COARSE AND RELIABLE GEOMETRIC ALIGNMENT FOR PROTEIN DOCKING*

Y. WANG,† P. K. AGARWAL,‡ P. BROWN,§ H. EDELSBRUNNER¶ and J. RUDOLPH‖
Duke University, Durham, North Carolina

Abstract. We present an efficient algorithm for generating a small set of coarse alignments between interacting proteins using meaningful features on their surfaces. The proteins are treated as rigid bodies, but the results are more generally useful as the produced configurations can serve as input to local improvement algorithms that allow for protein flexibility. We apply our algorithm to a diverse set of protein complexes from the Protein Data Bank, demonstrating the effectivity of our algorithm, both for bound and for unbound protein docking problems.

1. Introduction

Protein-protein docking is the computational approach to predicting interactions between proteins. In this paper, we contribute to this field by describing an algorithm for generating a small set of coarse alignments between protein structures.

Motivation. Highly organized transient or static assemblies of proteins control most cellular events. A better understanding of the protein-protein interactions involved in these assemblies would help elucidate how individual proteins form complexes and dynamically function in concert to generate the cell circuitry and its time-dependent responses to external stimuli. Protein structures determined at atomic resolution by X-ray crystallography, nuclear magnetic resonance, and increasingly by computer modeling provide one basis for the study of protein interactions. However, given the relative wealth of structural details for monomeric proteins compared to multimeric protein complexes, there exists a need for computational tools and thus for the field of protein docking.

Prior work. Current research on protein-protein docking focuses on either bound

*All authors are supported by NSF under grant CCR-00-86013. JR and HE are also supported by NIH under grant R01 GM61822-01. PA is also supported by NSF under grants EIA-01-31905 and CCR-02-04118 and by the U.S.-Israel Binational Science Foundation.
†Department of Computer Science.
‡Departments of Computer Science and Mathematics.
§Department of Computer Science.
¶Departments of Computer Science and Mathematics, and Raindrop Geomagic.
‖Departments of Biochemistry and Chemistry.

docking (the reassembly of known complexes from their constituents), or unbound docking (the assembly of as yet unknown complexes under the assumption of only small protein conformational changes). Most approaches to unbound docking consist of two stages[20]: the *rigid docking stage* produces a set of potential docking configurations by considering only rigid motions, and the *refinement stage* locally improves the docking configuration, possibly allowing for a limited amount of flexibility. The two essential components in both stages are: a scoring function that discriminates near-native from incorrect docking configurations and a search algorithm to find (approximately) the best configuration for the scoring function.

Approaches to the rigid docking stage rely mainly on geometric complementarity. Some are based on uniform discretizations of the space of rigid motions, which they search exhaustively[3]. This approach has been accelerated using the fast Fourier transform (FFT)[16], which forms the basis of the docking software FTDock[13], 3D-Dock[18], GRAMM[21], and ZDock[6]. Others sample a small number of rigid motions non-uniformly from the space by aligning feature points found on the molecular surfaces[14,17]. This idea goes back to Connolly[11], who proposed to use the minima and maxima of a function related to mean curvature, now known as the *Connolly function*. An example of this method has been described by Fischer *et al.*[12], who use geometric hashing to align critical points of a variant of the Connolly function. The refinement stage is usually modeled as an energy minimization problem, with the scoring function focusing on the thermodynamic aspects of the interaction. The difficulty of the problem increases with the dimension of the search space or, equivalently, the degree of freedom, which is large even if we keep the back-bone rigid and consider only side-chain flexibility. Recently, Vajda *et al.* have proposed a hierarchical, progressive refinement protocol[4,5], which seems to reliably converge to a near-native docking configuration starting with initial configurations up to 10Å root-mean-square-distance away from the native configuration. Little success has been reported on including back-bone conformational changes[19]. Since each step in the refinement stage is costly, it is essential that the set of potential configurations generated in the rigid docking stage is small and reliably contains configurations not too far from the native configuration. Current solutions to the rigid docking stage fall short on at least one of the two requirements.

