TOPOLOGICAL PERSISTENCE ON A JORDAN CURVE

Ying Zheng, Steve Gu and Carlo Tomasi

Department of Computer Science
Duke University, Durham, NC USA 27708

ABSTRACT

Topological persistence measures the resilience of extrema of a function to perturbations, and has received increasing attention in computer graphics, visualization and computer vision. While the notion of topological persistence for piece-wise linear functions defined on a simplicial complex has been well studied, the time complexity of all the known algorithms are super-linear (e.g. $O(n \log n)$) in the size n of the complex. We give an $O(n)$ algorithm to compute topological persistence for a function defined on a Jordan curve. To the best of our knowledge, our algorithm is the first to attain linear asymptotic complexity, and is asymptotically optimal. We demonstrate the usefulness of persistence in shape abstraction and compression.

Index Terms— Topological Persistence, Algorithms

1. INTRODUCTION

Topological persistence [1] measures how resilient the local extrema of a function are to perturbations of the function values. The higher the topological persistence of a critical point, the more likely the point is to survive a perturbation of the shape or function. Because of this, persistence is also intimately related to the notion of stability in control theory.

Topological persistence and its computation have been studied in a general setting [2, 1]. In this paper, we focus on functions defined on a Jordan curve. This restriction has two benefits: First, it allows for a simple, self-contained definition of topological persistence. Second, it leads to a simple and efficient algorithm, whose linear-time performance is asymptotically optimal and improves – for the 1D case – upon the complexity of more general algorithms [1].

We give anecdotal examples of the potential usefulness of persistence for curve simplification, also known as shape abstraction [3, 4, 5, 6, 7, 8]. Curve simplification is useful for noise removal, efficient storage, and a simplified representation of shape contours as a basis for further reasoning [9, 10, 11]. Our algorithm yields high compression ratios through very fast computation. Our experiments are mainly proof-of-concept, but hint at the usefulness of our method in several applications, including at least shape compression and de-noising; stylized editing of hand-drawn curves; and the simplification of object boundaries in images.

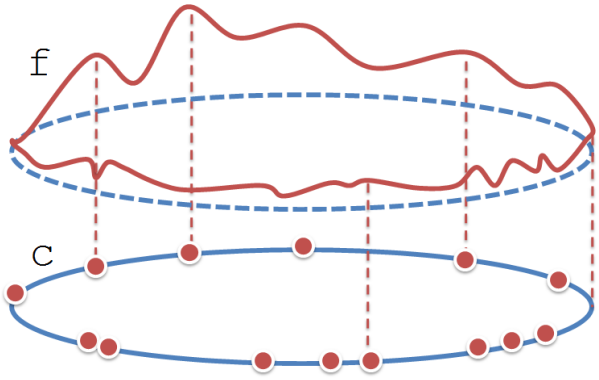

Fig. 1. A function f defined on a Jordan curve C. Red dots are local maxima of f.

2. FUNCTIONS ON A JORDAN CURVE

A Jordan curve is a non-self intersecting, continuous loop on the two dimensional plane. For images, we discretize a closed curve by a sequence of pixels $C = \langle p_0, ..., p_{n-1} \rangle$ with $p_i \neq p_j$ for any $0 \leq i \neq j < n$. Any cyclic permutation of the sequence represents the same curve. Given a function $f : C \to \mathbb{R}$ (Figure 1), a total ordering $\prec$ of the pixels in C is:

Definition 1 (Order $\prec$)**.** *We say $p_i \prec p_j$ if (1) $f(p_i) < f(p_j)$, or (2) $i < j$ and $f(p_i) = f(p_j)$.*

Thus, index values break ties between pixel values, so that local maxima and minima of f can be defined unambiguously. Since the curve C is a closed loop, it is most convenient to use *modular indices*: For any integer $0 \leq i < n$ and for any integer x, let $s(i, x) \triangleq (i + x) \mod n$ be the cyclic shift of i by x. We define local extrema as follows:

Definition 2 (Local Extrema)**.** *We say that $p_i \in C$ is a local maximum of f if $p_{s(i,-1)} \prec p_i$ and $p_{s(i,1)} \prec p_i$; p_i is a local minimum of f if $p_i \prec p_{s(i,-1)}$ and $p_i \prec p_{s(i,1)}$.*

By our definition, any curve C has at least one local extremum. The number of extrema is typically large if the func-

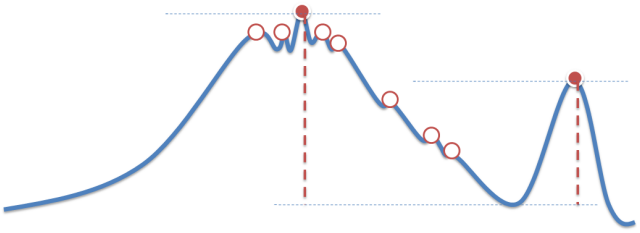

Fig. 2. Not all the local maxima of f are informative and most of them do not survive small perturbations of the function. In one dimension, topological persistence measures the stability of local extrema of f. The solid circles are the most persistent maxima, and the empty circles are not persistent ones.

