

# A Combinatorial Shape Matching Algorithm for Rigid Protein Docking

Vicky Choi<sup>1,\*</sup> and Navin Goyal<sup>2,\*\*</sup>

<sup>1</sup> Department of Computer Science, Duke University  
vchoi@cs.duke.edu

<sup>2</sup> Department of Computer Science, Rutgers University  
ngoyal@cs.rutgers.edu

**Abstract.** The protein docking problem is to predict the structure of protein-protein complexes from the structures of individual proteins. It is believed that shape complementarity plays a dominant role in protein docking. Recently, it has been shown empirically by Bespamaytnikh et al [4] that the shape complementarity (measured by a score function) is sufficient for the bound protein docking problem, in which proteins are taken directly from the known protein-protein complex and reassembled, treating each protein as a rigid body. In this paper, we study the shape complementarity measured by their score function from a theoretical point of view. We give a combinatorial characterization of the docked configuration achieved by the maximum score. This leads to a simple polynomial time algorithm to find such a configuration. The arrangement of spheres inspired by the combinatorial characterization plays an essential role in an efficient local search heuristic of Choi et al [7] for rigid protein docking. We also show that our general idea can be used to give simple algorithms for some point pattern matching problems in any dimension.

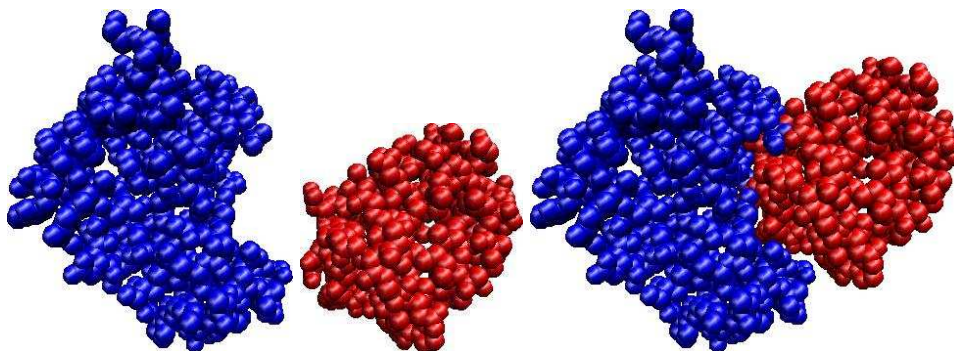
## 1 Introduction

Protein-protein docking problem is one of the current challenges in computational structural biology. In this problem, we are given two protein molecules which are known to dock with each other, the problem is to find the configuration of the docked protein-protein complex. See Figure 1 for an illustration. A recent survey can be found in Mendez et al [14]. The general problem is far from being solved as the physical chemistry of protein-protein interactions is not well understood and the conformational space is high dimensional because proteins undergo conformational changes upon binding. There are no known computationally feasible methods to perform conformational searches during docking. Hence researchers often start with *bound protein docking* problem in which proteins are taken directly from the known protein-protein complex and

---

\* Supported by NSF under grant CCR-00-86013 and a BGT Postdoc Program from Duke University.

\*\* Supported in part by NSF grant CCR-9988526 and in part by NSF Career 0315147.



**Fig. 1.** Given two proteins shown on left, the protein docking problem is to predict the docked configuration as shown on right.

reassembled. In this case the proteins are treated as rigid bodies, i.e., we assume that they do not undergo changes upon interaction as in the real (unbound) case, limiting the dimensionality of search space to six – three for translations and three for rotations. The protein docking problem with the rigidity assumption of proteins (not necessarily taken directly from the protein-protein complex) is known as *rigid protein docking* problem. Thus bound docking is a special case of rigid docking. Efficient rigid docking algorithms are useful because most of the conformational change on docking is believed to be small (or at least this has been and continues to be the assumption of most current research) [15].

For the bound protein docking case, it is believed that geometric complementarity alone is sufficient to solve the problem [15]. This is formalized using the notion of score, i.e., the shape complementarity is measured by certain score functions and the docked configuration is assumed to correspond to the one maximizing the score function. Many different score functions have been used in the literature, e.g. Fast Fourier Transform-based [12], Geometric Hashing-based [9], with varying degrees of success in that the near correctly docked configuration is generally among some number of top score configurations (and not necessarily the highest score configuration). Recently, in Bessamaynikh et al [4], it was shown empirically that the highest score configuration was always the near correctly docked configuration. Their score function (to be defined below; a similar score function was used by Chen et al [6] which was also shown empirically to be quite successful) approximates van der Waals force. This is consistent with the belief that the van der Waals (vdW) forces lead to surface complementarity [5]. In this paper we study this score function from a theoretical point of view. We give a combinatorial characterization of the docked configurations achieved by maximum score. This also leads to a simple polynomial-time algorithm to find such a configuration. The algorithm is not practical due to high running time. However, the arrangement of spheres inspired by the combinatorial characterization plays an essential role in an efficient local search heuristic for rigid protein docking [7].

The rest of this paper is organized as follows. Section 2 contains definitions, in section 3 we give our combinatorial characterization of docking configurations,

in section 4 we use this characterization to obtain a docking algorithm, sections 5 and 6 contain some applications, we conclude with a brief discussion in section 7.