New work. In this paper, we present an efficient algorithm for the rigid docking stage. We use geometric complementarity to guide the search for a small set of rigid motions so that the two proteins fit loosely into each other. Such a set of potential configurations can be further refined to obtain more accurate docking predictions[5,9]. We remark that for the case of unbound docking, it is especially

important to start with coarse (not tight) fits between proteins to take advantage of flexibility in the later refinement stage.

We describe our algorithm in Section 2. It relies on a novel approach to describe protrusions and cavities on molecular surfaces using a succinct set of point pairs computed from the elevation function[1]. We then align such pairs and evaluate the resulting configurations using a simple and rapid scoring function. Compared to similar approaches that align feature points[12,17], our algorithm inspects orders of magnitude fewer configurations. This is made possible by using slightly more complicated features that contain information useful in assessing their significance. We exploit this extra information twice, first to ignore insignificant features and second to be more discriminant in matching up features from different proteins. In Section 3, we demonstrate the efficacy of our approach by testing a set of 25 bound protein complexes from the Protein Data Bank[2]. We demonstrate that a combination of our algorithm with the local improvement procedure described in[9] efficiently finds near-native docking positions for all but two cases without creating false positives. In addition, we test our algorithm on the unbound protein docking benchmark[7]. In particular, we demonstrate that the algorithm generates poses sufficiently close to the native configuration such that refinement methods that take into account protein flexibility will succeed in bringing them within an acceptable neighborhood of the correct solution. We conclude and discuss future work in Section 4.

2. Methods

We represent each protein by a set or union of finitely many balls in three-dimensional Euclidean space, which we denote by $\mathbb{R}^3$. Specifically, we are given two proteins, $A = \{A_1, A_2, \ldots, A_n\}$ and $B = \{B_1, B_2, \ldots, B_m\}$, where A_i is the ball with center $a_i \in \mathbb{R}^3$ and van der Waals radius $r_i \in \mathbb{R}$ and B_j is the ball with center b_j and van der Waals radius s_j. We fix A in $\mathbb{R}^3$ and describe an algorithm that finds a small set of candidate transformations for B. Each transformation is a rigid motion μ that produces a candidate configuration $(A, \mu(B))$. We begin by describing the scoring function that assesses the fit between two proteins.

Scoring function. A good scoring function favors near-native configurations over configurations that are far from the native. Letting $d_{ij} = \|a_i - b_j\| - r_i - s_j$ be the distance between the balls A_i and B_j, we define

$$\text{contact}(i,j); \text{collision}(i,j) = \begin{cases} -3;1 & \text{if} \quad d_{ij} < 0, \\ 1;0 & \text{if } 0 \leq d_{ij} \leq \lambda, \\ 0;0 & \text{if } \lambda < d_{ij}, \end{cases}$$

where λ is a constant we refer to as the *contact-threshold*. The score of (A, B)

is based on the total *number of contacts*, $\#\mathrm{cont}(A,B) = \sum_{i,j} \mathrm{contact}(i,j)$, and the total *number of collisions*, $\#\mathrm{coll}(A,B) = \sum_{i,j} \mathrm{collision}(i,j)$. We call the configuration (A,B) *valid* if $\#\mathrm{coll}(A,B) \leq \chi$, where the constant *collision-threshold* χ defines the maximum number of collisions we tolerate. The score ignores invalid configurations and equals the number of contacts for valid configurations. This notion of score is similar to the ones in[3,6] but different because our score penalizes collisions twice, first by counting them toward a possibly invalid configuration and second by reducing the contact number. The reason for this difference is that we aim at coarse alignments and thus are more tolerant to collisions, using $\chi = 60$ rather than $\chi = 5$, as in[3]. The second penalty counteracts the usual increase in contact number that goes along with an increase in collision number. In other words, it seeks to avoid a bias toward configurations with higher collision number without unfairly discriminating against them.

