CPS260/BGT204.1 Algorithms in Computational Biology **November 18, 2003**

Lecture 23: Shape Analysis

Lecturer: Pankaj K. Agarwal *Scribe: Pankaj K. Agarwal*

23.1 Dual Complex

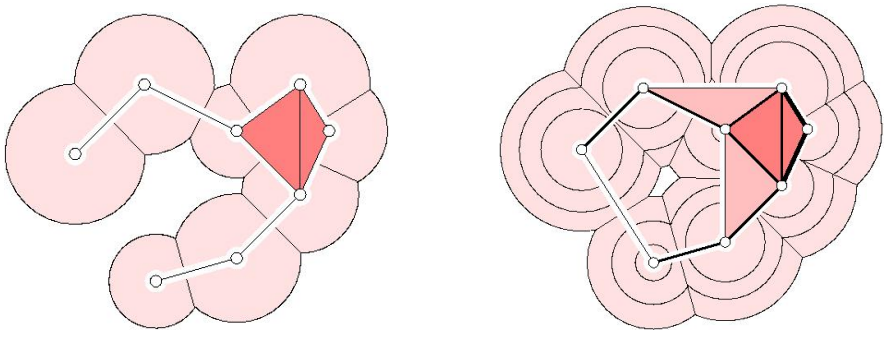
In the previous lecture, we discussed the relationship between the power diagram $\Pi(\mathbb{P})$ of a set $\mathbb{P} = \{B_1, \ldots, B_n\}$ of balls and its Delaunay triangulation $DT(\mathbb{P})$. Similarly, we can define the dual structure of the union U of the balls in $\mathbb{P}$. The power diagram $\Pi(\mathbb{P})$ partitions U into convex cells

$$U \cap \Pi(B_i) = B_i \cap \Pi(B_i).$$

Let $P = \{z_1, z_2, \ldots, z_n\}$ be the set of centers of the balls in $\mathbb{P}$. The *dual complex*, K, is defined as,

$$K = \{\text{conv}(R) \mid \bigcap_{z_i \in R} (\Pi(B_i) \cap B_i) \neq \emptyset, \forall R \subseteq P\}$$

where $\text{conv}(R)$ is the convex hull of R. $\text{conv}(R)$ can be a point, an edge or a triangle in $\mathbb{R}^2$ and a point, an edge, a triangle or a tetrahedron in $\mathbb{R}^3$. K essentially describes the non-empty intersections among the cells contained in $\bigcup B_i$. The underlying space of a dual complex is known as the *dual shape* of $\bigcup B_i$. We illustrates the dual complex of a union of disks in Figure 23.1. The edges correspond to the pairwise intersections of $B_i \cap \Pi(B_i)$, while the two triangles represent the triplet intersections.



(i) (ii)

Figure 23.1: (i) A dual complex. (ii) The filtration of a Delaunay triangulation. The figure is taken from [1].

At time τ, $-\infty < \tau < \infty$, we grow the radius r_i of ball B_i to $\sqrt{r_i^2 + \tau}$. As time progresses from $-\infty$ to ∞, the dual complex undergoes a series of monotonic transformations from ϕ to the Delaunay triangulation D,

which is defined by the set of centers of balls in $\bigcup B$. Notice that before time $-r_i^2$, the radius of ball B_i is imaginary. Since there are only finite number of elements in the Delaunay triangulation, the number of dual complexes that arise must be finite as well. Now let α equal to $\sqrt{\tau}$. Using α as the index of time, we get a set of balls B_α, each with radius $\sqrt{r_i^2 + \alpha^2}$, and a corresponding dual complex K_α at time α. K_α is referred to as the α-*complex*, and the underlying space as α-*shape*. The reason for choosing α in stead of τ as the time index is that the radius of a ball $B_i(z_i, r_i = 0)$ is α at time α. The sequence of α-complexes is referred to as the *dual-complex* (ii) of the Delaunay triangulation, $\phi = K^1 \subset K^2 \subset \cdots \subset K^m = D$. This is illustrated in Figure 23.1 (ii). The first dual complex is the set of vertices plus the two thick edges. There is no triangle in the first dual complex. The edges become thinner and lighter, and the color of the triangles changes from dark gray to light gray as the dual complex grows.

