Topology-Driven Trajectory Synthesis with an Example on Retinal Cell Motions

Chen Gu[1], Leonidas Guibas[2], and Michael Kerber[3]

[1] Google Inc., USA
guc@google.com
[2] Stanford University, USA
guibas@cs.stanford.edu
[3] Max-Planck-Institute for Informatics, Germany
mkerber@mpi-inf.mpg.de

Abstract. We design a probabilistic trajectory synthesis algorithm for generating time-varying sequences of geometric configuration data. The algorithm takes a set of observed samples (each may come from a different trajectory) and simulates the dynamic evolution of the patterns in $O(n^2 \log n)$ time. To synthesize geometric configurations with indistinct identities, we use the pair correlation function to summarize point distribution, and α-shapes to maintain topological shape features based on a fast persistence matching approach. We apply our method to build a computational model for the geometric transformation of the cone mosaic in retinitis pigmentosa — an inherited and currently untreatable retinal degeneration.

Keywords: trajectory, pair correlation function, alpha shapes, persistent homology, retinitis pigmentosa.

1 Introduction

The work presented in this paper is motivated from the investigation of a retinal disease called retinitis pigmentosa [18]. In this disease, a mutation kills the rod photoreceptors in the retina. A consequence of this death is that the geometry of the mosaic of cone photoreceptors deforms in an interesting way. Normally, cones form a relatively homogeneous distribution. But after the death of rods, the cones migrate to form an exquisitely regular array of holes.

Our central goal is to build a dynamic evolution model for the point distributions that arise from the cone mosaic in retinitis pigmentosa. In physics, the most classical method for modeling cell motions is to solve a system of differential equations from Newton's laws of motion with some predefined force field which specifies cell-to-cell interactions. However, in many cases it is difficult to understand how different types of cells (for example cones and rods) interact with each other. There are also mathematical models that do not presume much prior biological knowledge, such as flocking which has been widely used to simulate coordinated animal motions [16]. But as with all model-based approaches, the method is limited by the model chosen in the first place.

D. Brown and B. Morgenstern (Eds.): WABI 2014, LNBI 8701, pp. 326–339, 2014.
© Springer-Verlag Berlin Heidelberg 2014

Instead of fitting a predefined model, we propose an alternative approach which relies only on geometric and topological multi-scale summaries. The input is a set of geometric configurations, each of which may come from a different trajectory of a cone migration. We design a probabilistic algorithm to synthesize trajectories from observed data. In short, we let the points move randomly and check whether the transformation brings them "closer" to our next observation. To define closeness, we combine two high-level distance measures: firstly, we employ the *pair correlation function* (PCF) which extracts pairwise correlations in the point cloud data by measuring how density varies as a function of distance from a reference point. The PCF is widely accepted as an informative statistical measure for point set analysis, and has been used for trajectory synthesis in previous work [14]. As major novelty, we propose to combine the PCF with a topological distance measure: we compare *persistence diagrams of alpha-shape filtrations* which capture the evolution of holes that arise when the points are thickened to disks with increasing radius. Persistence diagrams are currently a popular research topic with many theoretical and practical contributions; we refer to the surveys [1,6] for contemporary overviews. We demonstrate that the combination of PCF and persistence diagrams results in trajectories with a much cleaner hole structure than for trajectories obtained only by PCF (see Figure 6). We believe the problem of trajectory synthesis for very sparse data to be of more general importance in biological and medical contexts, and hope that our model-free methodology can be applied to other contexts. Moreover, our approach provides evidence that topological methods are useful in the analysis of point distributions which have been extensively studied in computer graphics and point processes [7,15,17].

2 Biological Background: Retinitis Pigmentosa

The retina is a light-sensitive layer of tissue that lines the inner surface of the eye. It contains photoreceptor cells that capture light rays and convert them into electrical impulses. These impulses travel along the optic nerve to the brain where they are turned into images of the visual world.

There are two types of photoreceptors in the retina: cones and rods. In adult humans, the entire retina contains about 6 million cones and 120 million rods. Cones are contained in the macula, the portion of the retina responsible for central vision. They are most densely packed within the fovea, the very center portion of the macula. Cones function best in bright light and support color perception. In contrast, rods are spread throughout the peripheral retina and function best in dim light. They are responsible for peripheral and night vision.

Retinitis pigmentosa is one of the most common forms of inherited retinal degeneration. This disorder is characterized by the progressive loss of photoreceptor cells and may lead to night blindness or tunnel vision. Typically, rods are affected earlier in the course of the disease, and cone deterioration occurs later. In the progressive degeneration of the retina, the peripheral vision slowly constricts and the central vision is usually retained until late in the disease.