Features. Our algorithm generates rigid motions from feature sets Φ_A and Φ_B obtained by analyzing the shapes of the two proteins. We compute these features from (approximately) smooth surfaces representing the two shapes[8,10]. Letting $\mathbb{M}$ be the surface representing A, we briefly review the function $\mathrm{Elevation} : \mathbb{M} \to \mathbb{R}$ that underlies our definition of feature. To first approximation, it resembles the elevation on Earth, which is the height difference of a point and the mean sea level at that point. This definition makes sense on Earth, where we have a natural choice of origin (the center of mass) and mean sea level (a level set of the gravitational potential), neither of which exists for general surfaces. In the absence of both concepts, we associate each point $x \in \mathbb{M}$ with a canonically defined *partner* $y \in \mathbb{M}$, with same normal direction $\mathbf{n}_y = \mathbf{n}_x$, and define $\mathrm{Elevation}(x)$ as the absolute height difference between x and y in that common direction. For more details, in particular on how to define the canonical pairing, we refer to[1]. Loosely speaking, x is the top of a protrusion or the bottom of a cavity in the direction $\mathbf{n}_x$ and the pairing partner y is the saddle point that marks where the protrusion or cavity starts. It is also possible that the roles of x and y are reversed. For the purpose of protein docking, we are interested in points with locally maximal elevation, as they represent locally most significant features. Almost all points x on $\mathbb{M}$ have exactly one partner y, but most maxima arise at positions where the partner is ambiguous. More specifically, for a generic surface there are four types of maxima describing different types of features, as illustrated in Figure 1.

By definition, a *feature* consists of two points, u and its partner v, with common surface normal, $\mathbf{n}_u = \mathbf{n}_v$, and common elevation, $\mathrm{Elevation}(u) = \mathrm{Elevation}(v)$. Its *length* is the Euclidean distance between the two points, $\|u - v\|$. Each maximum of the elevation function is defined by $k \in \{2,3,4\}$

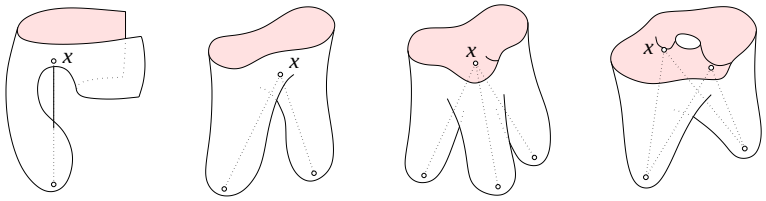

Figure 1. Left: a one-legged maximum characterized by x having a unique partner. Middle left: a two-legged maximum in which x has two partners, both with the same normal direction and the same height difference to x. Middle right: a three-legged maximum in which x has three partners, again sharing the same normal direction and the same height difference to x. Right: a four-legged maximum in which x has two partners and both partners have the same two partners each.

points and gives rise to $\binom{k}{2}$ features in Φ_A. For example, the 3-legged maximum (third from the left in Figure 1) consists of $k = 4$ points defining $\binom{k}{2} = 6$ point pairs each forming a feature in Φ_A. The length and elevation of a feature are used to estimate its importance, and both together with the normal direction are used to pair up features from the sets Φ_A and Φ_B.

Coarse alignment. Given two proteins A and B together with their feature sets Φ_A and Φ_B, our algorithm computes a set of potential coarse alignments Γ:

```
for every α ∈ Φ_A and every β ∈ Φ_B do
  if α, β form a plausible alignment then
    μ = Align(α, β);
    compute the contact and collision numbers for (A, μ(B));
    if (A, μ(B)) is valid then add μ to Γ endif
  endif
endfor;  sort Γ by contact number.
```

The rationale behind the algorithm is that good fits between the input proteins have aligned features, such as a protrusion of A fitting inside a cavity of B, or vice versa. If we pair up all features of A with all features of B, we surely cover all good fits. On the other hand, the information that comes with each feature can be used to discriminate between pairs and gain efficiency by filtering out alignments we deem not important or implausible. Specifically, we introduce an *importance filter* that eliminates features from Φ_A and Φ_B whose lengths or elevations are below threshold. The remaining features form pairs (α, β) which pass the *plausibility filter* provided α and β are not too different in length and they represent complementary types (a protrusion and a cavity). The constants used in the importance filter are given in the caption of Table1.

