Report for CSE 5339 2018 — (OTMLSA) Optimal Transport in Machine Learning and Shape Analysis

An Optimal Transport Approach to Robust Reconstruction and Simplification of 2D Shapes

Kritika Singhal

Abstract

This is the work of Fernando de Goes, David Cohen-Steiner, Pierre Alliez and Mathieu Desbrun. Given a point set $\mathcal{S}$ with Dirac masses, they provide an algorithm for constructing a coarse simplicial complex $\mathcal{T}$, such that the measure on $\mathcal{S}$ is well approximated by a linear combination of uniform measures on the edges and vertices of $\mathcal{T}$.

1 Introduction

Shape reconstruction from some input pointset is a problem encountered in many areas such as face recognition, biology, image processing et cetera. In this paper [7], shape reconstruction and simplification is seen as an optimal transport problem between measures. In particular, the input points are considered as Dirac masses, and the output is a 1-dimensional simplicial complex seen as the support of a piecewise uniform measure. The use of optimal transport allows robustness to large amounts of noise and outliers, and preserves boundaries and sharp features. The methods available prior to this work could handle presence of noise [2] [3], outliers [4] [5], features and boundaries individually, but none of them could handle all of these concurrently.

2 Preliminaries

We provide some definitions that are used in the rest of the document.

Definition (Delaunay Triangulation)**.** *A Delaunay triangulation for a finite set of points P on a plane is a triangulation $DT(P)$ such that no point in P lies inside the circumcircle of any triangle in $DT(P)$. For a point set P with $|P| = n$, such a triangulation can be computed in time $O(n \log n)$ [1].*

Definition (Half-edge)**.** *A half-edge is a directed line segment with an origin vertex and a destination vertex.*

Definition (One-ring of a point)**.** *The one-ring of a point x in a triangulation $\mathcal{T}$ is the set of all vertices adjacent to x in $\mathcal{T}$.*

Definition (Kernel of a polygon)**.** *The kernel of a polygon Q is a non-empty set K of points in the interior of Q such that there exists a line segment from every point in K to every other point in Q lying entirely inside Q. The kernel of a convex polygon is its interior.*

Definition (Flippable Edge)**.** *An edge e in a triangulation $\mathcal{T}$ is called flippable if its end points and its two opposite vertices form a convex quadrilateral.*

3 Optimal Transport Formulation

Let $\mathcal{S} = \{p_1, p_2, \ldots, p_n\}$ denote the input point set. Each point p_i is seen as a Dirac measure μ_i centered at p_i and having mass m_i. Therefore, the point set is considered as a measure $\mu = \sum_i \mu_i$ with $\mu_i(p_j) = m_i \delta_{ij}$. We assume that we are given a triangulation $\mathcal{T}$ of $\mathcal{S}$ and a *point to simplex assignment* that maps every point p_i to either a vertex v or an edge e of $\mathcal{T}$. The assignment will be described later. Every vertex v of $\mathcal{T}$ is seen as a Dirac measure and every edge e of $\mathcal{T}$ is a uniform 1D measure defined over the edge e.

Let π denote the transport plan satisfying the point to simplex assignment and let $W_2(\pi)$ denote its transport cost. For every vertex v of $\mathcal{T}$, let $\mathcal{S}_v$ denote the set of points of $\mathcal{S}$ assigned to the vertex v, and for every edge e of $\mathcal{T}$, let $\mathcal{S}_e$ denote the points of $\mathcal{S}$ assigned to the edge e. We assume that the sets $\{\mathcal{S}_e \mid e \in \mathcal{T}\} \cup \{\mathcal{S}_v \mid v \in \mathcal{T}\}$ are mutually disjoint, and $\cup_{v \in \mathcal{T}} \mathcal{S}_v \cup \cup_{e \in \mathcal{T}} \mathcal{S}_e = \mathcal{S}$. Let M_v denote the total mass of points in $\mathcal{S}_v$ and let M_e denote the total mass of points in $\mathcal{S}_e$. Then, we have $\sum_{v \in \mathcal{T}} M_v + \sum_{e \in \mathcal{T}} M_e = \sum_i m_i$. We formulate the cost of assigning a point p_i to either an edge or a vertex of $\mathcal{T}$.