## 2 Problem Statement

A protein molecule consists of a set of atoms, each of which is represented by a ball (a solid sphere) in  $R^3$ . Let  $\mathcal{A} = \{\alpha_1, \dots, \alpha_n\}$ , where  $\alpha_i = (a_i, r_i)$  specifies the  $i$ th atom of protein  $\mathcal{A}$  with center  $a_i \in R^3$  and radius  $r_i$  (which is Van der Waals radius in Å). Denote the corresponding sphere by  $\partial(a_i, r_i)$ . Similarly, protein  $\mathcal{B} = \{\beta_1, \dots, \beta_m\}$  where  $\beta_j = (b_j, s_j)$  specifies the  $j$ th atom of protein  $\mathcal{B}$ . Our proteins are rigid: So the distance between any two atoms in a protein remains constant. There are five typical atom types found in a Protein Data Bank (PDB) file and is shown in the following table:

Atom Type	C	N	O	P	S
Radius in Angstrom	1.548	1.400	1.348	1.880	1.808

Fix the configuration of  $\mathcal{A}$ , we want to move  $\mathcal{B}$  (using rigid motion) towards  $\mathcal{A}$  such that the transformed  $\mathcal{B}$  best complements  $\mathcal{A}$ , where the complementarity is measured by the score defined below. For  $\alpha = (a, r) \in \mathcal{A}$ ,  $\beta = (b, s) \in \mathcal{B}$ ,

$$\begin{aligned} score(\alpha, \beta) &= \begin{cases} 1 & \text{if } r + s \leq \|a - b\| \leq r + s + \lambda, \\ 0 & \text{otherwise.} \end{cases} \\ bump(\alpha, \beta) &= \begin{cases} 1 & \text{if } r + s > \|a - b\|, \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

where  $\|\cdot\|$  is the Euclidean distance and constant  $\lambda = 1.5$  (Å). Define  $score(\mathcal{A}, \mathcal{B}) = \sum_{i=1}^n \sum_{j=1}^m score(\alpha_i, \beta_j)$ ,  $bump(\mathcal{A}, \mathcal{B}) = \sum_{i=1}^n \sum_{j=1}^m bump(\alpha_i, \beta_j)$ . That is,  $score(\mathcal{A}, \mathcal{B})$  counts the number of atom pairs within a distance cutoff  $\lambda$ .  $bump(\mathcal{A}, \mathcal{B})$  counts the number of atom pairs colliding. In [4], it was shown empirically by exhaustive search, which consists of a sampling of rigid motion space and evaluating each rigid motion using the score function, that the configuration corresponding to the maximum score and  $bump \leq \Theta$  (constant  $\Theta = 5$ ) is always near the correct docked configuration (measured by root-mean-square-distance). So our rigid docking problem for the above specific score function is: Fix the configuration of protein  $\mathcal{A}$  in  $R^3$ , the goal is to find a rigid motion  $p$  of protein  $\mathcal{B}$  such that  $score(\mathcal{A}, p(\mathcal{B}))$  is maximized and  $bump(\mathcal{A}, p(\mathcal{B})) \leq \Theta$ , where constant  $\Theta = 5$ .

It is easy to see that the above setting makes sense for  $R^d$  in general – by treating  $\mathcal{A}$  and  $\mathcal{B}$  as sets of balls in  $R^d$ , and our characterization also work in this more general setting.

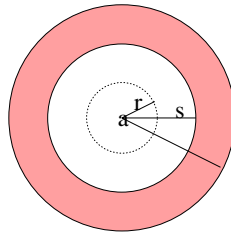
### 3 Combinatorial Characterization

Fix the configuration of  $\mathcal{A}$  in  $R^d$  (we are mostly interested in  $d = 2, 3$ ), let  $\mathcal{C}$  be the space of all rigid motions of  $\mathcal{B}$ . For example, for  $d = 3$ ,  $\mathcal{C} = R^3 \times SO(3)$  ( $R^3$  for translations and  $SO(3)$  for rotations). Each point (rigid motion)  $p \in \mathcal{C}$  corresponds to a configuration of  $\mathcal{B}$ , denoted by  $p(\mathcal{B})$ .

If a rigid body is allowed to move in some restricted way, then the dimension of the space of the motion of that body is known as its *degree of freedom* (DOF). So, for example, a rigid body in  $R^3$  has 6 DOFs. In general, there are  $d(d+1)/2$  DOFs for a rigid body in  $R^d$ . In the Robot Motion Planning community,  $\mathcal{C}$  is called the configuration space of  $\mathcal{B}$  and each point  $p \in \mathcal{C}$  is a configuration of  $\mathcal{B}$ , while  $p(\mathcal{B})$  is called the placement of  $\mathcal{B}$  (and denoted by  $\mathcal{B}(p)$ ). See, e.g., [13] and references therein for background on these notions including DOFs.