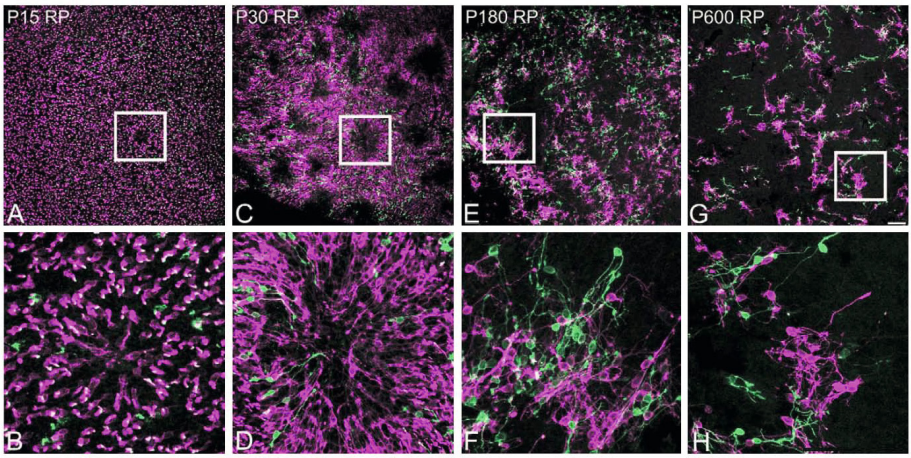


Fig. 1. Cone mosaic rearrangement in retinitis pigmentosa. The confocal micrographs in the top row show middle (red) and short (green) wavelength sensitive cones in whole mount retinas. Enlarged micrographs of marked regions are shown in the bottom row. This figure is taken from [8].

At present, there is no cure for retinitis pigmentosa. Researchers around the world are constantly working on development of treatments for this condition.

There have been some recent studies on the spatial rearrangement of the cone mosaic in retinitis pigmentosa [8,13]. These experiments are performed on a rat model in which a mutation in the retina triggers the cell death of rods, similar to those causing symptoms in humans. Figure 1 shows an example for the morphology and distribution of cones at postnatal days 15, 30, 180, and 600. In healthy retinas, the mosaic of cones exhibits a spatially homogeneous distribution. However, the death of rods causes cones to rearrange themselves into a mosaic comprising an orderly array of holes. These holes first begin to appear at random regions of the retina at day 15 and become ubiquitous throughout the entire tissue at day 30. Holes start to lose their form at day 180 and mostly disappear at day 600, at which time the cones are almost all dead.

Furthermore, it has been observed that both cones and rods follow the same retinal distribution. But the mechanisms of formations of holes of cones are different from those of rods. In fact, retinitis pigmentosa is caused by the initial loss of rods in the center of these holes, and then the death of rods tends to propagate as circular waves from the center of the holes outward. In contrast, the number of cones in normal and retinitis pigmentosa conditions do not show significant differences at stages as late as day 180. Therefore, holes of cones do not form by cell death at their centers, but by cell migration.

Since cones take a long time to die out, understanding whether and how these hole structures improve the survival of cones would provide scientific and clinical communities with better knowledge of how to preserve day and high acuity vision

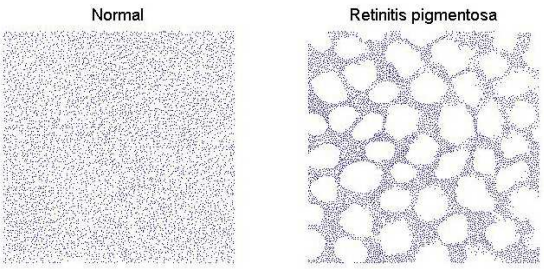


Fig. 2. Distribution of cones in normal (7523 cells in 1mm^2) and retinitis pigmentosa (6509 cells in 1mm^2) retinas at day 25

in retinitis pigmentosa. This motivated us to build a model for the geometric transformation of the cone mosaic in the retinal degeneration. The challenge in building such a model is that we only have access to one snapshot per rat, because the animals are killed before their retinas can be dissected. Therefore, we have very limited data and there is no correspondence between the cells in different snapshots.

3 Synthesis Algorithm

Suppose we are given a point set $X = \{x_1, x_2, \ldots, x_n\}$ at time t and we want to simulate the time evolution from t to $t + \Delta t$. Since we do not presume any biological knowledge about the system, in each step we simply move a point x_i to some random location x_i' within its neighborhood. We then compare both the old configuration $\{x_1, \ldots, x_i, \ldots, x_n\}$ and the new configuration $\{x_1, \ldots, x_i', \ldots, x_n\}$ after this point update to the real data at time $t + \Delta t$. If the new configuration is closer to the data than the old configuration, we accept this movement for x_i, otherwise we accept it with some probability which depends on their difference. We iteratively repeat this process for each point in X until the result converges.