Assuming (α, β) passes the importance and the plausibility filters, we com-

pute an aligning rigid motion μ as follows. Writing $\alpha = (u, v, \mathbf{n}_\alpha)$ and $\beta = (p, q, \mathbf{n}_\beta)$ for the points and normals, we define the bi-normals $b_\alpha = \mathbf{n}_\alpha \times \frac{v-u}{\|v-u\|}$ and $b_\beta = \mathbf{n}_\beta \times \frac{p-q}{\|q-p\|}$. We obtain the rigid motion in three steps:

1. translate β so that the two midpoints coincide: $\frac{u+v}{2} = \frac{p+q}{2}$;
2. rotate β about the common midpoint so that u, v, p, q are collinear;
3. rotate β about the common line so that $b_\alpha = b_\beta$.

We note that there is an ambiguity in Step 2, allowing for two different alignments distinguished by having $v-u$ and $q-p$ point in the same or in opposite directions. We are interested in both but simplify the description by pretending that Function Align returns only one rigid motion, instead of two as it really does. Observe that Step 3 positions the two features to maximize the angle between the two normal vectors. Given μ, we compute the score and the number of collisions using a hierarchical data structure storing A and B. Letting n and m be the number of balls in the two sets, this takes time $O((n+m)\log(n+m))$.

3. Results

In this section, we present the results of testing our algorithm on twenty-five bound docking problems obtained from the Protein Data Bank[2] and on forty-nine unbound docking problems from the benchmark in[7]. We begin with a detailed study of a well known protein complex.

A case study. We use the barnase/barstar complex (pdb-id 1BRS, chains A and D, with 864 and 693 atoms) as a sample system to introduce the capabilities of our algorithm. We generate molecular surfaces of the two chains with the MSMS software (available as part of the VMD software distribution[15]) and obtain triangulations with 8,959 and 7,248 vertices. In Table 1, we show the total number

	chain A, # legs			chain D, # legs		
	2	3	4	2	3	4
total # of features	1,044	696	156	828	510	154
#s after importance filter	112	205	50	68	160	49

Table 1. Compare the total number of features obtained from two-, three- and four-legged maxima for chains A and D of 1BRS with the number of features that pass the importance filter, having length at least 3.0Å and elevation at least 0.2Å. (There are no one-legged maxima for this data set.)

of features generated from the maxima of the elevation function and the number of features that survive the importance filter. The latter form the input to our coarse alignment algorithm. We note that a substantially larger number of features

obtained from 3-legged maxima are retained than features obtained from 2- and 4-legged maxima.

Given the two sets of input features, our algorithm takes about three minutes on a single processor PIII 1GHz computer to generate a family Γ of 5,021 valid configurations with contact number larger than or equal to 150. Each configuration in Γ corresponds to a transformation μ for chain D. We use the root-mean-

before local improvement				after local improvement		
rank	#cont	#coll	RMSD	rank	#cont'	RMSD
12	327	24	3.23	1	359	0.54
5	342	48	2.42	2	338	0.80
1	427	23	1.59	3	328	0.72
4	353	49	3.57	4	314	0.80
2	391	39	1.70	5	311	0.91
59	269	12	2.84	6	310	0.78
3	373	29	2.32	7	307	1.50
11	339	18	3.07	8	281	1.47
15	318	16	3.00	9	251	2.09
76	263	29	39.39	10	213	39.96

Table 2. Top ten configurations after local improvement and their ranks before local improvement. The first nine have small RMSD and may be considered near-native configurations. We use different definitions for the number of contacts before and after the local improvement: #cont is defined as in Section 2, and #cont' is as computed by the local improvement algorithm, which is the number of non-overlapping spheres at distance at most 1.5Å.

square-distance (RMSD) between the centers of the matching atoms in D and μ(D) to measure how close the configuration is to the native one. Ranking by score, the top configuration in Γ has an RMSD of 1.59Å, and six of the top ten configurations have RMSD smaller than or equal to 4.0Å. Letting Γ^* be the subset of top $N = 100$ configurations, we refine each one using the local improvement heuristic of Choi et al.[9] We then re-rank the configurations in Γ^* based on the new scores, limiting ourselves to configurations with collision number at most five. The results in Table 2 show that our algorithm generates multiple coarse alignments that are useful, in the sense that the local improvement heuristic succeeds in refining them to near-native configurations.