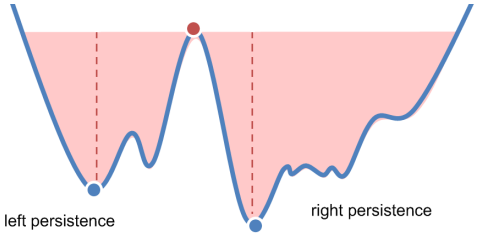

Fig. 3. The persistence of the local maximum at the red dot is the smaller of the maximal vertical variations – called the left and right persistence – in the two shaded regions.

tion f contains noisy measurements (Figure 2). For example, perturbing a constant function arbitrarily but within a small magnitude yields multiple local extrema, none of which are truly informative. This justifies ranking the stability of each extremum through the notion of topological persistence.

3. TOPOLOGICAL PERSISTENCE

In the one-dimensional case, saddles are irrelevant to the topology of sub-level sets, and topological persistence measures the lifetime of a local maximum or a local minimum of f. Since the local maxima of f become the local minima of $-f$, we only discuss how to define and compute the persistence of local maxima of f.

The notion of persistence can be understood from many perspectives and the most general version is introduced through homology theory. In this paper, we find it simpler to start with a notion related directly to a type of stability:

Definition 3 (δ-Stability)**.** *The point $p_i \in C$ is δ-stable if there exist $-n < l < 0 < r < n$ such that $p_{s(i,x)} \prec p_i$ for any $x \in [l, r]$ and $f(p_i) \geq \max\left\{f(p_{s(i,l)}), f(p_{s(i,r)})\right\} + \delta$.*

In words, the function f at a δ-stable maximum p_i is greater than its values in an interval around p_i, and greater by a margin of at least δ than the values at the endpoints of that interval. Clearly, all local maxima of f are at least 0-stable. We use δ-stability to define topological persistence:

Definition 4 (Topological Persistence)**.** *The topological persistence of $p \in C$, denoted $\varrho(p)$, is the maximal δ under which p is δ-stable. In other words, p is $\varrho(p)$-stable but is not $(\varrho(p) + \varepsilon)$-stable for any $\varepsilon > 0$.*

A physical picture gives an intuitive notion of persistence: The blue dot on the left in Figure 3 is the lowest point that a ball that rolls under gravity and without friction reaches when released from just to the left of the red dot. The blue dot on the right is the lowest point if the ball is released just to the right of the red dot. Persistence is the smaller of the two changes in altitude, each measured as positive.

From a signal processing point of view, the higher the persistence of a local maximum p, the more stable p. Persistence is a global quantity for each local maximum, since its computation can possibly involve the entire domain of the curve. Local extrema can be ranked by persistence, and low-persistence extrema can be removed, resulting in a simplification of the curve. Section 5 shows one way to "remove" low-persistence extrema. We turn to the computation of persistence first.

4. THE ALGORITHM

We introduce a linear-time algorithm to compute the persistence for all the local maxima of a function f on a Jordan curve. First, note that only maxima and minima of f are needed to this end, since values of the remaining pixels do not affect the computation. Second, according to the physical picture of persistence, we can compute the left and right persistence of each maximum separately, and then take the minimum of the two.

In order to compute, say, left persistence, we traverse the curve C clockwise starting from the global maximum of f. Each time, we check whether a pixel is a local maximum or minimum and update the left persistence of any maximum as it is visited. We use a stack S to store local maxima by decreasing function values (when read bottom to top), and bookmark the minimal values between consecutive local maxima in another stack V. Algorithm 1 computes the persistence of each maximum of f.

A step of the process for updating persistence is depicted in the first two panels of Figure 4. Note that *push()*, *pop()*, and *top()* are the standard stack operations.

4.1. Analysis

Although the algorithm has a *while* loop, each pixel appears in either stack at most twice. Therefore, the complexity of the algorithm is $O(n)$, where $n = |C|$ is the length of the curve. The correctness of the algorithm follows from the fact

Algorithm 1 Compute the persistence of local maxima of f.

```
Set persistence ϱ(p) ← +∞ for each local maximum of p;
Set the persistence of the global maximum p_i: ϱ(p_i) ←
max_{q∈C} f(q) − min_{q∈C} f(q);
for Δ ∈ {+1, −1} do
  Push the global maximum p_i of f to an empty stack S;
  Initialize an empty stack V to hold local minima of f;
  for j = 1 to n − 1 do
    Move to the next point: v ← p_{s(i,jΔ)};
    if v is a local minimum of f then
      V.push(v);
    else if v is a local maximum of f then
      u ← V.top();
      while S.top() ≺ v do
        S.pop(), V.pop();
        if V.top() ≺ u then
          u ← V.top();
        end if
      end while
      Update: ϱ(v) = min {ϱ(v), f(v) − f(u)};
      S.push(v), V.pop(), V.push(u);
    end if
  end for
end for
```

that whenever a local maximum is pushed into the stack, its left (or right) persistence is correctly computed by finding the lowest local minimum to the left (or right) side of it. This is realized by the *while* loop in Algorithm 1.