The details of the trajectory synthesis algorithm are shown in Algorithm 1. It can be seen as a variant of the simulated annealing algorithm [9], in which the acceptance probability also depends on a temperature parameter to avoid local minima in optimization. There are two questions we have not addressed:

- how do we compare synthetic configurations to the real data at time $t + \Delta t$?
- what happens if we do not have observation at time $t + \Delta t$?

In fact, we have reduced the problem of motion modeling to quantifying some kind of distances between point sets in the synthetic and real data. Note that the number of points n in the synthesis algorithm is kept constant during simulation, but the point sets in the observed data may have different cardinalities (see Figure 2). Furthermore, since we only have access to one snapshot per animal, there is no correspondence between point sets in different snapshots. In the next

Algorithm 1. Trajectory synthesis

Input: sample point sets $\{X^{t_0}, X^{t_1}, \ldots, X^{t_M} \mid t_0 = 0 < t_1 < \ldots < t_{M-1} < t_M = 1\}$, number of frames N.
Output: synthesis point sets $\{Y^t \mid t = 0, 1/N, \ldots, (N-1)/N, 1\}$.
Procedure:

1: set time $t = 0$, point set $Y^0 = X^0$.
2: **while** $t < 1$ **do**
3: set $t = t + 1/N$, initialize $Y^t = Y^{t-1/N}$.
4: find time interval $t_i < t \leq t_{i+1}$.
5: interpolate target pair correlation function g_{X^t} between $g_{X^{t_i}}$ and $g_{X^{t_{i+1}}}$.
6: interpolate target persistence diagram P_{X^t} between $P_{X^{t_i}}$ and $P_{X^{t_{i+1}}}$.
7: pick three random directions for persistence matching.
8: set iteration $k = 0$, select initial temperature T_0.
9: **repeat**
10: set $k = k + 1$, $T = T_0/k$.
11: **for** each point y in Y^t **do**
12: replace y by a random neighbor y' to form a new point set Y'.
13: compute distance d_1 between pair correlation functions g_{X^t} and g_{Y^t}.
14: compute distance d_2 between persistence diagrams P_{X^t} and P_{Y^t} based on their three 1D projections.
15: define distance between X^t and Y^t as $d = d_1 + \lambda d_2$.
16: repeat lines 13–15 to compute distance d' between X^t and Y'.
17: **if** $d' < d$ **then**
18: accept new point set $Y^t = Y'$.
19: **else**
20: accept Y' with probability $\exp(\frac{d-d'}{T})$.
21: **end if**
22: **end for**
23: **until** Y^t converges.
24: **end while**

two sections, we will describe how to define distances on configurations with indistinct identities and use them to interpolate missing data.

4 Geometry: Pair Correlation Function

Given a point set $X = \{x_1, x_2, \ldots, x_n\}$ in $\mathbb{R}^d$ with number density ρ, the pair correlation function is defined as

$$g_X(r) = \frac{1}{S_{d-1}(r)\rho n} \sum_{i=1}^{n} \sum_{j=1}^{n} G(||x_i - x_j|| - r), \ \forall r \geq 0 \tag{1}$$

where $S_{d-1}(r)$ is the surface area of a ball of radius r in $\mathbb{R}^d$, and G is a 1D Gaussian kernel $G(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{x^2}{2\sigma^2})$.

The PCF provides a compact representation for the characteristics of point distribution. Note that in (1) there are two normalization factors ρ and $S_{d-1}(r)$.

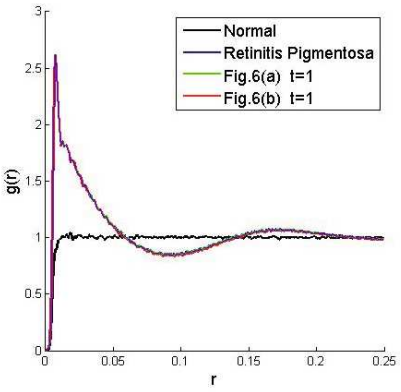


Fig. 3. Pair correlation function

The number density ρ is an intensive quantity to describe the degree of concentration of points in the space, and is typically defined as $\rho = n/V$ where V is the volume of the observation region. Since it is more likely to find two points with a given distance in a more dense system, this factor is used for comparing point sets with different cardinalities. The other inverse weighting factor $S_{d-1}(r)$ is the surface area of a ball of radius r in $\mathbb{R}^d$ (for example $S_1(r) = 2\pi r$). This accounts for the fact that as r gets larger, there will be naturally more points with the given distance from a reference point. After these normalizations, it can be shown that $\lim_{r\to\infty} g_X(r) = 1$ for any infinite point set X, and hence most information about the point set is contained in $g_X(r)$ for the lower values of r. For a finite point set, we can apply periodic boundary conditions to remove the window edge effects.

