	89
1CGI	1CHG	1HPT	α-chymotrypsinogen/pancreatic secretory trypsin inhibitor	226	56
1CHO	5CHA	2OVO	α-chymotrypsin/ovomucoid 3rd Domain	236	56
1CSE	1SCD	1ACB(I)	Subtilisin Carlsberg/Eglin C	274	63
1DFJ	2BNH	7RSA	Ribonuclease inhibitor/Ribonuclease A	456	124
1FSS	2ACE	1FSC	Acetylcholinesterase/Fasciculin II	527	61
1MAH	1MAA	1FSC	Mouse Acetylcholinesterase/inhibitor	536	61
1PPE*	2PTN	1PPE	Trypsin/CMT-1	223	29
1STF*	1PPN	1STF	Papain/Stefin B	212	98
1TAB*	2PTN	1TAB	Trypsin/BBI	223	36
1TGS	2PTN	1HPT	Trypsinogen/Pancreatic secretory trypsin inhibitor	223	56
1UDI*	1UDH	1UDI	Virus Uracil-DNA glycosylase/inhibitor	228	83
1UGH	1AKZ	1UGI	Human Uracil-DNA glycosylase/inhibitor	223	83
2KAI	2PKA	6PTI	Kallikrein A/Trypsin inhibitor	231	56
2PTC	2PTN	6PTI	β-trypsin/ Pancreatic trypsin inhibitor	223	56
2SIC	1SUP	3SSI	Subtilisin BPN/Subtilisin inhibitor	275	108
2SNI	1SUP	2CI2	Subtilisin Novo/Chymotrypsin inhibitor 2	275	65
2TEC*	1THM	2TEC	Thermitase/Eglin C	279	63
4HTC*	2HNT	4HTC	α-Thrombin/Hirudin	259	61
1AHW	1FGN	1BOY	Antibody Fab 5G9/Tissue factor	221	211
1BQL*	1BQL	1DKJ	Hyhel - 5 Fab/Lysozyme	217	129
1BVK	1BVL	3LZT	Antibody Hulys11 Fv/Lysozyme	224	129
1DQJ	1DQQ	3LZT	Hyhel - 63 Fab/Lysozyme	215	129
1EO8*	1EO8	2VIR	Bh151 Fab/Hemagglutinin	219	267
1FBI*	1FBI	1HHL	IgG1 Fab fragment/Lysozyme	226	129
1JHL*	1JHL	1GHL	IgG1 Fv Fragment/Lysozyme	224	129
1MEL*	1MEL	1LZA	Vh Single-Domain Antibody/Lysozyme	132	127
1MLC	1MLB	1LZA	IgG1 D44.1 Fab fragment/Lysozyme	220	129
1NCA*	1NCA	7NN9	Fab NC41/Neuraminidase	221	388
1NMB*	1NMB	7NN9	Fab NC10/Neuraminidase	229	388
1WEJ	1QBL	1HRC	IgG1 E8 Fab fragment/Cytochrome C	220	104
2JEL*	2JEL	1POH	Jel42 Fab Fragment/A06 Phosphotransferase	228	85

Table 2. Results obtained for the test set. We list the PDB code of each complex, results for bound cases, level of penetration in the unbound complexes superimposed on bound and results for the unbound cases. [a] Best RMSD(Å) result. In the brackets is its rank in the list of the results for the patch it was found in. [b] Best ranking of the result with RMSD under 5Å among all the results. [c] Maximal penetration between the surfaces in Å. [d] Number of residues that penetrate the surface with distance greater than 2.2Å for receptor and ligand respectively. [e] The running time of matching and scoring step for the unbound cases.