5. CURVE SIMPLIFICATION

Topological persistence is expected to be useful for curve simplification. In particular, we use persistence to determine the stable extrema in the curvature of C, and we approximate the curve between consecutive extrema by arcs of circles. This yields a cleaner, more compact, and typically faithful representation of the original curve, in which salient cusps are preserved. Our method requires three steps: computation of local convexity, removal of low-persistence extrema of convexity, and approximation of intervals between high-persistence extrema with arcs of circles. These steps are described next.

5.1. Measuring Local Convexity

Let p, q, r be three consecutive points on the curve C, traversed clockwise. We define the *local convexity* $f(q)$ at q as the clockwise rotation angle (in $[0, 2\pi]$) between the rays $\overrightarrow{qp}$ and $\overrightarrow{qr}$. We say that p is convex if $f(p) < \pi$ and concave if $f(p) > \pi$. For greater resilience to noise, we use a method similar to the Parzen window to estimate the convexity: we first estimate the convexity at p_i from the triples (p_{i-1}, p_i, p_{i+1}), (p_{i-2}, p_i, p_{i+2}), ..., (p_{i-k}, p_i, p_{i+k}), and then average the results with Gaussian weights (smaller weights for more distant points).

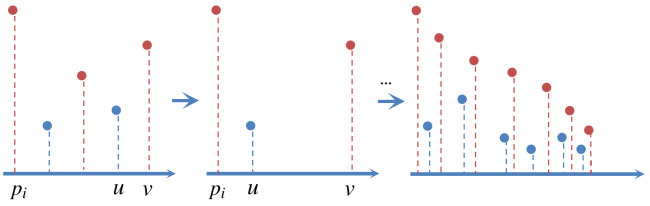

Fig. 4. Left to middle: The change of state resulting from the visit of local maximum v. Red dots show the contents of stack S, and blue dots those of stack V. The top of each stack is on the right. Right: The final states in the two stacks. The remaining maxima end up sorted by decreasing function values in S (bottom to top of stack).

5.2. Persistence-Based Simplification

We preserve only the persistent extrema (local maxima and minima) of the local curvature function $f : C \to [0, 2\pi]$ in order to give a compact representation of the shape. Specifically, we preserve a local maximum $p \in C$, if and only if:

$$\varrho(p) \geq \max \left\{ \alpha[\max_{q \in C} f(q) - \min_{q \in C} f(q)], \beta \right\}$$

where $\alpha, \beta \in [0, 1]$ are parameters and $\max_{p \in C} f(p) - \min_{p \in C} f(p)$ is the persistence of the global maximum of f. The first term in the braces specifies that the persistence of the local maximum p must be large enough compared to the persistence of the global maximum, and the second term β handles degenerate case properly. A degenerate curve has almost equal curvature everywhere (e.g. a not so perfect circle). In our experiments, we fix $\alpha = \beta = 0.2$. The global maximum is always preserved, since otherwise the curve may be simplified to nothing. Local minima of f are handled by retaining the persistent maxima of $-f$.

5.3. Curve Approximation via Partial Circles

Let $C = \langle p, p_1, \ldots, p_m, q \rangle$ be the curve segment bounded by two local extrema p and q. We fit a partial circle passing through p and q exactly. The center of the circle has to lie on the perpendicular bisector of $\overline{pq}$. Let O be the bisecting point. We first subtract O from each point in C so that O becomes the new origin of the coordinate system. The center of the fitting circle is therefore a point of the form $c = \lambda u$, where u is a unit vector perpendicular to the segment $\overline{pq}$. To find λ, we observe that from Pythagoras's theorem the radius of the circle is $r = \sqrt{\lambda^2 + \frac{\|p-q\|^2}{4}}$. We thus look for a value of λ that satisfies the system of m equations

$$\|p_i - \lambda u\| = r \quad \text{for} \quad i = 1, \ldots, m \, .$$

Squaring both sides and rearranging terms yields:

$$u^T p_i \lambda = \frac{\|p_i\|^2}{2} - \frac{\|p-q\|^2}{8}$$

and stacking all these equations into a matrix results in the following over-constrained system of linear equations in λ:

$$\begin{pmatrix} u^T p_1 \\ \vdots \\ u^T p_m \end{pmatrix} \lambda = \begin{pmatrix} \frac{\|p_1\|^2}{2} - \frac{\|p-q\|^2}{8} \\ \vdots \\ \frac{\|p_m\|^2}{2} - \frac{\|p-q\|^2}{8} \end{pmatrix}$$