Voids and pockets. The dual compelx is a powerful tool to capture topological features of bio-molecules because the topology of $K(\mathbb{P})$ is the same as that of $\mathbb{P}$. For example, voids and pockets of $\mathbb{P}$ can be defined and computed using $K(\mathbb{P})$. A *void* of $\mathbb{P}$ is a bounded connected component of $\mathbb{R}^3 \setminus \bigcup_{i=1}^n B_i$. For each void in $\mathbb{P}$, there is a corresponding void in $K(\mathbb{P})$, and a void in $K(\mathbb{P})$ is much more amiable to computation. (Similarly, a tunnel can also be easily captured and computed using the $K(\mathbb{P})$.)

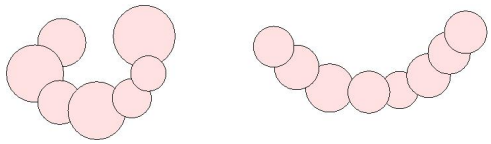


Figure 23.2: Two distinct types of pockets.

The motivation behind the study of the pocket is because it often defines a docking site for the modeled protein. A pocket can be roughly viewed as a dent on the surface of a shape. However, the precise definition of a *pocket* is more complicated. We give one type of definition of the pocket by viewing it as a relaxed version of a void. We now explain this by distinguishing two possible types of pockets in Figure 23.2. Our way to distinguish the left pocket from the right pocket exploits the growth model previously discussed. If we grow the balls consisting $\bigcup B$ uniformly, the left pocket will produce a void in the life time of the filtration while the right pocket will not. Thus the pocket feature can be precisely defined and computed with the growth model. Similarly, a *groove* on the surface of a molecule can be transformed into a $tunnel$, and be captured by our model.

The model we discussed so far represents the topological features of a molecule in a hierarchical way. For example, by growing the union of balls, we are able to distinguish big voids or tunnels from smaller ones. This is also proved to be convenient for the task of topological simplification.

23.2 Shape Matching

Given two shapes A and B, the *shape matching* problem measures how much they resemble each other. In biological study, the problem is motivated by applications such as protein shape classification and protein docking. While protein shape classification measures the similarity between two shapes, protein docking examines the complementarity between two shapes. Shape matching is usually carried out as a two stage

process. The first stage is referred to as *initial matching stage*. During this stage, two shapes are roughly matched based on geometric characteristics. The second stage is known as *local improvement stage*. During this stage, the matching is further refined according to factors such as energy properties.

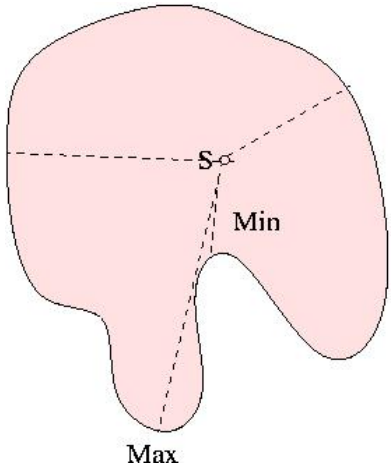


Figure 23.3: Critical points.

Since it is hard to match two surfaces, a common practice is to identify geometric and topological features on each surafce and match them. One common approach to characterize the geometrical feature of a shape uses the so-called *critical points*. For example, to characterize the shape of a closed curve C in the plane shown in Figure 23.3, we can set a fixed view point S in the area bounded by C. Let $F : C \rightarrow R$ be a continuous function measuring the distance from S to a point $x \in C$ as x moves continuously along C. The maximum and minimum points of F are referred to as the *critical points* of C. Two shapes can be roughly distinguished by comparing their critical points. In 3D teh critical points consist of minima, maxima, and saddle points.

We give a flavor of the research in shape matching by discussing the problem of point sets matching. Given a set of red points R, and a set of green points G, the problem is to determine the similarity between R and G. As a measure of *similarity*, we use a metric called *Hausdorff distance*. Given two sets of points R and G, Hausdorff distance $H(R, G)$ is defined as:

$$\max \quad \{\max_{g \in G} \min_{r \in R} d(g, r), \quad \max_{r \in R} \min_{g \in G} d(r, g)\}$$