Figure 3 shows the PCF for the photoreceptor point sets in Figure 2 with $\sigma = 0.1r_{\max}$, where $r_{\max} = \sqrt{\frac{1}{2\sqrt{3}n}}$ is the maximum possible radius for n identical circles that can be packed into a unit square [12]. For the normal point set, we see that the density is almost 1 everywhere except for $r < 0.005$mm, which is about the diameter of cone cell bodies — such a pattern is called blue-noise where points are distributed randomly with a minimum distance between each pair. For healthy primate retinas, it is well-known that photoreceptor distributions may follow a blue-noise-like arrangement to yield good visual resolution [19]. In contrast, for the retinitis pigmentosa point set, the high densities at small distances show the clustering of cones in the sick retina, implying the cells become closer by migration. After we have computed the PCFs, it is natural to define their distance as

$$d(g_X, g_Y) = \left(\int_r (g_X(r) - g_Y(r))^2 \, \mathrm{d}r\right)^{1/2} \tag{2}$$

It is obvious that computing the PCF for a point set $X = \{x_1, x_2, \ldots, x_n\}$ takes $O(n^2)$ time in (1). However, when we move a point x_i to x_i' in the synthesis algorithm, it only takes $O(n)$ time to update the PCF for the new point set X':

$$g_{X'}(r) = g_X(r) + \frac{2}{S_{d-1}(r)\rho n} \sum_{j \neq i} (G(||x_i' - x_j|| - r) - G(||x_i - x_j|| - r))$$

Of course since we compute the densities at different distances, the running time may also depend on the range and discretization of the distance r. But for Gaussian kernels we can set a cutoff threshold δ, so that for each pairwise distance $||x_i - x_j||$ we only need to update $g_X(r)$ at distance $||x_i - x_j|| - \delta < r < ||x_i - x_j|| + \delta$, which contains $O(1)$ discretized values of r.

Data Interpolation. Now we answer the two questions proposed at the end of Section 3. Consider we have a set of observed samples $\{X^{t_0}, X^{t_1}, \ldots, X^{t_M}\}$. Without loss of generality, we can assume the observation time $t_0 = 0 < t_1 < \ldots < t_{M-1} < t_M = 1$. We start with $Y^0 = X^0$ as the initial point set, and run the synthesis algorithm to simulate its time evolution. By matching the PCFs of $\{X^{t_1}, X^{t_2}, \ldots, X^{t_M}\}$, we can obtain a sequence of point sets $\{Y^{t_1}, Y^{t_2}, \ldots, Y^{t_M}\}$ at all observation time. For each sample X^{t_i}, the goal is to minimize the distance between $g_{X^{t_i}}$ and $g_{Y^{t_i}}$ defined in (2). Furthermore, if there is more than one sample observed at time t_i, we can extend the objective function in standard ways, by taking the minimum or average distance from the synthetic point set to all samples at that time.

Note that in the above approach we can only synthesize point sets at the observation time $\{t_0, t_1, \ldots, t_M\}$. But how do we simulate during the time intervals between successive observations? Suppose we want to generate a point distribution at time $t_i < t < t_{i+1}$. Although there is no real data X^t, it is possible to approximate the PCF g_{X^t} by linear interpolation

$$g_{X^t} = \frac{t_{i+1} - t}{t_{i+1} - t_i}\, g_{X^{t_i}} + \frac{t - t_i}{t_{i+1} - t_i}\, g_{X^{t_{i+1}}}$$

It has been shown that such a simple linear interpolation can generate valid PCFs from which distributions can be synthesized [14]. Thus, we can use the synthesis algorithm to generate data at any time $t_0 \leq t \leq t_M$.

5 Topology: Distance of Persistence Diagrams

In Section 4, we have seen that the PCF can be used to characterize the distributions of photoreceptor point sets. However, this function only considers pairwise correlations and misses higher-order information in the data. As we will show in Section 6, there are point sets with almost same PCF but very different shape features. In this section, we present another way to summarize point distribution without correspondence from a topological perspective.

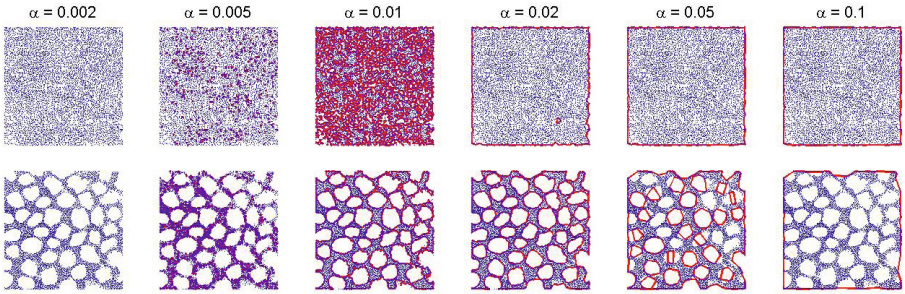


Fig. 4. Alpha shapes

Alpha Shapes. Suppose we are given a point set and we want to understand the shape formed by these points. Of course there are many possible interpretations for the notion of shape, the α-shape being one of them [4]. In geometry, α-shapes are widely used for shape reconstruction, as they give linear approximations of the original shape.