The least-squares solution can be computed in closed form:

$$\lambda = \frac{4u^T \sum_{i=1}^m \|p_i\|^2 p_i - \|p-q\|^2 u^T \sum_{i=1}^m p_i}{8 \sum_{i=1}^m (u^T p_i)^2} .$$

5.4. Results

Sample results are shown in Figure 5. Our abstraction process preserves only 1% of the original set of pixels! This great reduction comes at the price that not all the abstracted shape resemble the original ones closely. However, the "gist" of the shapes are by-and-large retained. Simplification is very inexpensive computationally: Given binary images of resolution 320×240, the shape abstraction process runs in more than 200 images per second on a laptop.

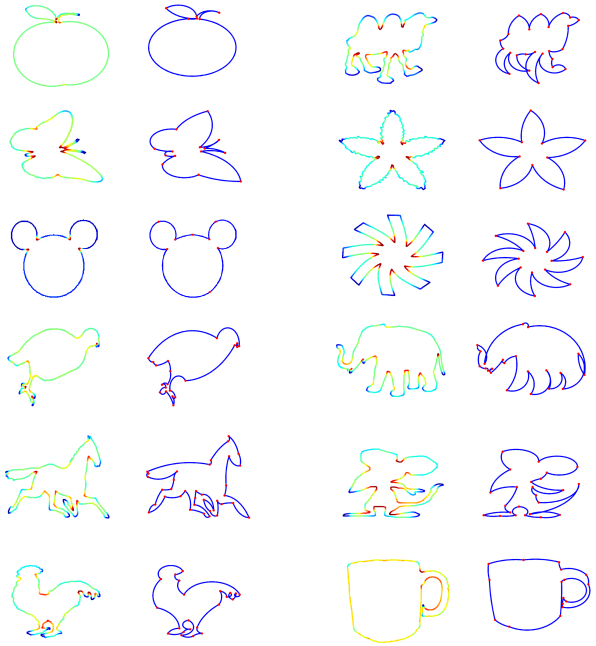

Fig. 5. Shape abstraction. The first and third column show the original shape with color representing the (signed) curvature. The warmer the color, the higher the curvature. The second and the last column show the abstracted shape using typically one percent of the total set of points! Note that the abstraction process preserves details. Best viewed when enlarged.

6. CONCLUSIONS

We present an efficient algorithm to compute the topological persistence of a function defined on a Jordan curve. To the best of our knowledge, we are the first to attain linear asymptotic complexity in calculating the topological persistence in 1D case, and promising results are demonstrated in shape abstraction. Future work will consider the efficient computation of topological persistence in higher dimensions.

Acknowledgement: This work is supported by the Army Research Office under Grant No. W911NF-10-1-0387.

7. REFERENCES

[1] H. Edelsbrunner, D. Letscher, and A. Zomorodian, "Topological persistence and simplification," *Discrete & Comp. Geometry*, vol. 28, no. 4, pp. 511–533, 2002.

[2] H. Edelsbrunner and J. Harer, "Persistent homology — a survey," in *Contemporary Mathematics*, 2008.

[3] H. Hoppe, "Progressive meshes," in *SIGGRAPH*, 1996.

[4] J. Popovic and H. Hoppe, "Progressive simplicial complexes," in *SIGGRAPH*, 1997.

[5] M. Garland and P. Heckbert, "Surface simplification using quadric error metrics," in *SIGGRAPH*, 1997.

[6] P. Agarwal, S. Peled, N. Mustafa, and Y. Wang, "Near-linear time approximation algorithms for curve simplification," *Algorithmica*, vol. 42, no. 3-4, pp. 203–219, 2005.

[7] J. Hershberger and J. Snoeyink, "An o(n log n) implementation of the douglas-peucker algorithm for line simplification," in *Symposium on Computational Geometry*, 1994, pp. 383–384.

[8] D. Douglas and T. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," in *The Canadian Cartographer*, 1973, pp. 112–122.

[9] D. DeMenthon, V. Kobla, and D. Doermann, "Video summarization by curve simplification," in *ACM Multimedia*, 1998, pp. 211–218.

[10] H. Etou, Y. Okada, and K. Niijima, "Feature preserving motion compression based on hierarchical curve simplification," in *ICME*, 2004, pp. 1435–1438.

[11] X. Mi, D. DeCarlo, and M. Stone, "Abstraction of 2d shapes in terms of parts," in *NPAR*, 2009, pp. 15–24.