We define an equivalence relation on  $\mathcal{C}$ :  $p \in \mathcal{C}$  is equivalent to  $q \in \mathcal{C}$  iff  $score(\alpha_i, p(\beta_j)) = score(\alpha_i, q(\beta_j))$  and  $bump(\alpha_i, p(\beta_j)) = bump(\alpha_i, q(\beta_j))$  for all  $\alpha_i \in \mathcal{A}$ ,  $\beta_j \in \mathcal{B}$ . Call the decomposition of  $\mathcal{C}$  by the equivalence classes (called *cells*) the *arrangement*  $\Pi(\mathcal{A}, \mathcal{B})$  of  $\mathcal{C}$ . Note that our cells are not necessarily connected. By the definition of a cell, the score and bump of all configurations in a cell are the same. Hence we can talk about the score and bump of a cell.

For ease of exposition, we assume henceforth that the balls in  $\mathcal{B}$  all have the same radius  $s$  (generalization to include the case where radii may be different easily follows). Observe that for two balls  $(a, r) \in \mathcal{A}$  and  $(b, s) \in \mathcal{B}$ ,  $score((a, r), (b, s)) = 1 \iff r + s \leq \|a - b\| \leq r + s + \lambda \iff b$  lies on the annulus formed by spheres  $\partial(a, r + s), \partial(a, r + s + \lambda)$  as shown in Figure 2. Let  $G_s(\mathcal{A}) = \{\partial(a_i, r_i + s), \partial(a_i, r_i + s + \lambda) : \forall (a_i, r_i) \in \mathcal{A}\}$  the set of two enlarged spheres corresponding to each ball in  $\mathcal{A}$ .



**Fig. 2.** The three circles in the figure have radii  $r, r + s, r + s + \lambda$ .  $score((a, r), (b, s)) = 1 \iff r + s \leq \|a - b\| \leq r + s + \lambda \iff b$  lies on the shaded region.

The assumption above that the balls in  $\mathcal{B}$  all have the same radius affords a somewhat more intuitive picture of the arrangement  $\Pi(\mathcal{A}, \mathcal{B})$ : A cell in  $\Pi(\mathcal{A}, \mathcal{B})$  is defined by assigning to each center  $b_i$  of balls in  $\mathcal{B}$  a cell  $\gamma_i$  in the arrangement of spheres in  $G_s(\mathcal{A})$  (this arrangement should not be confused with  $\Pi(\mathcal{A}, \mathcal{B})$ ) and consists of  $p \in \mathcal{C}$  such that  $p(b_i) \in \gamma_i$  for all  $i$ . This is the way we will think about  $\Pi(\mathcal{A}, \mathcal{B})$ .

We have to deal with one technical issue. The boundary of a cell can be complicated in the sense that it may be open at some places and closed at others. Our goal will be to find a representative point for each cell, but the procedure that finds this point guarantees only that the point lies in the closure of the cell. This causes the problem that a point may belong to more than one such closures and cannot be used to uniquely characterize a cell. This problem can be resolved by making two copies of each sphere in  $G_s(\mathcal{A})$ , slightly enlarging one of them and shrinking the other, and appropriately defining the cells. Under this new definition cells are closed sets and their boundaries are disjoint. We omit the tedious details. To simplify matters, in this paper we will just deal with the closures of cells.

We need some more definitions. For given  $\mathcal{A}$  and  $\mathcal{B}$  and  $p \in \mathcal{C}$ , a *contact-constraint* on configuration  $p(\mathcal{B})$  is specified by the center  $b$  of a ball  $(b, s) \in \mathcal{B}$ , a sphere  $\omega$  of  $G_s(\mathcal{A})$  and the condition that  $p(b)$  lie on  $\omega$ . Also, we slightly abuse the definitions by saying a contact-constraint is satisfied by a rigid motion  $p$  if the corresponding configuration  $p(\mathcal{B})$  satisfies the contact-constraint.

We assume throughout that  $\mathcal{A}$  and  $\mathcal{B}$  are in *general position*. We say that  $\mathcal{A}$  and  $\mathcal{B}$  are in general position, if no set of more than  $d(d+1)/2$  contact-constraints is satisfiable by any rigid motion in  $\mathcal{C}$ . This is a natural assumption, as we expect one constraint to reduce DOF by one, and so if we have more constraints than available DOF, then we can no longer satisfy all of them. This assumption can be ascertained by applying small perturbation to the positions of the centers of balls.

**Theorem 1.** *Let  $\mathcal{A}$  and  $\mathcal{B}$  be two sets of balls in  $R^d$  in general position, and let  $\Pi(\mathcal{A}, \mathcal{B})$  be defined as above. Then for any  $F \in \Pi(\mathcal{A}, \mathcal{B})$ , there is a set  $S$  of  $k \leq d(d+1)/2$  contact-constraints such that the rigid motions corresponding to the configurations satisfying them form  $2^{O(d^3)}$  connected components, and at least one of the components is contained in the closure of  $F$ .*

*Proof.* Fix a cell  $F \in \Pi(\mathcal{A}, \mathcal{B})$ . Let  $p$  be a point in  $F$  such that  $p(\mathcal{B})$  satisfies the maximal number of contact-constraints. Denote by  $S$  the set of contact-constraints; by  $\text{Center}_S$  the set of centers of balls in  $\mathcal{B}$  involved in contact-constraints in  $S$ ; and  $\text{Spheres}_S$  the set of spheres in  $G_s(\mathcal{A})$  involved in  $S$ . By our general position assumption we have  $|S| \leq d(d+1)/2$  and thus  $|\text{Center}_S| \leq d(d+1)/2$  and  $|\text{Sphere}_S| \leq d(d+1)/2$ .

