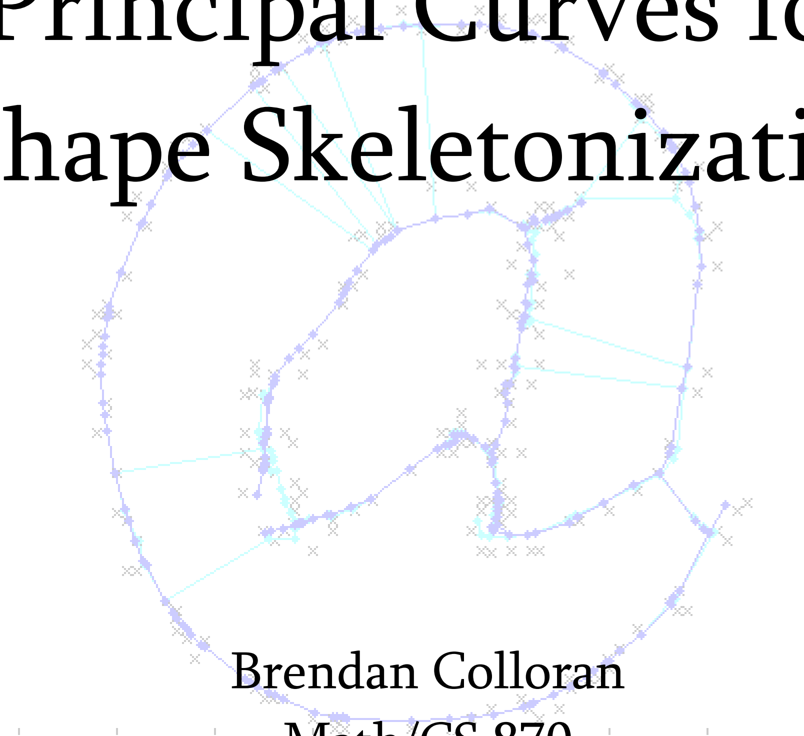
Proximity Graphs and Principal Curves for Shape Skeletonization



Brendan Colloran
Math/CS 870
May 3, 2007

The Problem:

Investigate the relationship between shape skeletons, proximity graphs, and principal curves.

So what is a shape skeleton?

What are Proximity Graphs? (Redux)

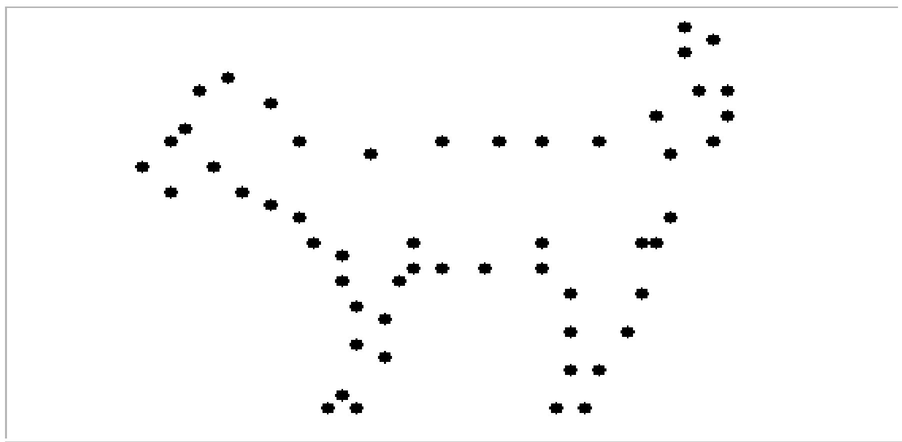
- A proximity graph is a simply a graph in which two vertices are connected by an edge if and only if the vertices satisfy particular geometric requirements.

- "Proximity" here means spatial distance.

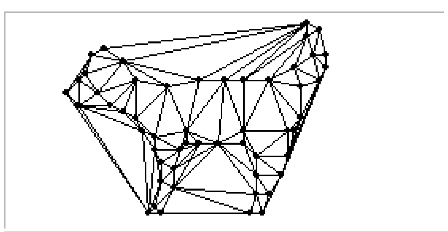
- Many of these graphs can be formulated with respect to many metrics, but the Euclidean metric is used most frequently.

Why Proximity Graphs?

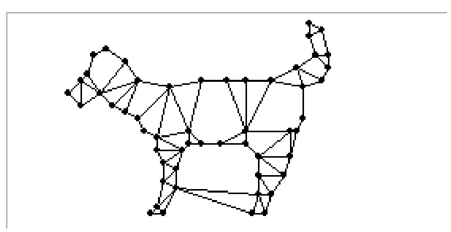
- Proximity graphs have been suggested as a convenient and relatively efficient way of generating a "primal sketch" of the shape of a point set—all of the graphs below can be created in $O(n \log n)$ time.