More bound protein complexes. We extend our experiments to a collection of twenty-five protein complexes obtained from the Protein Data Bank. Each complex consists of two chains, and we generate a set of features for each. For a typical chain, the number of features that survive the importance filter is on the same order of magnitude as the number of atoms. In Table 3, we show a low-RMSD configuration for each protein complex, as well as its rank in the list of

configurations output by our algorithm (using contact-threshold $\lambda = 2.0$Å and collision-threshold $\chi = 60$). With only one exception (1JAT), we have at least one low-RMSD configuration ranked among the top one hundred. The last column shows the running time for the coarse alignment algorithm, which does not include the time to compute the triangulated surface and the maxima of the elevation function.

pdb-id	chains	rank	#coll	RMSD	time
1A22	A, B	2	23	2.75	20
1BI8	A, B	12	43	2.48	26
1BRS	A, D	1	11	1.52	3
1BUH	A, B	5	14	1.85	2
1BXI	B, A	3	34	2.54	8
1CHO	E, I	1	14	2.71	3
1CSE	E, I	2	22	2.21	9
1DFJ	I, E	78	11	3.09	27
1F47	B, A	15	1	1.49	1
1FC2	D, C	5	49	4.13	6
1FIN	A, B	11	44	3.70	41
1FS1	B, A	1	29	1.62	5
1JAT	A, B	522	20	1.20	9
1JLT	A, B	8	23	3.64	10
1MCT	A, I	1	27	3.49	3
1MEE	A, I	1	23	1.33	9
1STF	E, I	1	43	1.18	8
1TEC	E, I	9	54	3.07	7
1TGS	Z, I	1	46	2.61	6
1TX4	A, B	2	4	3.35	14
2PTC	E, I	1	18	4.55	6
3HLA	A, B	1	19	1.87	16
3SGB	E, I	1	38	3.21	5
3YGS	C, P	6	7	1.07	6
4SGB	E, I	10	33	2.33	4

Table 3. For each protein complex, we show data for the highest ranking configuration with RMSD at most 5.0Å. The running time of the coarse alignment algorithm is given in minutes.

Next, we apply the local improvement heuristic[9] to the top $N = 100$ configurations of each complex (except 1JAT, for which we need $N = 522$ to get a near-native configuration) and re-rank them based on the new scores. Eliminating all configurations with more than 5 collisions, Table4 shows before and after data for the configuration that is ranked at the top after local improvement. In all but two cases, the top ranked configuration is near-native, and in one of the two exceptional cases, the second ranked configuration is near-native. In the remaining exceptional case (1BI8), we can obtain a near-native configuration by relaxing

the threshold of allowed collisions to eight. In summary, for 23 of the 25 test complexes, our coarse alignment algorithm combined with the local improvement heuristic[9] predicts a near-native configuration without false positives.