	Bound cases		Penetrations in unbound		Unbound cases		
PDB code	RMSD(Å) (patch rank)[a]	Best rank[b]	penetration distance[c]	# of residues[d]	RMSD(Å) (patch rank)[a]	Best rank[b]	CPU (min)[e]
1ACB	1.05(23)	1	2.43	2,0	1.12(34)	10	59:48
1AVW	0.82(1)	1	2.79	7,3	1.92(223)	330	55:11
1BRC	1.07(96)	1	4.76	7,0	4.76(168)	179	15:10
1BRS	0.66(1)	1	2.79	3,1	3.19(85)	143	24:09
1CGI	1.21(2)	1	3.30	4,4	1.74(338)	135	21:06
1CHO	1.77(198)	2	3.28	7,0	1.25(4)	5	14:25
1CSE	1.43(1)	1	2.99	3,0	1.76(478)	603	28:34
1DFJ	1.84(15)	15	4.81	16,9	7.04(10)	-	35:16
1FSS	2.35(115)	4	3.14	5,7	1.64(360)	142	32:38
1MAH	0.71(2)	1	3.25	6,2	2.47(24)	736	21:07
1PPE*	0.98(1)	1	2.18	0,0	0.96(1)	1	07:31
1STF*	0.56(1)	1	1.94	0,0	1.32(1)	1	30:39
1TAB*	1.63(21)	1	2.62	1,0	1.72(35)	81	07:01
1TGS	0.71(2)	1	4.30	8,5	2.82(445)	573	17:17
1UDI*	1.35(2)	1	3.02	2,4	1.83(6)	73	20:33
1UGH	0.84(1)	1	3.31	6,8	2.48(151)	44	17:06
2KAI	0.89(2)	1	4.67	14,10	3.21(235)	224	20:59
2PTC	0.74(3)	1	4.07	9,2	1.86(222)	10	13:45
2SIC	1.49(3)	3	2.91	7,6	1.30(6)	122	29:59
2SNI	2.22(1)	1	5.64	12,5	6.95(392)	-	14:09
2TEC*	0.93(27)	1	2.61	1,0	0.77(29)	241	31:29
4HTC*	1.76(3)	1	2.72	2,2	1.54(1)	1	09:04
1AHW	0.83(1)	1	3.02	4,3	1.73(98)	8	52:06
1BQL*	0.96(3)	1	2.30	0,0	0.66(1)	1	22:51
1BVK	1.32(10)	1	2.01	0,0	2.91(185)	577	08:37
1DQJ	1.32(1)	1	4.08	12,13	5.24(244)	-	20:56
1EO8*	1.19(1)	1	2.75	3,4	2.27(168)	1071	33:22
1FBI*	1.46(2)	1	4.01	7,1	1.05(228)	282	20:41
1JHL*	1.30(167)	3	1.94	0,0	2.91(134)	274	27:01
1MEL*	1.45(2)	1	2.39	1,0	1.20(3)	2	07:30
1MLC	1.17(112)	3	4.23	7,9	3.10(397)	689	15:55
1NCA*	1.69(1)	1	1.74	0,0	1.92(16)	43	20:30
1NMB*	2.79(17)	17	1.40	0,0	2.07(44)	218	15:09
1WEJ	1.47(11)	11	2.16	0,0	2.09(260)	417	16:06
2JEL*	3.67(4)	4	2.38	3,0	3.37(45)	112	08:39

4 Conclusions

We have presented a novel and highly efficient algorithm for docking of two molecules. While here we have shown results obtained by applying the algorithm to the docking of two protein molecules, the algorithm can be applied to receptor–drug cases as well (not shown here). The attractive running times of the algorithm and the high quality of the results compared to other state of the art method [26,12,3] are the outcome of several components. First, the algorithm divides the molecular surface into shape-based patches. This division addresses both the efficiency and at the same time, distinguishes between residue types (polar/non-polar) in the patches. Further, we make use of residue hot spots in the patches. Second, the method utilizes distance transform to improve the shape complementarity function. Third, it implements faster scoring, based on multi-resolution surface data structure. Our improved shape complementarity function further contributes to the quality of the results. While here the docking is rigid, the utilization of the last three components enables us to permit more liberal intermolecular penetration (up to 5 Å here).