For two fixed point sets R and G, the Hausdorff distance can be relatively efficiently computed. However, it is more interesting to assume that R or G is not fixed but can be moved under a set of rigid transformations I, such as *translations* and *rotations*. We can fix one set, say R, and move G around to match it as well as possible with R, and then determine the similarity of the resulting shapes. More precisely, we state the problem of shape matching under rigid motion as follows. Let μ be the cost function measuring the similarity of R and G. The smaller the value of μ, the more similar the two shapes R and G. We would like to capture $\min_{i \in I} \mu(R, i(G))$, i.e., to find the best transformation such that R and $i(G)$ are most similar with respect to μ. Here $i(G)$ refers to the copy of G after applying transformation i to it. A widely used algorithm to solve the above problem is *Iterative Closest Point* algorithm, or ICP algorithm for short. Define a mapping from R

to G as $\pi : R \ni r_j \to \pi(r_j) \in G$. Let I be the allowed transformation space. Let

$$\mu(\pi, i) = \sum_{r_j \in R} d^2(r_j, i(\pi(r_j))$$

be the cost function to be minimized. Let $\tau > 0$ be the threshold tolerance value. We can state the ICP algorithm as follows:

1. Compute the best mapping $\pi : R \to G$ so that $f(\pi, i)$ is minimized
2. Fix π, compute the best rigid motion $i \in I$ that minimizes $f(\pi, i)$
3. Update R to be $i(R)$
4. Terminate the iteration if the change in mean square error falls below the tolerance threshold τ. Otherwise, repeat

For the translation only case, a quick way to match R and G is to align their *centroids*. A centroid of a point set is the *center of mass* of the points in the set. Points such as the centroids belong to a more general category known as *reference points*. Reference points are widely used to derive approximate algorithms for shape matching.

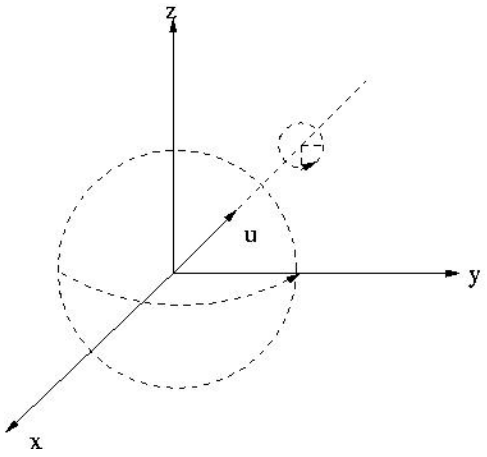


Figure 23.4: Rotation in $\mathbb{R}^3$.

Rotations in 3D and quaternions. Shape matching under both translation and rotation, i.e., rigid motion, is considerably more complicated than shape matching under only translation. The degree of freedom for translation only is three, while the degree of freedom for rigid motion is six. Rotation in three-dimensional space can be specified by a rotational axis vector $(u_x, u_y, u_z) \in \mathbb{R}^3$ and a rotational angle $\theta \in \mathbb{R}$, as shown in Figure 23.4. However, a better way to represent rotation is to use *quaternions*. A quaternion q has the form

$$q = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$$

where $q_o, q_1, q_2, q_3 \in \mathbb{R}$ and $\mathbf{i}, \mathbf{j}, \mathbf{k}$ satisfy

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1.$$

This in turn implies that

$$\mathbf{ij} = -\mathbf{ji} = \mathbf{k}, \quad \mathbf{jk} = -\mathbf{kj} = \mathbf{i}, \quad \mathbf{ki} = -\mathbf{ik} = \mathbf{j}$$

The *conjugate* q^* of q is $q^* = q_0 - q_1\mathbf{i} - q_2\mathbf{j} - q_3\mathbf{k}$. The *$L_2$-norm* $\|q\|$ of q is $\sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$. If $\|q\| = 1$, then q is called a *unit quaternion*. For two quaternions $q = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$ and $r = r_0 + r_1\mathbf{i} + r_2\mathbf{j} + r_3\mathbf{k}$,

$$\begin{aligned}
\langle q, r \rangle &= q_0r_0 + q_1r_1 + q_2r_2 + q_3r_3 \\
q + r &= (q_0 + r_0) + (q_1 + r_1)\mathbf{i} + (q_2 + r_2)\mathbf{j} + (q_3 + r_3)\mathbf{k} \\
q \cdot r &= (q_0r_0 - q_1r_1 - q_2r_2 - q_3r_3) + \\
&\quad (q_0r_1 + q_1r_0 + q_2r_3 - q_3r_2)\mathbf{i} + \\
&\quad (q_0r_2 - q_1r_3 + q_2r_0 + q_3r_1)\mathbf{j} + \\
&\quad (q_0r_3 - q_1r_2 + q_2r_1 + q_3r_0)\mathbf{k}.
\end{aligned}$$

Sometimes it is more convenient to write the product in a matrix form:

$$qr = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = Rq.$$

The product $r \cdot q$ can also be represented as $\bar{Q}r$, where $\bar{Q}$ is the matrix orthogonal to Q, obtained by transposing the lower right 3×3 minor.