	before local improvement				after local improvement		
pdb-id	rank	#cont	#coll	RMSD	rank	#cont'	RMSD
1A22	2	363	23	2.75	1	475	1.08
1BI8	62	324	10	30.00	1	234	29.88
1BRS	12	327	37	3.23	1	349	0.54
1BUH	5	311	14	1.85	1	256	0.61
1BXI	16	261	21	5.59	1	289	0.63
1CHO	1	375	14	2.71	1	305	0.99
1CSE	23	276	36	2.57	1	317	0.82
1DFJ	78	273	11	3.09	1	220	1.28
1F47	15	238	1	1.49	1	221	0.56
1FC2	5	323	49	4.13	2	200	1.33
1FIN	34	361	54	9.94	1	413	0.61
1FS1	2	402	27	1.59	1	326	0.89
1JAT	522	203	21	1.20	1	288	0.87
1JLT	3	362	14	6.17	1	310	1.77
1MCT	84	280	34	3.57	1	322	0.32
1MEE	1	542	23	1.33	1	372	0.57
1STF	1	444	43	1.18	1	314	0.79
1TEC	10	334	51	4.51	1	304	1.28
1TGS	2	373	13	2.71	1	348	0.44
1TX4	80	296	25	4.34	1	355	0.36
2PTC	1	346	18	4.55	1	314	0.66
3HLA	1	402	19	1.97	1	416	0.70
3SGB	1	364	38	3.21	1	257	2.24
3YGS	6	315	7	1.03	1	209	0.85
4SGB	10	298	33	2.33	1	266	2.50

Table 4. For each protein complex, we locally improve the N top ranked configurations and show the data for the highest re-ranked configuration with small RMSD. After local improvement we admit only configurations with at most five collisions, as usual. The number of contacts before and after the improvement, #cont and #cont', are computed as described in the caption of Table 2.

It is interesting to compare the data in Tables 3 and 4 and notice that the highest ranked configuration after local improvement is the highest ranked configuration with small RMSD before the local improvement in only slightly more than half the cases. Consider for example 1FIN, which has a configuration at 3.70Å RMSD with 44 collisions but the one that leads to the best final configuration has RMSD $= 9.94$Å and $\#\text{coll} = 54$.

Unbound docking benchmark. We further test our algorithm on the protein-protein docking benchmark provided in[7]. We omit the seven complexes classified as difficult in[7] because they have significantly different conformations in the un-

bound vs. bound structures. We also omit complexes 1IAI, 1WQ1 and 2PCC for which we had difficulties to generate surface triangulations of required quality. Of the remaining forty-nine complexes, twenty-five are so-called *bound-unbound*

bound-unbound						unbound-unbound					
C-id	#hits	min*	rank	size	min	C-id	#hits	min*	rank	size	min
1ACB	20	3.70	3,951	14,426	1.75	1MLC	7	3.71	6,949	29,747	3.32
1AVW	8	5.51	4,698	23,565	5.42	1WEJ	3	6.27	4,659	18,194	5.86
1BRC	35	4.66	1,629	12,770	4.66	1BQL	11	6.98	10,388	23,308	4.39
1BRS	7	1.60	426	11,607	1.60	1EO8	1	2.31	11	45,512	2.31
1CGI	5	3.04	695	10,135	3.04	1FBI	8	6.49	11,783	26,036	2.30
1CHO	27	2.35	92	11,815	2.35	1JHL	18	3.47	14,185	32,091	2.61
1CSE	7	3.15	15,271	21,068	2.74	1KXQ	2	5.99	1,495	37,218	5.99
1DFJ	2	6.44	1,433	35,231	6.44	1KXT	12	4.52	153	39,240	4.52
1FSS	2	7.65	10,721	25,609	5.15	1KXV	7	2.48	321	46,368	2.48
1MAH	4	2.78	1,561	25,402	2.78	1MEL	8	2.21	73	17,741	2.21
1TGS	18	5.27	543	11,383	5.27	1NCA	7	1.75	621	49,600	1.75
1UGH	3	7.95	8,268	14,656	7.16	1NMB	7	7.18	14,202	42,066	2.72
2KAI	26	6.55	2,560	13,478	3.41	1QFU	4	1.97	12	47,693	1.97
2PTC	32	4.55	4,983	13,929	4.16	2JEL	19	3.46	115	34,072	3.46
2SIC	27	4.04	76	20,065	4.04	2VIR	11	1.08	1	40,813	1.08
2SNI	10	6.34	4,894	15,830	4.58	1AVZ	8	4.06	4,243	7,895	3.52
1PPE	10	4.13	37	7,660	4.13	1L0Y	2	2.75	1,136	34,044	2.75
1STF	8	1.41	1	15,082	1.41	2MTA	40	2.91	19,167	36,903	2.07
1TAB	3	3.78	48	8,296	3.78	1A0O	3	5.95	3,950	9,113	4.35
1UDI	3	4.50	1,124	21,133	4.50	1ATN	8	1.52	1	50,729	1.52
2TEC	5	1.42	6	21,134	1.42	1GLA	-	-	25,307	33,879	2.82
4HTC	2	5.94	396	14,032	5.94	1IGC	3	2.48	3,260	25,303	2.06
1AHW	1	9.38	2,781	32,919	4.37	1SPB	3	2.83	617	13,728	2.83
1BVK	5	1.95	1,189	24,611	1.95	2BTF	2	5.02	10,132	33,480	3.28
1DQJ	7	4.59	710	28,694	4.59						