The concept of α-shapes is generally applicable to point sets in any Euclidean space $\mathbb{R}^d$, but for our application we will illustrate in the 2D case. Given a point set S in $\mathbb{R}^2$, the α-shape of S is a straight line graph whose vertices are points in S and whose edges connect pairs of points that can be touched by the boundary of an open disk of radius α containing no points in S. The parameter α controls the desired level of detail in shape reconstruction. For any value of α, the α-shape is a subgraph of the Delaunay triangulation, and thus it can be computed in $O(n \log n)$ time.

Figure 4 shows the α-shapes for the photoreceptor point sets in Figure 2 with different values of α. As α increases, we see that edges appear in the graph and some of them form cycles. For the normal point set, these edges and cycles disappear very quickly since there is no space for empty disks of large radius α. In contrast, for the retinitis pigmentosa point set, some cycles can stay for long time in the large empty regions. Therefore, α-shapes can successfully capture the hole structures formed by cone migration.

In Figure 4, we see that $\alpha = 0.02$mm gives a nice example to distinguish between the two photoreceptor point sets. However, in general how do we choose the right value of α? Indeed, what we are more interested in is to summarize information of α-shapes at different scale levels. So, we next turn to its topological definition — the α-complex. Given a point set in $\mathbb{R}^d$, the α-complex is a simplicial subcomplex of its Delaunay triangulation. For each simplex in the Delaunay triangulation, it appears in the α-complex $K(\alpha)$ if its circumsphere is empty and has a radius less than α, or it is a face of another simplex in $K(\alpha)$.

Although we can choose infinite numbers for α, there are only finite many α-complexes for a point set S. They are totally ordered by inclusion giving rise to filtration of the Delaunay triangulation $K_0 = \phi \subset K_1 = S \subset ... \subset K_m = \mathrm{Del}(S)$. For a point set in $\mathbb{R}^2$, α-complexes consist of vertices, edges, and triangles.

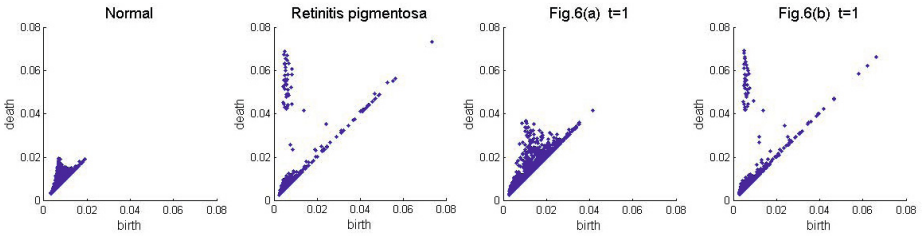


Fig. 5. Persistence diagram

The first non-empty complex K_1 is the point set S itself. As α increases, edges and triangles are added into $K(\alpha)$ until we eventually arrive at the Delaunay triangulation. The relation between α-shape and α-complex is that the edges in the α-shape make up the boundary of the α-complex.

Persistence. In Figure 4, we have seen that cycles appear and disappear in the α-complexes during the filtration. The cycles that stay for a while are important ones since they characterize major shape features of the data set. In algebraic topology, the cycles are defined based on homology groups: there is one group of cycles H_d per dimension d, and the rank of H_d is called the d-th Betti number β_d which can be considered as the number of d-dimensional holes in the space [5]. For example in the 2D case, β_0 is the number of connected components and β_1 is the number of holes in the plane. In the evolution from K_0 to K_m, adding an edge will create a new hole (except for $n-1$ edges in a spanning tree which change β_0 by merging connected components), while adding a triangle will fill a hole. The persistence of a hole is the difference between its birth time and death time which are paired by following the elder rule.

Given a point set S, the information about persistence of holes can be encoded into a two-dimensional persistence diagram P_S. As depicted in Figure 5, each point in the diagram represents a hole (or a class of cycles) during the filtration, where the x and y coordinates are the birth time and death time respectively. In the normal case all cycles have short persistence, while in the retinitis pigmentosa case some cycles have very long persistence and they capture the large hole features in the point set. Note that there are also some cycles with large birth time and very short persistence (the points near the diagonal). This is because the holes in the point set may not be perfectly round (such as ellipses), and thus some cycles can be split by adding long edges at large α. These cycles of short persistence can be considered as noise and ignored in the analysis of the data.