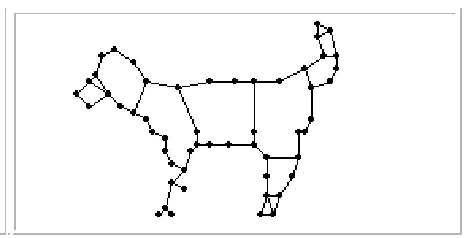
Delaunay Triangulation



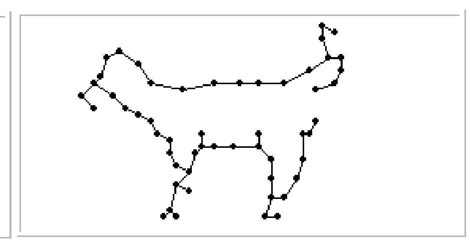
Gabriel Graph



Relative Neighbourhood Graph



Minimum Spanning Tree

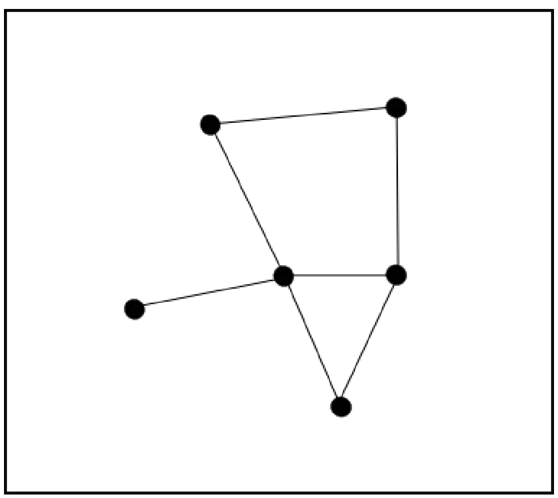


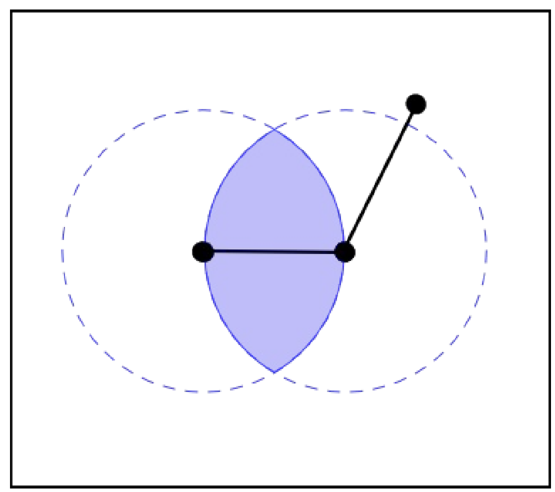
The RNG

- Let $\Lambda(p,q)$ be the intersection of the circle about p with a radius of dist(p,q) and the circle about q with a radius of dist(q,p). This is called a *lune*.

- The *relative neighborhood graph, RNG* (V) of a set of points V, is the graph that has an edge (p,q) if and only if the intersection of $\Lambda(p,q)$ and V is empty.

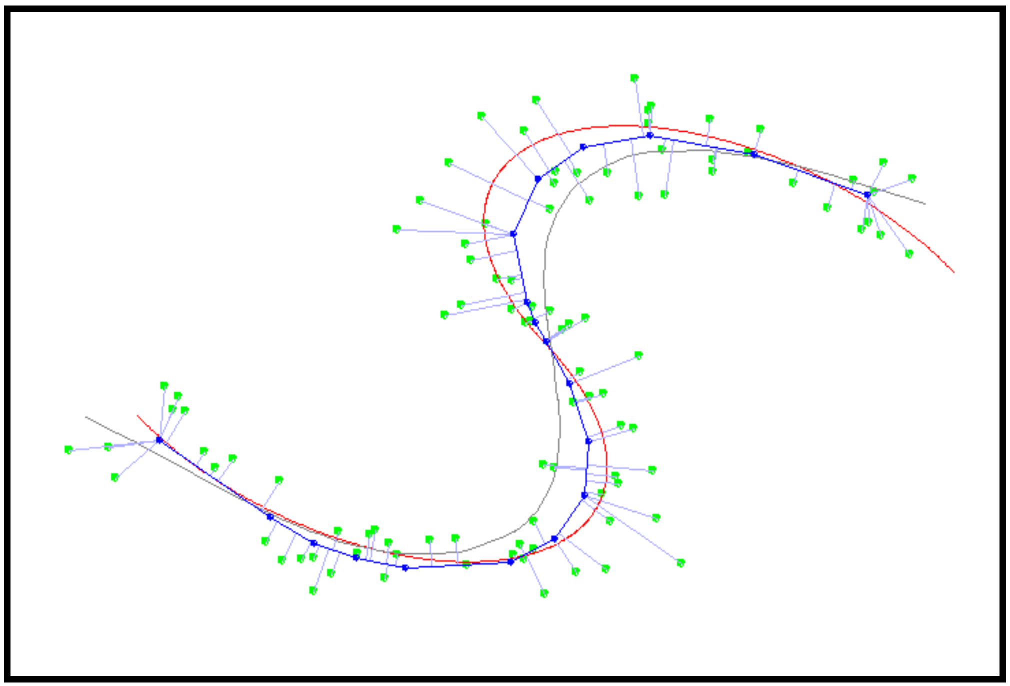
- You can think of the *RNG* as a graph which connects each point to its nearest neighbors in "each direction."