References

1. C. Branden and J. Tooze. *Introduction to Protein Structure*. Garland Publishing, Inc., New York and London, 1991.
2. J.C. Camacho, D.W. Gatchell, S.R. Kimura, and S. Vajda. Scoring docked conformations generated by rigid body protein–protein docking. *PROTEINS: Structure, Function and Genetics*, 40:525–537, 2000.
3. R. Chen and Z Weng. Docking unbound proteins using shape complementarity, desolvation, and electrostatics. *PROTEINS: Structure, Function and Genetics*, 47:281–294, 2002.
4. M.L. Connolly. Analytical molecular surface calculation. *J. Appl. Cryst.*, 16:548–558, 1983.
5. M.L. Connolly. Solvent-accessible surfaces of proteins and nucleic acids. *Science*, 221:709–713, 1983.
6. M.L. Connolly. Shape complementarity at the hemoglobin $\alpha_1\beta_1$ subunit interface. *Biopolymers*, 25:1229–1247, 1986.
7. T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms, chapter 26*. The MIT Press, 1990.
8. M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 2000.
9. T.J.A. Ewing, Makino S., Skillman A.G., and I.D. Kuntz. Dock 4.0: Search strategies for automated molecular docking of flexible molecule databases. *J. Computer-Aided Molecular Design*, 15:411–428, 2001.
10. D. Fischer, S. L. Lin, H.J. Wolfson, and R. Nussinov. A geometry-based suite of molecular docking processes. *J. Mol. Biol.*, 248:459–477, 1995.
11. H.A. Gabb, R.M. Jackson, and J.E. Sternberg. Modelling protein docking using shape complementarity, electrostatics, and biochemical information. *J. Mol. Biol.*, 272:106–120, 1997.
12. E.J. Gardiner, P. Willett, and P.J. Artymiuk. Protein docking using a genetic algorithm. *PROTEINS: Structure, Function and Genetics*, 44:44–56, 2001.

13. B.B. Goldman and W.T. Wipke. Molecular docking using quadratic shape descriptors (qsdock). *PROTEINS: Structure, Function and Genetics*, 38:79–94, 2000.
14. I. Halperin, B. Ma, H. Wolfson, and R. Nussinov. Principles of docking: An overview of search algorithms and a guide to scoring functions. *PROTEINS: Structure, Function and Genetics*, 47:409–443, 2002.
15. Z. Hu, B. Ma, H.J Wolfson, and R. Nussinov. Conservation of polar residues as hot spots at protein–protein interfaces. *PROTEINS: Structure, Function and Genetics*, 39:331–342, 2000.
16. R.M. Jackson. Comparison of protein-protein interactions in serine protease-inhibitor and antibody-antigen complexes: Implications for the protein docking problem. *Protein Science*, 8:603–613, 1999.
17. F. Jiang and S.H. Kim. Soft docking : Matching of molecular surface cubes. *J. Mol. Biol.*, 219:79–102, 1991.
18. G. Jones, P. Willet, R. Glen, and Leach. A.R. Development and validation of a genetic algorithm for flexible docking. *J. Mol. Biol.*, 267:727–748, 1997.
19. E. Katchalski-Katzir, I. Shariv, M. Eisenstein, A.A. Friesem, C. Aflalo, and I.A. Vakser. Molecular Surface Recognition: Determination of Geometric Fit between Protein and their Ligands by Correlation Techniques. *Proc. Natl. Acad. Sci. USA*, 89:2195–2199, 1992.
20. I.D. Kuntz, J.M. Blaney, S.J. Oatley, R. Langridge, and T.E. Ferrin. A geometric approach to macromolecule-ligand interactions. *J. Mol. Biol.*, 161:269–288, 1982.
21. H.P. Lenhof. Parallel protein puzzle: A new suite of protein docking tools. In *Proc. of the First Annual International Conference on Computational Molecular Biology RECOMB 97*, pages 182–191, 1997.
22. S. L. Lin, R. Nussinov, D. Fischer, and H.J. Wolfson. Molecular surface representation by sparse critical points. *PROTEINS: Structure, Function and Genetics*, 18:94–101, 1994.
23. R. Norel, S. L. Lin, H.J. Wolfson, and R. Nussinov. Shape complementarity at protein-protein interfaces. *Biopolymers*, 34:933–940, 1994.
24. R. Norel, S. L. Lin, H.J. Wolfson, and R. Nussinov. Molecular surface complementarity at protein-protein interfaces: The critical role played by surface normals at well placed, sparse points in docking. *J. Mol. Biol.*, 252:263–273, 1995.
25. R. Norel, D. Petrey, H.J. Wolfson, and R. Nussinov. Examination of shape complementarity in docking of unbound proteins. *PROTEINS: Structure, Function and Genetics*, 35:403–419, 1999.
26. P.N. Palma, L. Krippahl, J.E. Wampler, and J.G Moura. Bigger: A new (soft)docking algorithm for predicting protein interactions. *PROTEINS: Structure, Function and Genetics*, 39:372–384, 2000.
27. M. Rarey, B. Kramer, and Lengauer T. Time-efficient docking of flexible ligands into active sites of proteins. In *3'rd Int. Conf. on Intelligent Systems for Molecular Biology (ISMB'95)*, pages 300–308, Cambridge, UK, 1995. AAAI Press.
28. B. Sandak, H.J. Wolfson, and R. Nussinov. Flexible docking allowing induced fit in proteins. *PROTEINS: Structure, Function and Genetics*, 32:159–174, 1998.
29. G. Stockman. Object recognition and localization via pose clustering. *J. of Computer Vision, Graphics, and Image Processing*, 40(3):361–387, 1987.
30. I.A. Vakser. Protein docking for low resolution structures. *Protein Engineering*, 8:371–377, 1995.
31. I.A. Vakser. Main chain complementarity in protein recognition. *Protein Engineering*, 9:741–744, 1996.