Table 5. Twenty-five bound-unbound cases on the left plus twenty-four unbound-unbound cases on the right. From left to right: the complex identification, the number of configurations in Γ^* with RMSD* less than or equal to 10.0Å, the smallest RMSD* value of any configuration in Γ^* (min*), the rank of this configuration within Γ, the number of configurations in Γ, and the smallest RMSD* value of any configuration in Γ (min).

cases, in which one of the components is rigid. For each complex, we fix one chain as A, which is the rigid chain for each bound-unbound case and the receptor for each unbound-unbound case. We generate Γ, a set of the potential configurations, each corresponding to a rigid motion μ applied to the other chain, B. For each μ, we measure the root-mean-square-distance between the matching interface C_α atoms of B and μ(B), and refer to it as RMSD*. Similar to the bound docking case, this value is a good estimate for the distance to the native configuration since the benchmark provides the unbound structures superimposed onto their corresponding crystallized bound structures. For each complex, we let Γ^* be the subset of top $N = 2,000$ configurations in Γ. We show the results of our experiments in Table 5, demonstrating a number of favorable characteristics of our coarse alignment algorithm:

1. Within the relatively small set of 2,000 top-scoring configurations, Γ^*, about 78% of the complexes yield a configuration below 6.0Å RMSD and about 98% yield a configuration below the 10.0Å cut-off needed as input for the hierarchical, progressive refinement protocol in[4,5].

2. For most complexes, our algorithm generates multiple hits, implying that a local refinement is not likely to get trapped in a local minimum and instead find a near-native configuration.
3. Within the set of all generated configurations, Γ, about 96% of the complexes yield a configuration below 6.0Å, typically within the top 10,000 scores. All 49 complexes generate at least one configuration below 7.5Å within the top 25,000 scores.

We remark that there are at least two ways to further improve the results: use a different ranking mechanism that moves more low-RMSD configurations into the top ranks, and reduce the size of Γ by clustering similar configurations[12].

4. Discussion

We conclude this paper with a brief comparison of our results with prior work on bound and unbound docking. We classify the bound docking methods by how they sample the search space of rigid motions. Methods that sample densely and more or less uniformly predict more accurate rigid docking configurations, but at a high computational cost. To adapt these methods to unbound docking, we may run the algorithms at low resolution or select a small set of promising candidate configurations for further refinement. As of today, neither approach has produced a workable solution to the problem of unbound docking. Methods that sample the space of rigid motions in a biased manner rely on some sort of shape analysis, aimed at detecting locally complementary configurations. All prior work is based on point features marking protrusions and cavities. Alignments are created by matching the points, e.g. all pairs from one set with all pairs from another. The running time is often improved using geometric hashing, as in[12].

Our algorithm belongs to the second class of methods but differs from prior work in the nature of the features, which are point pairs with extra information useful in estimating the scale level and in finding promising matches. Using this information, we generate significantly sparser samples of the search space. Our experiments provide evidence that despite the lower density, we always get candidates that can be refined to near-native configurations. The algorithm is reasonably fast and improvements are still possible.

Acknowledgement. The authors would like to thank Vicky Choi for the local improvement software used in our experiments.