The rotation about a unit vector $\langle u_x, u_y, u_z \rangle \in \mathbb{R}^3$ by an angle θ can be represented by a unit quaternion

$$q = \cos(\theta/2) + \sin(\theta/2)(u_x\mathbf{i} + u_y\mathbf{j} + u_z\mathbf{k}).$$

Since $\|q\| = 1$, q can also be regarded as a point in $\mathbb{S}^3$. If we represent a vector $\langle r_x, r_y, r_z \rangle \in \mathbb{R}^3$ as the quaternion $r = 0 + r_x\mathbf{i} + r_y\mathbf{j} + r_z\mathbf{k}$, then the rotation of r by angle θ with respect to the axis u can be represented as

$$I_q(r) = qrq^* = (Qr)q^* = \bar{Q}^T(Qr) = (\bar{Q}^TQ)r.$$

Optimal rotation. Armed with the concept of quaternions, we are ready to solve the following problem. Given two sets of points $R = \{r_j \in \mathbb{R}^3 \mid 1 \le j \le n\}$ and $G = \{g_j \in \mathbb{R}^3 \mid 1 \le j \le n\}$, we would like to find a rotation $q \in \mathbb{S}^3$ (represented as a quaternion) that minimizes $\sum_{j=1}^n \|I_q(r_j), g_j\|^2$. In othere words, the goal is to minimize

$$\begin{aligned}
f(q) &= \min_{q \in \mathbb{S}^3} \sum_{j=1}^n \|qr_jq^* - g_j\|^2 \\
&= \min \sum_{j=1}^n (\|r_j\|^2 + \|g_j\|^2 - 2\langle qr_jq^*, g_j \rangle).
\end{aligned}$$

Since $\|r_j\|^2 + \|g_j\|^2$ does not depend on q, it suffices to maximize the last term. Using the matrix form of the product of two quaternions, let $qr_j = \bar{R}_j q$ and $g_j q = G_j q$. Then

$$\begin{aligned}\sum_{j=1}^{n} \langle qr_j q^*, g_j \rangle &= \sum_{j=1}^{n} \langle qr_j, g_j q \rangle = \sum_{j=1}^{n} \langle \bar{R}_j q, G_j q \rangle \\ &= \sum_{j=1}^{n} q^T \bar{R}_j^T G_j q = q^T \left(\sum \bar{R}_j^T G_j \right) q \\ &= q^T A q,\end{aligned}$$

where $A = \sum_{j=1}^{n} R_j^T G_j$ is a 4×4 symmetric matrix. Let X_l and λ_l, for $l = 1, 2, 3, 4$ be its eigenvectors and eigenvalues, and assume $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4$. We have

$$AX_l = \lambda_l X_l \qquad (l = 1, 2, 3, 4)$$

Rewrite q as $\sum_{l=1}^{4} \alpha_l X_l$. Note that $\sum_l \alpha_l^2 = 1$, therefore

$$\begin{aligned}q^T A q &= q^T A \left(\sum_{l=1}^{4} \alpha_l X_l \right) \\ &= q^T \left(\sum \alpha_l A X_l \right) \\ &= q^T \left(\sum \alpha_l \lambda_l X_l \right) \\ &= \sum \alpha_l^2 \lambda_l.\end{aligned}$$

Clearly, the expression is maximized when $\alpha_1 = 1$, $q = X_1$, and we have computed $f(q)$.

Bibliographic Notes.

These lecture notes are adapted from [1] and the last year's lectures notes for CPS260. Refer to the book [2] for details on dual complex. The paper by Horn [3] provides an excellent exposition to rotations using quaternions.

References

[1] H. Edelsbrunner, Biogeometric Modeling, `http://www.cs.duke.edu/education/courses/fall02/cps296.1/`. 23-1, 23-6

[2] H. Edelsbrunner, *Geometry and Topology for Mesh Generation*, Cambridge University Press, 2001. 23-6

[3] B. K. Horn, Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Amer.* 4 (1987), 629–642.