32. I.A. Vakser, O.G. Matar, and C.F. Lam. A systematic study of low resolution recognition in protein–protein complexes. *Proc. Natl. Acad. Sci. USA*, 96:8477–8482, 1999.
33. P.H. Walls and J.E. Sternberg. New algorithms to model protein-protein recognition based on surface complementarity; applications to antibody-antigen docking. *J. Mol. Biol.*, 228:227–297, 1992.
34. H.J. Wolfson and I. Rigoutsos. Geometric hashing: An overview. *IEEE Computational Science and Eng.*, 11:263–278, 1997.

Appendix

Distance Transform Grid

Distance transform grid is a data structure for efficient queries of type *distance from surface*. The molecule is represented by a 3D grid, where each voxel (i, j, k) holds a value corresponding to the distance transform $DT(i, j, k)$. There are three types of voxels: surface (MS surface point maps to the voxel), interior (inside the molecule) and exterior (outside the molecule). The distances are zero at the surface voxels and change as the distance from the surface increases/decreases. The distance transform is negative for inside molecule voxels and positive for outside voxels.

Supported Queries.

- *distance from surface:* Given a point p, we access the grid with its coordinates and return a value of the voxel corresponding to the point. Clearly this query consumes $O(1)$ time.
- *shape function and volume Normal:* A sphere of radius R is placed at a given point p. We count the ratio of negative grid voxels inside the sphere and the total number of sphere voxels. The volume normal is computed in the same manner. We compute the gravity center of the positive grid voxels inside the sphere. This sets the direction of the normal.

Multi-resolution Surface

In order to support fast geometric scoring and filtering of the transformations, we construct multi-resolution data structure for the ligand's surface dots. We build a surface tree, where each layer represents the ligand surface at a different resolution. This data structure allows us to work with very high MS surface density and to reduce greatly the number of queries in the receptor distance transform grid. The main idea is that if the point falls outside the interface, there is no need to check where the adjacent points fall.

Supported Queries.

- *isPenetrating:* Given a transformation and a *penetration threshold* we check whether the surface points of the ligand penetrate the surface of the receptor with more then the given threshold.

- *maxPenetration:* Given a transformation find the maximum surface penetration.
- *score:* Given a transformation and a list of ranges, the goal is to count the number of lowest level surface points in each range.
- *interface:* Selects the interface surface points for a given transformation. The output is all the nodes with the distance transform value less then *interface threshold*.

All the queries employ the DFS search with iterative implementation using a stack. The complexity of the queries is proportional to high level size + interface size.