For a Delaunay triangulation with m simplicies, the persistence diagram can be computed using a matrix reduction in $O(m^3)$ time. In the 2D case, $m = O(n)$ and the running time can be reduced to $O(n\alpha(n))$ using the union-find data structure [5], where $\alpha(n)$ is the inverse of Ackermann function which grows very slowly with n. We also apply periodic boundary conditions by computing the periodic Delaunay triangulation of a point set [2,10].

There are two distances often used to measure the similarity between persistence diagrams: the bottleneck and Wasserstein distances [5]. Computing both distances reduces to the problem of finding the optimal matching in a bipartite graph. With the optimal matching, we can also interpolate between two persistence diagrams by linearly interpolating between the matched pairs of points. However, solving a minimum cost perfect matching problem in non-Euclidean spaces takes $O(n^3)$ time [11], so we should avoid recomputing this matching distance after each point update in the synthesis algorithm.

Matching. In this section, we present a faster algorithm to measure the similarity between persistence diagrams from their 1D projections. Instead of computing the optimal matching between 2D persistence diagrams, we take several directions and match their 1D projections in each direction independently. Given two persistence diagrams X, Y and k directions $w_1, w_2, \ldots, w_k$, we define the distance between X and Y as the sum of their 1D matching costs

$$d(X,Y) = \sum_{i=1}^{k} \Big(\min_{f_i : X_{w_i} \to Y_{w_i}} \sum_{x \in X_{w_i}} |x - f_i(x)| \Big) \tag{3}$$

where X_{w_i} is the projection of X onto direction w_i, and f_i is a bijection between X_{w_i} and Y_{w_i} (for simplicity we first assume that X and Y have same cardinality). It is easy to verify that the minimal matching cost over all bijections between X_{w_i} and Y_{w_i} can be computed in $O(n \log n)$ time by sorting X_{w_i} and Y_{w_i}, and matching pairs in ascending order. Furthermore, by randomly choosing three directions, we can uniquely reconstruct a point set from its three 1D projections with high probability.

Theorem 1. *Given a 2D multiset of points $P = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ in general position*[1]*, the set of directions $x + cy$ such that P cannot be uniquely reconstructed from its 1D projections P_x, P_y, and P_{x+cy} has measure zero.*

Proof. Assuming there is another multiset of points $P' \neq P$ with the same three 1D projections. We take a point $p \in P' - P$ which consists of points that appear more times in P' than P. We first claim that $p \notin P$, otherwise since P is in general position there is no point other than p in P with the same y-coordinate, and thus the y-coordinate of p will appear more times in P'_y than P_y.

Let p be reconstructed from projection lines $x = x_i$, $y = y_j$, and $x + cy = x_k + cy_k$ where (x_i, y_i), (x_j, y_j), and (x_k, y_k) are all in P. So $x_i + cy_j = x_k + cy_k$. If $y_j = y_k$, then $x_i = x_k$. Thus $p = (x_i, y_j) = (x_k, y_k)$ is also in P — a contradiction. So $y_j \neq y_k$ and $c = (x_k - x_i)/(y_j - y_k)$. Therefore, there are at most $O(n^3)$ values of c without a unique reconstruction. □

[1] We define a 2D multiset of points $P = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ to be in general position if two points in P cannot have same y-coordinate unless they are coincident ($y_i = y_j \Rightarrow x_i = x_j$). If a multiset of points is not in general position, we can always rotate the point set by some angle ω clockwise to make it in general position. This is equivalent to reconstruct the original point set if we rotate directions x, y, and $x + cy$ by angle ω counter-clockwise.

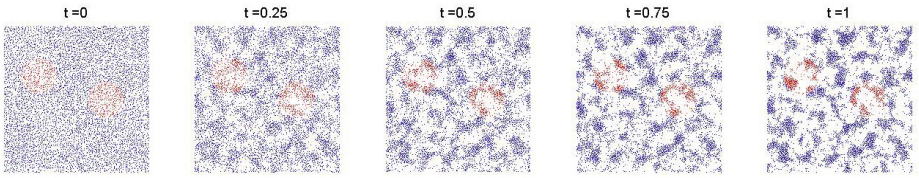
To focus on major shape features of point sets, we choose the projections as follows: we randomly select three directions $x + cy$, where $c = \tan\theta$ and the angle θ is uniformly chosen from $(0, \pi/2) \cup (\pi/2, 3\pi/4)$. For each direction, we project all points (x, y) to $f_n(x, y) = ((x\cos\theta + y\sin\theta)^n - (x\cos\theta + x\sin\theta)^n)^{1/n}$, where n is a large positive number. It is easy to verify that $\lim_{y\to x} f_n(x, y) = 0$ and $\lim_{n\to\infty} f_n(x, y) = \begin{cases} x\cos\theta + y\sin\theta & , \text{if } x < y \\ 0 & , \text{if } x = y \end{cases}$. As $n \to \infty$, the function $f_n(x, y)$ captures the projection of persistence diagram onto direction $x + cy$, while it ignores noise near the diagonal. In practice, we find that $n = 8$ is usually good enough to serve as ∞. Finally, when comparing persistence diagrams with different cardinalities, we may assume that there exist infinitely many extra points on the diagonal — which all map to zero after projection.