Each contact-constraint in  $S$  amounts to an equation of degree 2 of a sphere in  $\text{Sphere}_S$  in the variables corresponding to the coordinates of some center in  $\text{Center}_S$ . Configurations satisfying the contact-constraints in  $S$  can be found by solving a system of equations including contact-constraints equation and centers distance equations (because  $\mathcal{B}$  is rigid). Denote by  $\text{Solution}_S$  the set of solutions to the system of these equations.

Now the theorem of Oleinik-Petrovsky/Thom/Milnor (see, e.g. [3]) from real algebraic geometry asserts that the number of connected components of the set of solutions to a system of equations of degree  $\leq D$  in  $v$  variables is bounded above by  $D(2D-1)^{v-1}$ . In our case,  $D = 2$  and  $v \leq d \cdot d(d+1)/2$  (the number

of centers whose coordinates appear in the equations is  $\leq d(d+1)/2$ , and each has  $d$  coordinates, hence the bound on the number of variables). So the number of connected components in  $\text{Solution}_S$  is  $\leq 2^{O(d^3)}$ .

Let the set of rigid motions such that the corresponding configurations satisfy the contact-constraints in  $S$  be  $\text{Motion}_S$ . We claim that the number of connected components in  $\text{Motion}_S$  is the same as that in  $\text{Solution}_S$ . Elements in  $\text{Solution}_S$  provide values of the coordinates of the centers in  $\text{Center}_S$ , and from these we can obtain the set of rigid motions whose corresponding configurations satisfy the constraints. Given the coordinates of the centers in  $\text{Center}_S$  we may have two cases: (1) These are sufficient to determine a unique rigid motion. (2) We still have some degree of freedom left and we can find a set of rigid motions on each of the corresponding configuration the coordinates of the centers take on the given values; call this set of rigid motions the *image* of the given element in  $\text{Center}_S$ . It is clear that in the first case, the number of connected components in  $\text{Motion}_S$  is the same as that in  $\text{Solution}_S$ . For the second case, note that given some (valid) values of coordinates of centers in  $\text{Center}_S$ , its image is connected. E.g, if we fix two points of a rigid body in  $R^3$  then the remaining motion is just the rotation around the line connecting the two fixed points; these rigid motions form a connected subset. Therefore, the image of any connected component of  $\text{Solution}_S$  in  $\text{Motion}_S$  is also connected. So the number of connected components in  $\text{Solution}_S$  is the same as that in  $\text{Motion}_S$ .

Denote the component of  $\text{Motion}_S$  containing  $p$  by  $P$ . We claim that  $P$  is contained in the closure of  $F$ . Suppose not, i.e. there exists  $p' \in P \setminus \text{cl}(F)$  and there is a path connecting  $p$  and  $p'$  within  $P$ . Consider a point  $q$  where this path crosses a component of  $F$ . The reason the path gets out of the closure of  $F$  is that some center  $b$  in  $\mathcal{B}$  gets out of its cell in the arrangement of  $G_s(\mathcal{A})$  by crossing some sphere  $\omega$ . The contact-constraint that  $b$  lie on  $\omega$  is not in  $S$ , because it is not always satisfied in  $P$ , in particular on some points of the above path. So at point  $q$ , which is in the closure of  $F$ , we have contact-constraints in  $S$  and one extra contact-constraint satisfied. But this contradicts the maximality assumption. Hence  $P$  must be contained in the closure of  $F$ .  $\square$

The theorem above says that any cell in  $\Pi(\mathcal{A}, \mathcal{B})$  can be almost uniquely characterized by a set of contact-constraints in the sense that at least one of the small number of connected components formed by the configurations satisfying these contact-constraints is in the given cell. While the above proof may appear somewhat abstract, the principle behind it is quite intuitive, which is illustrated below by an example in  $R^2$ .

**Example in  $R^2$ .** As mentioned above, for each cell  $F \in \Pi(\mathcal{A}, \mathcal{B})$ , we want to find a point which has a simple description. This point will be found by satisfying as many contacts between centers of the balls in  $\mathcal{B}$  and circles in  $G_s(\mathcal{A})$  as possible while staying inside  $\text{cl}(F)$ , so that these (almost) uniquely identify the cell. Let  $p \in F$  and suppose that  $p(\mathcal{B})$  does not satisfy any contact-constraints. Choose a direction to translate  $p(\mathcal{B})$  so that the center  $b_1$  of some ball of  $\mathcal{B}$  hits a circle  $\omega_1$  from  $G_s(\mathcal{A})$ . This is always possible (e.g. the direction of the closest ball pairs). This translation corresponds to a new point  $p'$  in  $\mathcal{C}$