References

1. P. K. Agarwal, H. Edelsbrunner, J. Harer and Y. Wang. Extreme elevation on a 2-manifold. In *Proc. 20th Ann. Sympos. Comput. Geom.*, 357–365, 2004.

2. H. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. Shinkdyalov and P. E. Bourne. The protein data bank. *Nucleic Acid Res.*, 28:235–242, 2000.
3. S. Bespamyatnikh, V. Choi, H. Edelsbrunner and J. Rudolph. Accurate protein docking by shape complementarity alone. Manuscript, Duke Univ., Durham, NC, 2004.
4. C. J. Camacho, D. W. Gatchell, S. R. Kimura and S. Vajda. Scoring docked conformations generated by rigid-body protein-protein docking. *Proteins: Struct. Funct. Genet.*, 40:525–537, 2000.
5. C. J. Camacho and S. Vajda. Protein docking along smooth association pathways. *Proc. Natl. Acad. Sci.*, 98:10636–10641, 2001.
6. R. Chen, L. Li, and Z. Weng. ZDOCK: an initial-stage protein docking algorithm. *Proteins: Struct. Funct. Genet.*, 52:80–87, 2003.
7. R. Chen, J. Mintseris, J. Janin and Z. Weng. A protein-protein docking benchmark. *Proteins: Struct. Funct. Genet.*, 52:88–91, 2003.
8. H.-L. Cheng, T. K. Dey, H. Edelsbrunner and J. Sullivan. Dynamic skin triangulation. *Discrete Comput. Geom.*, 25:525–568, 2001.
9. V. Choi, P. K. Agarwal, H. Edelsbrunner and J. Rudolph. Local search heuristic for rigid protein docking. In *Proc. 4th Intl. Workshop Alg. Bioinform.*, 2004, to appear.
10. M. L. Connolly. Analytic molecular surface calculation. *J. Appl. Crystallogr.*, 6:548–558, 1983.
11. M. L. Connolly. Shape complementarity at the hemo-globin albl subunit interface. *Biopolymers*, 25:1229–1247, 1986.
12. D. Fischer, S. L. Lin, H. Wolfson and R. Nussinov. A geometry-based suite of molecular docking processes. *J. Mol. Biol.*, 248:459–477, 1995.
13. H. A. Gabb, R. M. Jackson and M. J. Sternberg. Modeling protein docking using shape complementarity, electrostatics and biochemical information. *J. Mol. Biol.*, 272:106–120, 1997.
14. B. B. Goldman and W. T. Wipke. QSD: quadratic shape descriptors. 2. Molecular docking using quadratic shape descriptors (QSDock). *Proteins*, 38:79–94, 2000.
15. W. Humphrey, A. Dalke and K. Schulten. VMD—Visual Molecular Dynamics. *J. Mol. Graphics*, 15:33–38, 1996.
16. E. Katchalski-Katzir, I. Shariv, M. Eisenstein, A. Friesen, C. Afhalo and I. Vakser. Molecular surface recognition: determination of geometric fi t between protein and their ligands by correlation techniques. *Proc. Natl. Acad. Sci. (USA)*, 89:2195–2199, 1992.
17. H. Lenhof. An algorithm for the protein docking problem. In *Bioinformatics: From Nucleic Acids and Proteins to Cell Metabolism*, eds. D. Schomburg and U. Lessel, Wiley, 1995, 125–139.
18. G. Moont and M. J. E. Sternberg. Modelling protein-protein and protein-dna docking. In *Bioinformatics: From Genomes to Drugs*, ed. T. Lengauer, Wiley, 2002, 361–404.
19. B. Sandak, R. Nussinov and H. J. Wolfson. A method for biomolecular structural recognition and docking allowing conformational flexibility. *J. Comput. Biol.*, 5:631–654, 1998.
20. G. R. Smith and M. J. E. Sternberg. Prediction of protein-protein interactions by docking methods. *Curr. Opin. Struct. Bio.*, 12:29–35, 2002.
21. I. A. Vakser. Protein docking for low-resolution structures. *Protein Engin.*, 8:371–377, 1995.