6 Experimental Results: Cone Mosaic Rearrangement

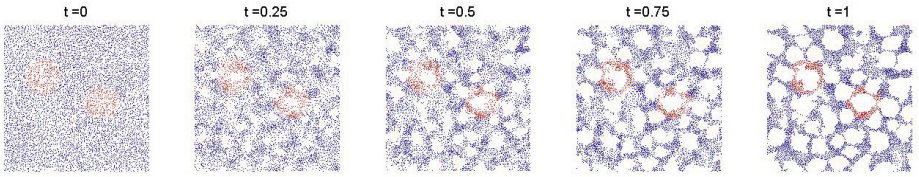
We first test the performance of the synthesis algorithm using PCF only [14]. In this case, the distance between two point sets measures the difference between their pairwise correlations in (2). For point update, in each step we move a point to a random location within its neighborhood of radius $r_{\max}$ (as defined in Section 4). We generate $N = 16$ frames to simulate the cone mosaic rearrangement in retinitis pigmentosa. In Figure 6(a), we have labeled some points in red color to show their correspondences in different snapshots. By matching PCFs, we see that the algorithm creates several sparse regions in the point set. However, the synthetic point set ($t = 1$) looks very different from the real data shown in Figure 2 — there are many outliers inside the sparse regions by the synthesis algorithm, while the holes of cones in retinitis pigmentosa seem to be very clean. If we compare the shape features for these two point sets, their PCFs are almost well-matched (see Figure 3). On the other hand, there is a big difference between their persistence diagrams because these outliers would significantly shorten the persistence of cycles in the α-complex (see Figure 5).

So, we next incorporate α-shapes to maintain the topological features. In this case, the distance function involves two parts: let d_1 be the distance between PCFs of two point sets in (2), and d_2 be the distance between their persistence diagrams in (3). We define the new distance as $d = d_1 + \lambda d_2$, where $\lambda > 0$ is a weight parameter and in our implementation we set the two parts to be equally weighted. The synthesis result using both PCF and α-shapes is shown in Figure 6(b). We see that holes appear in random positions and grow gradually in size as time increases. At the end of simulation, the points labeled in red color move close to the boundaries of holes. In Figures 3 and 5 we can see that the shape features for the synthetic point set match the targets very well. There are only some small differences between persistence diagrams near the diagonal, but they are considered as noise.

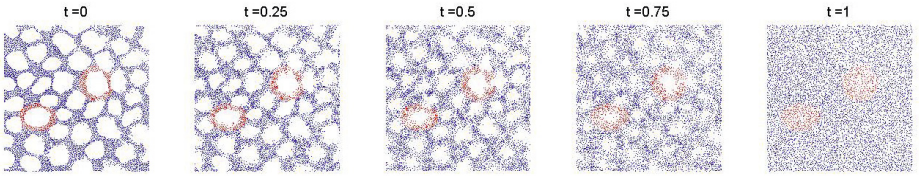
Figure 6(c) shows the simulation result in the reverse direction where we start with a retinitis pigmentosa distribution and move points towards a normal distribution. Actually the synthesis in this direction is much easier because we



(a) Pair correlation function only



(b) Pair correlation function + α-shapes



(c) Reverse direction

Fig. 6. Simulation results

already know the hole positions. After filling the holes we end up with a blue-noise pattern. By reversing the sequence of snapshots in Figure 6(c), it gives another example on retinal cell motions in retinitis pigmentosa. Furthermore, we can start with a point set at any time t and run bidirectional simulations to synthesize trajectories for the time evolution of this sample.

Running Time. There are four main components of the trajectory synthesis algorithm (see Table 1). For PCF, we only need to compute it for the initial and target point sets, which takes $O(n^2)$ time. After that, it takes $O(n)$ time per point update. For α-shapes, it takes $O(n \log n)$ time for Delaunay triangulation and persistence matching, as well as $O(n\alpha(n))$ time for persistence diagram. Therefore, the running time for all these four parts is almost linear per point update, and hence the algorithm runs in $O(n^2 \log n)$ time per iteration.