and it is not difficult to see that  $p' \in cl(F)$ . Next rotate  $p'(\mathcal{B})$  about  $p'(b_1)$ , let  $p'' \in \mathcal{C}$  be the corresponding point in the configuration space (if any) such that the center  $b_2$  of some ball in  $\mathcal{B}$  hits a circle  $\omega_2$  from  $G_s(\mathcal{A})$ . (Note: in this example, we only consider rotating  $p'(\mathcal{B})$  around  $p'(b_1)$  and we might not find a second-contact, but it is possible that the second-contact does exist by a rigid motion which moves  $p'(b_1)$  to a different position on  $\omega_1$ , and then rotates  $p'(\mathcal{B})$ . Also, we only require  $(b_1, \omega_1) \neq (b_2, \omega_2)$ , and it is possible that  $b_1 = b_2$  (in this case the constraint is that  $b_1 (= b_2)$  is at an intersection point of  $\omega_1$  and  $\omega_2$ ), and similarly it is possible  $\omega_1 = \omega_2$  with distinct  $b_1, b_2$ .) Since  $\mathcal{B}$  has 3 DOFs in  $R^2$ , and each condition of the above kind reduces one DOF (by a general position assumption), hence  $\mathcal{B}$  has one more DOF to move while preserving the 2 contact-constraints. One of the following three events happens in the course of this rigid motion, each giving a set of contact-constraints. (1) One of the two points  $b_1$  and  $b_2$  hits a vertex (an intersection point of two circles) in  $G_s(\mathcal{A})$ . (2) A new point  $b_3$  in  $\mathcal{B}$  hits a circle in  $G_s(\mathcal{A})$ . (3) None of the above. In the first two cases we only have a constant number of solutions, while in the last case we get a curve in the rigid motion space.

For general  $d$ , let  $h = \frac{1}{2}d(d+1)$ . Each cell in  $\Pi(\mathcal{A}, \mathcal{B})$  is characterized by a system of contact-constraints as in Theorem 1. Each such system of contact-constraints is specified by at most  $h$  spheres in  $\mathcal{A}$  and  $h$  spheres in  $\mathcal{B}$ . There are a total  $O(m^h n^h)$  possible systems of contact-constraints. From Theorem 1, each cell contains at least one connected component of some system of contact-constraints. Since each such system has at most  $2^{O(d^3)}$  components, we have

**Corollary 1.** *In  $R^d$ ,  $\Pi(\mathcal{A}, \mathcal{B})$  has at most  $O(2^{O(d^3)} m^h n^h)$  cells, where  $h = \frac{1}{2}d(d+1)$ .*

## 4 The Algorithm

In this section we describe our algorithm for finding a maximum score configuration with bump less than a given threshold. We content ourselves with a high-level description of the algorithm, as in its present form it is mainly of theoretical interest.

Basic idea is simple: Enumerate all systems of contact-constraints which can arise in Theorem 1. For each system of (at most  $d(d+1)/2$ ) contact-constraints, it corresponds to a set of equations of degree 2. Then we generate a point (the coordinates of the centers) in each connected component of the solutions to the system of equations. The generation can be done by one of the many available algorithms in real algebraic geometry, see for example Chapter 11.6 of [3], which takes time, in our situation,  $2^{O(d^3)}$ . We then recover a rigid motion for each solution, which corresponds to a point in each connected component of the configurations satisfying these contact-constraints. This can be done in time polynomial in  $d$ . Finally, compute the score and bump for each configuration. Having examined all systems, take the one with the maximum score and bump below the threshold. By theorem 1, this procedure will generate at least one point in each cell in  $\Pi(\mathcal{A}, \mathcal{B})$ , and thus will find the desired optimal solution.

To analyze the running time, note that the number of constraint systems that we examine is  $O(m^h n^h)$ , with  $h = \frac{1}{2}d(d+1)$ , and for each such system we spend  $2^{O(d^3)}$  time generating at most  $2^{O(d^3)}$  points. For each of the corresponding configurations  $p(\mathcal{B})$ , we compute  $\text{score}(\mathcal{A}, p(\mathcal{B}))$  and  $\text{bump}(\mathcal{A}, p(\mathcal{B}))$ . Doing this in the obvious way takes time  $O(mn)$ . Hence, the total time taken is  $O(2^{O(d^3)} m^{h+1} n^{h+1})$ .

In the following we will improve on this algorithm in  $R^3$  by making use of a property satisfied by protein molecules. The following density property of the spheres in a protein (and also in  $G_s(\mathcal{A})$ ) was given by [10]. It says that the atoms in a protein overlap in a somewhat sparse fashion and do not crowd together. For the sake of completeness, we include the easy proof; bound here is better than in [10].