Principal Curves

- Principle curves are smooth curves that pass through the “middle” of a set of points (or a distribution)—“continuous curves of a given length which minimize the expected squared distance between the curve and points of the space randomly chosen according to a given distribution.” [Kegl, et al., 2000]
- Non-linear generalization of *principal components*.



green = data; red = generator curve;
gray = HS principal curve; blue = KKLZ principal curve

Selected Prior Work

- Hastie, Stuetzle, 1989.
  - $O(n^2)$
- Tibshirani, 1992.
- Kegl, Krzyzak, Linder, Zeger, 2000. (k-segments)
  - $O(n^2)$
  - $O(n^{5/3})$ with standard assumptions
  - adding more than one vertex at a time $O(n^{5/3})$ $O(nk \log k)$
  - setting k constant $O(nk)$
- Singh, Cherkassy, Papanikolopoulos, 2000. (Self-Organizing Map)
- Verbeek, Vlassis, Kröse, 2002 (Local principal components).
  - $O(kn^2)$

My Approach

- Similar to Dr. Singh's approach, I use a proximity graph to initialize the topology of the skeleton. However, I conducted most of my experiments using relative neighborhood graphs instead of minimum spanning trees.
- Similar to the original HS principal curve algorithm, I performed a perpendicular linear regression and projection on the n nearest points along the graph to update the position of points on the graph.
- Using an RNG to create the initial primal sketch has two main advantages:
    - It allows the final skeleton to have a much more complicated topological structure than just a curve, including loops, self-intersections, and branches.
    - The RNG of a planar point set can be found in $O(n \log n)$ time, so we can get a rough approximation of the shape very quickly.

...and here it is! Or rather, here they are (2 versions).

Problems!

- This algorithm may not converge! In fact, it some cases it definitely doesn't.
- There are several parameters that must be hand-tuned.
  - For version 1 and 2:
    - The number of nearest points along the RNG to consider when doing the local smoothing.
    - The number of iterations must be specified explicitly, since I haven't developed convergence criteria (and convergence may not even happen).
    - The smoothing function must also be specified. (I have deviated slightly from HS here, using a function the gives point near the current point more weight, and points farther away less weight.)
  - In addition, version 2, takes another parameter: the number of steps to be taken along the previous RNG when constructing new "local" RNGs.
- Supposing the algorithm does converge, the resulting graph may need to be trimmed to get rid of edges that are "too long" relative to the rest.

Provisional Complexity

Version 1 (per iteration), for a fixed number *m* of points along the graph to be used in the local smoothing step.

Initialize the RNG of the point set $O(n \log n)$

Then, for each iteration:

1. Find the RNG of the current graph $O(n \log n)$
2. For each point $O(nm^2)$
   1. Find the *m* nearest points along the graph. Assuming that the worst possible case is a triangular grid of points, and traversing the graph using a BFS strategy $O(m^2)$
      (This is very much a back-of-the envelope estimate—I haven't worked out the proof in detail; it could be better or worse.)
   2. Regression/projection $O(m)$
   3. Iterate again

Provisional Complexity

So the algorithm has a *provisional* complexity of $O(n(m^2+\log n))$. This gives $O(n \log n)$ for fixed m, but this is cold comfort, since m will need to grow as n does in order for the curve to maintain a reasonable level of smoothness.

Also, this does no one any good until some type of convergence can be guaranteed.

References

- Hastie, T., Stuetzle, W., 1989. Principal curves. *J. Am. Stat. Assoc*. 84 (406), 502–516.
- Jaromczyk, J.W.,Toussaint, G.T., 1992. “Relative Neighborhood Graphs And Their Relatives”. *P-IEEE* 80, 1502-1517.
- Kegl, B., Krzyzak, A., Linder, T., Zeger, K., 2000. “Learning and design of principal curves”. *IEEE Trans. Pattern Anal. Machine Intell*. 22 (3), 281–297.
- Singh, R., Wade, M. C., Papanikolopoulos, N. P., 1998. “Letter-Level Shape Description by Skeletonization in Faded Documents”, *Proc. Fourth IEEE Workshop on Applications of Computer Vision*, 121-126.
- Singh, R., Cherkassky, V., Papanikolopoulos, N. P., 2000. “Self-organizing maps for the skeletonization of sparse shapes,” *IEEE Transactions on Neural Networks*, vol. 11, no. 1, 241–248.
- Tibshirani, R., 1992. “Principal curves revisited”. *Stat. Comput.* 2, 183–190.
- Verbeek, J.J., Vlassis, N., Krose, , 2002, “A *k*-Segments Algorithm for Finding Principal Curves”. *Pattern Recognition Letters* 23, 1009–1017.