We have also tested the real running time for each part of the synthesis algorithm on the photoreceptors data set. The experiment is performed on a computer with Intel® Core™ 2 Quad Processor Q6600 and 4GB Memory. In the current implementation, the periodic Delaunay triangulation is the slowest part which takes about half of the computation time. However, for each point update there is no need to recompute the whole Delaunay triangulation, and indeed it

Table 1. Running time of trajectory synthesis algorithm

Algorithm	Computation	Update	Real time
Pair correlation function	$O(n^2)$	$O(n)$	8 %
Delaunay triangulation (periodic)	$O(n \log n)$	$[O(\log n)]$	46 %
Persistence diagram	$O(n\alpha(n))$	—	27 %
Persistence matching	$O(n \log n)$	—	18 %

can be maintained in $O(\log n)$ expected time per point update [3]. So, by using dynamic Delaunay triangulation we can improve the real running time by almost a factor of 2, but theoretically the algorithm still takes $O(n \log n)$ time per point update — for persistence matching the input is the persistence diagram and it is not clear how to bound its change after we move a point.

Note that in the initialization part, we may need to interpolate the target persistence diagram at time t if we do not have the real data at that time. As mentioned in Section 5, this would take $O(n^3)$ time. Therefore, if we synthesize N frames and run L iterations per frame, the total running time is bounded by $O(N(n^3 + Ln^2 \log n))$. Although the initialization part has a larger theoretical cost $O(n^3)$, in practice the main synthesis part $O(Ln^2 \log n)$ may take longer time because its unit cost $O(1)$ is more expensive. For the simulation results shown in Figure 6(b–c), they take about 3450 seconds for initialization and 290 seconds per iteration, with $L = 20$ iterations for each frame. Furthermore, since the synthesis algorithm is probabilistic, we can use it to generate multiple trajectories from a data set, while the initialization can be considered as a preprocessing step and only needs to be computed once.

Acknowledgements. We wish to thank Dr. Norberto Grzywacz for providing us with the biological data for our experiments. This work was partially supported by the NSF grant DMS 0900700 and the Max Planck Center for Visual Computing and Communication.

References

1. Carlsson, G.E.: Topology and data. Bulletin of the American Mathematical Society 46, 255–308 (2009)
2. Caroli, M., Teillaud, M.: Computing 3D periodic triangulations. In: Proceedings of the European Symposium on Algorithms, pp. 59–70 (2009)
3. Devillers, O., Meiser, S., Teillaud, M.: Fully dynamic Delaunay triangulation in logarithmic expected time per operation. Computational Geometry: Theory and Applications 2(2), 55–80 (1992)
4. Edelsbrunner, H.: Alpha shapes — a survey. Tessellations in the Sciences (2011)
5. Edelsbrunner, H., Harer, J.L.: Computational Topology. An Introduction. American Mathematical Society (2010)
6. Edelsbrunner, H., Morozov, D.: Persistent homology: theory and practice. In: Proceedings of the European Congress of Mathematics, pp. 31–50 (2012)

7. Illian, J., Penttinen, A., Stoyan, H., Stoyan, D.: Statistical Analysis and Modelling of Spatial Point Patterns. Wiley Interscience (2008)
8. Ji, Y., Zhu, C.L., Grzywacz, N.M., Lee, E.-J.: Rearrangement of the cone mosaic in the retina of the rat model of retinitis pigmentosa. The Journal of Comparative Neurology 520(4), 874–888 (2012)
9. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220(4598), 671–680 (1983)
10. Kruithof, N.: 2D periodic triangulations. CGAL User and Reference Manual (2009)
11. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Research Logistics 2(1-2), 83–97 (1955)
12. Lagae, A., Dutre, P.: A comparison of methods for generating Poisson disk distributions. Computer Graphics Forum 27(1), 114–129 (2008)
13. Lee, E.-J., Ji, Y., Zhu, C.L., Grzywacz, N.M.: Role of muller cells in cone mosaic rearrangement in a rat model of retinitis pigmentosa. Glia 59(7), 1107–1117 (2011)
14. Oztireli, A.C., Gross, M.: Analysis and synthesis of point distributions based on pair correlation. Transactions on Graphics 31(6), 170 (2012)
15. Schlomer, T., Deussen, O.: Accurate spectral analysis of two-dimensional point sets. Journal of Graphics, GPU, and Game Tools 15(3), 152–160 (2011)
16. Vicsek, T., Czirok, A., Ben-Jacob, E., Cohen, I., Shochet, O.: Novel type of phase transition in a system of self-driven particles. Physical Review Letters 75(6), 1226–1229 (1995)
17. Wilkinson, L., Anand, A., Grossman, R.: Graph-theoretic scagnostics. In: Proceedings of the Symposium on Information Visualization, pp. 157–164 (2005)
18. Yanoff, M., Duker, J.S.: Ophthalmology. Elsevier (2009)
19. Yellott, J.I.: Spectral consequences of photoreceptor sampling in the rhesus retina. Science 221(4608), 382–385 (1983)