**Theorem 2.** *Let  $\mathcal{A} = \{\alpha_i = (a_i, r_i) : i = 1, \dots, n\}$  be a set of  $n$  spheres in  $R^3$ . Suppose there is a  $\delta > 0$  such that  $\|a_i - a_j\| > \delta$ ,  $1 \leq i, j \leq n$  ( $i \neq j$ ). Then for any  $1 \leq i \leq n$ ,  $|\{\alpha_j \in \mathcal{A} : \alpha_i \cap \alpha_j \neq \emptyset\}| \leq (4 \frac{r_{max}}{\delta} + 1)^3$ , where  $r_{max} = \max_{j=1..n} r_j$ .*

**Proof.** Since for any  $1 \leq i, j \leq n$  ( $i \neq j$ ),  $\|a_i - a_j\| > \delta$ , we have  $(a_i, \frac{\delta}{2}) \cap (a_j, \frac{\delta}{2}) = \emptyset$ . Consider  $\alpha_i \in \mathcal{A}$ , for any  $\alpha_j \in \mathcal{A}$  with  $\alpha_i \cap \alpha_j \neq \emptyset$ ,  $a_j \in (a_i, r_i + r_{max})$ . Thus,  $(a_j, \frac{\delta}{2}) \subset (a_i, r_i + r_{max} + \frac{\delta}{2})$ . By volume argument,  $|\{\alpha_j \in \mathcal{A} : \alpha_i \cap \alpha_j \neq \emptyset\}| \leq (\frac{\alpha_i \cdot r + r_{max} + \frac{\delta}{2}}{\frac{\delta}{2}})^3 \leq (4 \frac{r_{max}}{\delta} + 1)^3$ .  $\square$

For the set of spheres in a protein (or  $G_s(\mathcal{A})$ ),  $\frac{r_{max}}{\delta}$  is a constant. That is, each sphere with radius at most  $r_{max}$  intersects only constantly many spheres in the set. With this property, we can then use data structure of [10] to compute the score and bump of each corresponding configuration in  $O(m)$  time (by preprocessing a data structure using  $O(n)$  space in  $O(n)$  expected time for the  $2n$  spheres of  $G_s(\mathcal{A})$ ), and output a configuration with the maximum score and bump  $\leq \Theta$ . It is possible that there are more than one such configurations. For the application to rigid protein docking this is not problematic in the light of empirical results of Bepamaynikh et al [4], which show that such configurations are all very close to each other.

Therefore, we have

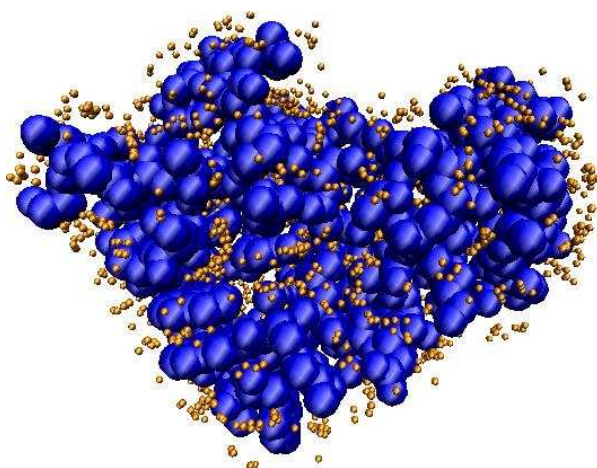
**Theorem 3.** *Let  $\mathcal{A}, \mathcal{B}$  be two proteins (in general position) in  $R^3$  and the score function defined as above. The docked configuration achieved by the maximum score and bump  $\leq \Theta$  can be computed in  $O(m^6 n^6 \min\{m, n\})$  time.*

This algorithm is not practical due to the high running time. However, the arrangement of the spheres in  $G_s(\mathcal{A})$  used in the algorithm has important applications described in the next section.

## 5 Applications of the Arrangement of Spheres

In this section, we describe two immediate applications of the arrangement of spheres in  $G_s(\mathcal{A})$ . It provides a useful representation of the complement of a

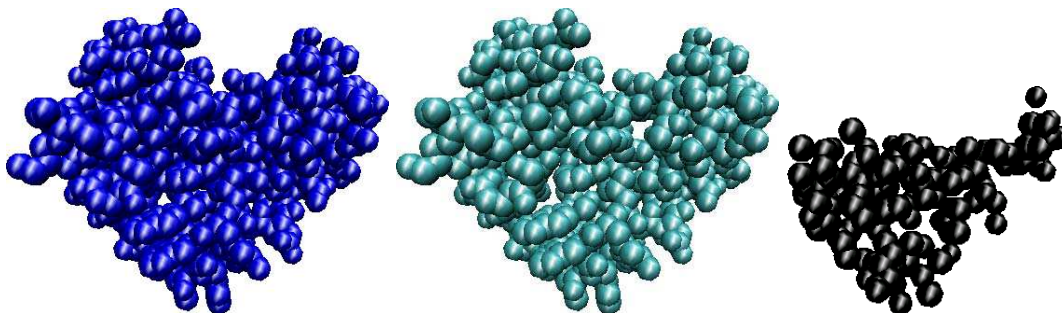
protein; and second, it gives a simple way to identify surface atoms. Recall that the arrangement of spheres in  $G_s(\mathcal{A})$  decomposes the space into regions (might be disconnected) with score and bump constant in a region. By the above theorem on the density property of spheres in  $G_s(\mathcal{A})$ , each sphere intersects with only constantly many other spheres and hence there are only linearly many vertices in the arrangement of spheres in  $G_s(\mathcal{A})$ . Again, by the same data structure as in [10], we can compute the vertex set of the arrangement in  $O(n \log n)$  time implemented by binary search tree or  $O(n)$  expected time using randomized perfect hashing, both with  $O(n)$  space. Also, we can compute the score and bump of each vertex, and associate each vertex with the atoms which contribute to the score, in extra  $O(1)$  time. In particular, the bump-free vertices (i.e. with bump = 0) play an important role in an efficient local search algorithm [7] in that they describe discrete positions of the complement of the protein measured by the score function. See Figure 3 for an example of these vertices.



**Fig. 3.** The bump-free vertices of protein A are shown by dots.

Another application of the arrangement of spheres is an easy and useful way to identify “surface” atoms from “interior” atoms. The set of surface atoms is just the set of atoms associated with the bump-free vertices. It is possible that some of these atoms actually lie inside the molecule, and the bump-free vertices associated to them are bump-free because of holes in the molecule. For applications, we are generally not interested in such surface atoms. To identify the set of surface atoms which are not inside the molecule we define a graph on the bump-free vertices such that two vertices are adjacent if they are close. Now the vertices corresponding to the outer surface form a connected component in this graph and this can be easily identified. See Figure 4 for an example.

Our surface atom definition is tailored for use in rigid docking. To solve rigid docking problem one can restrict attention to the configurations for which surface atoms contribute to the interaction; the configurations for which interior atoms contribute can be safely ignored. While the formal molecular surfaces such as



**Fig. 4.** The original protein (left) = surface atoms only (middle) + interior atoms (right).

solvent accessible surface ([10]), which involves computing the surface area, can also be used to identify the surface atoms but they are much complicated to compute [10].

## 6 Application of Our Framework to Point Pattern Matching

Recently, Agarwal et al [1] considered using partial shape matching under Hausdorff distance to measure the shape complementarity under translation. See the references there for related literature on point pattern matching. An observation somewhat similar to ours has been made by Alt et al [2] in solving a congruence problem in  $R^2$ . In the general  $d$ -dimensional version of this problem we are given two sets  $\mathcal{A}$  and  $\mathcal{B}$  in  $R^d$  of  $n$  points each, and a number  $\delta$ . The goal is to decide if there exists a one-one matching between the points in  $\mathcal{A}$  and  $\mathcal{B}$  and a rigid motion  $p$  of  $\mathcal{B}$  so that the maximum of the distance between the matched pairs in  $\mathcal{A}$  and  $p(\mathcal{B})$  is at most  $\delta$ . The algorithm given in Alt et al [2] for  $d = 2$ , which considered enumerates sets of constraints of size 2, has  $O(n^8)$  time. They are not able to generalize their algorithm to higher dimensions. Here we show how to use our framework and results of [8] to get an algorithm with time  $\tilde{O}(n^{7.5})$  in  $R^2$ ; an  $O(n^{13+5/6+\epsilon})$  time algorithm in  $R^3$  (for any positive  $\epsilon$ ); and an  $O(n^{d(d+1)+2.5})$  time algorithm in  $R^d$  for  $d > 3$ .

Similarly to the algorithm in the Section 4, the present algorithm has two parts: (1) Enumeration, and (2) testing. Algorithm enumerates a set of constraints, and for each such set it checks if the configurations satisfying these constraints provide a positive answer. Enumeration part is very similar to our previous algorithm; the testing part is done using bipartite bottleneck matching.

We first describe the enumeration part which works for general  $d$ . We define the notion of a constraint similar to the earlier one. Assume that  $\mathcal{A}$  is fixed. A constraint on the configuration of  $\mathcal{B}$  specifies a sphere with center in  $\mathcal{A}$  and radius  $\delta$ , and a point in  $\mathcal{B}$ , and requires that this point lie on the given sphere. Denote by  $G(\mathcal{A})$  the arrangement of  $R^d$  induced by spheres of radius  $\delta$  with centers in  $\mathcal{A}$ . We can decompose the configuration space of  $\mathcal{B}$  into equivalence classes: Two

configurations  $p(\mathcal{B})$  and  $p'(\mathcal{B})$  are equivalent if for all points  $b \in \mathcal{B}$ ,  $p(b)$  and  $p'(b)$  lie in the same cell of  $G(\mathcal{A})$ . If the answer to the problem is yes then there is a one-one correspondence between  $\mathcal{A}$  and  $\mathcal{B}$ , and there is nonempty equivalence class such that the configurations  $p(\mathcal{B})$  in it satisfy:  $\forall b \in \mathcal{B}$ ,  $p(b)$  is in the sphere of radius  $\delta$  around the point of  $\mathcal{A}$  matched to  $b$ . Denote this class by  $M$ . By the arguments similar to those in the proof of Theorem 1, there is a set of at most  $d(d+1)/2$  constraints with the following property. At least one connected component of the configurations satisfying these constraints is contained in  $M$ , and there are at most  $2^{O(d^3)}$  connected components. The enumeration part is now clear: We consider all possible sets of  $\leq d(d+1)/2$  constraints; for each such set we generate a point in each connected component of the configurations satisfying the constraints in the set.

The testing part checks if a given configuration  $p(\mathcal{B})$  solves the problem. We can do this using the bipartite bottleneck matching: Given two sets of equal cardinality, the problem is to find a one-one matching between them such that the maximum distance (cost) between the matched points is minimized. Clearly, for our problem, if we have a configuration  $p(\mathcal{B})$  which is in  $M$ , then the bottleneck matching between  $\mathcal{A}$  and  $p(\mathcal{B})$  will have cost at most  $\delta$ . And also, if the answer to our decision problem is no, then for all configurations  $p(\mathcal{B})$ , bottleneck matching between  $\mathcal{A}$  and  $p(\mathcal{B})$  will have cost more than  $\delta$ . In general, we can construct a bipartite graph with points in  $\mathcal{A}$  on one side, and points in  $p(\mathcal{B})$  on the other side, and two points on different sides are connected if their distance is  $\leq \delta$ . Then check if the bipartite graph has a perfect matching. This can be done by the algorithm of Hopcroft and Karp [11] in time  $O(v^{2.5})$  where  $v$  is the number of vertices. There is a bottleneck matching between  $\mathcal{A}$  and  $p(\mathcal{B})$  with cost at most  $\delta$  iff the bipartite graph has a perfect matching. This gives an algorithm of  $O(n^{d(d+1)+2.5})$  time for general  $d$ . For  $d = 2, 3$ , the bipartite bottleneck matching problem can be solved more efficiently in  $\tilde{O}(n^{1.5})$  time in  $R^2$ , and  $O(n^{11/6+\epsilon})$  time in  $R^3$  (for any positive  $\epsilon$ ) by Efrat et al [8]. Thus, we have an algorithm of  $\tilde{O}(n^{7.5})$  for  $d = 2$  and  $O(n^{13+5/6+\epsilon})$  for  $d = 3$ .

## 7 Discussion

We have given a simple algorithm for rigid protein docking under the score function of [4]. This algorithm is not practical due to high running time and numerical issues. A natural question is if it can be substantially improved. Note that we did not use the protein density property in the analysis of the arrangement. We believe that by using this property and by pruning the search space, e.g. by excluding the configurations with large bumps from the enumeration, we can get faster algorithms.

## Acknowledgment

We thank Saugata Basu for answering our question regarding real algebraic geometry. We also thank Pankaj K. Agarwal, Herbert Edelsbrunner and Raimund Seidel for comments.

## References

1. P. Agarwal, S. Har-Peled, M. Sharir and Y. Wang. Hausdorff distance under translation for points, disks and balls. *Proc. 19th ACM Symp. on Computational Geometry* (2003), 282-291.
2. H. Alt, K. Mehlhorn, H. Wagnen, E. Welzl. Congruence, similarity, and symmetries of geometric objects. *Discrete Comput. Geom.* 3 (1988), no. 3, 237–256.
3. S. Basu, R. Pollack, M.-F. Roy. Algorithms in Real Algebraic Geometry. *Springer* 2003.
4. S. Bspamaytnikh, V. Choi, H. Edelsbrunner and J. Rudolph. Accurate Bound Protein Docking by Shape Complementarity Alone. *Technical Report, Department of Computer Science, Duke University, 2003.*
5. C.J. Camacho and S. Vajda , Protein docking along smooth association pathways. *Proc Natl Acad Sci USA* 98 (2001), pp. 10636-10641.
6. R. Chen, Z. Weng. A novel shape complementarity scoring function for protein-protein docking. *Proteins: Structure, Function, and Genetics* 51:3, 2003, 397-408.
7. V. Choi, P. K. Agarwal J. Rudolph and H. Edelsbrunner. Local Search Heuristic for Rigid Protein Docking. *To be submitted.*
8. A. Efrat, A. Itai and M. J. Katz. Geometry helps in bottleneck matching and related problems. *Algorithmica* 31 (2001), no. 1, 1–28.
9. D. Fischer, S.L. Lin, H.J. Wolfson and R. Nussinov. A suite of molecular docking processes. *J Mol Biol* 248 (1995), pp. 459-477.
10. D. Halperin and M.H. Overmars. Spheres, molecules, and hidden surface removal. *Computational Geometry: Theory and Applications* 11 (2), 1998, 83–102.
11. J. Hopcroft and R.M.Karp. An  $n^{5/2}$  algorithm for maximum matchings in bipartite graphs. *SIAM J. Computing*, 2 (1973), 225-231.
12. E. Katchalski-Katzir, I. Shariv, M. Eisenstein, A.A. Friesem, C. Aflalo and I.A. Vakser. Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques. *Proc Natl Acad Sci USA* 89 (1992), pp. 2195-2199.
13. J.-C. Latombe. Robot Motion Planning. Kluwer Academic, Boston, 1991.
14. R. Mendez, R. Leplae, L. D. Maria, S. J. Wodak. Assessment of blind predictions of protein-protein interactions: Current status of docking methods. *Proteins: Structure, Function, and Genetics* 52:1, 2003, 51-67.
15. G.R. Smith and M. JE. Sternberg. Predictions of Protein-Protein Interactions by Docking Methods. *Current Opinion in Structural Biology.* 12:28-35, 2002.