Points to vertex - For a vertex $v \in \mathcal{T}$, the cost to transport the measure on $\mathcal{S}_v$ to the Dirac measure centered on v with mass M_v is given by

$$W_2(v, \mathcal{S}_v) = \sqrt{\sum_{p_i \in \mathcal{S}_v} m_i ||p_i - v||^2}.$$

Points to edge - The measure on an edge $e \in \mathcal{T}$ is the uniform measure of value $\frac{M_e}{|e|}$, where $|e|$ denotes the length of edge e. The transport plan here is decomposed into a normal and a tangential component to e. The normal component is derived from the orthogonal projection of $\mathcal{S}_e$ onto e. Precisely, for every $p_i \in \mathcal{S}_e$, let q_i denote the orthogonal projection of p_i onto e. The transport cost of the normal plan is given by

$$N(e, \mathcal{S}_e) = \sqrt{\sum_{p_i \in \mathcal{S}_e} m_i ||p_i - q_i||^2}.$$

The tangential plan is obtained as follows: the projected points $\{q_i\}$ are sorted along the edge e. By abuse of notation, let q_i denote the points in sorted order. The edge e is partitioned into $|\mathcal{S}_e|$ segment bins, with the i-th bin having length $l_i = (m_i/M_e)|e|$. Here, m_i is the mass of point p_i whose projection is q_i. For every $1 \leqslant i \leqslant |\mathcal{S}_e|$, we set a 1D coordinate axis along the edge e with origin at the center of the i-th bin, and denote by c_i the coordinate of q_i in this coordinate axis. The tangential cost t_i of p_i is given by

$$t_i = \frac{M_e}{|e|} \int_{-l_i/2}^{l_i/2} (x - c_i)^2 dx = m_i \left(\frac{l_i^2}{12} + c_i^2 \right).$$

We add these costs to get the tangential component of the optimal transport cost for the entire edge e:

$$T(e, \mathcal{S}_e) = \sqrt{\sum_{p_i \in \mathcal{S}_e} m_i \left(\frac{l_i^2}{12} + c_i^2 \right)}.$$

The decomposition of the transport plan into a normal and tangential component ensures that boundaries and sharp features are preserved. Refer to [7] for further details. The total cost to transport $\mathcal{S}$ to $\mathcal{T}$ through the transport plan π is given by

$$W_2(\pi) = \sqrt{\sum_{e \in \mathcal{T}} [N(e, \mathcal{S}_e)^2 + T(e, \mathcal{S}_e)^2] + \sum_{v \in \mathcal{T}} W_2(v, \mathcal{S}_v)^2}.$$

3.1 Point to simplex assignment

Given a triangulation $\mathcal{T}$, we now describe the method of assigning points of $\mathcal{S}$ to edges and vertices of $\mathcal{T}$. Each point $p_i \in \mathcal{S}$ is first assigned temporarily to the closest edge of $\mathcal{T}$ (breaking ties arbitrarily). This results into a partition of $\mathcal{S}$ into subsets $\mathcal{S}_e$. Now for every edge e, the points in $\mathcal{S}_e$ are either kept assigned to e or every point of $\mathcal{S}_e$ is assigned to its closest endpoint of e. The assignment that minimizes the optimal transport cost, as described above, is chosen.

4 Algorithm

We now give the pseudocode of the algorithm.

Input: point set $\mathcal{S} = \{p_1, p_2, \ldots, p_n\}$.
1: Construct Delaunay triangulation $\mathcal{T}_0$ of $\mathcal{S}$.
2: Compute initial plan π_0 from $\mathcal{S}$ to $\mathcal{T}_0$.
3: $k \leftarrow 0$.
4: **repeat**
5: Pick best half-edge $e = (x_i, x_j)$ to collapse (Simplification)
6: Set $\mathcal{N}_{i,1}$ the 1-ring of x_i
7: Create $\mathcal{T}_{k+1}$ by merging x_i onto x_j
8: $\pi'_{k+1} := \pi_k$ with local reassignments in $\mathcal{N}_{i,1}$
9: Optimize position of vertices in $\mathcal{N}_{i,1}$ (Relocation)
10: $\pi_{k+1} := \pi'_{k+1}$ with local reassignments in edges adjacent to $\mathcal{N}_{i,1}$
11: $k \leftarrow k + 1$
12: **until** (desired vertex count)
13: Filter edges based on relevance(optional)
Output: vertices and edges of $\mathcal{T}_m$

The first step of the algorithm is constructing a Delaunay triangulation $\mathcal{T}_0$ of the input point set $\mathcal{S}$. The initial transport plan π_0 is the trivial assignment of every point to its corresponding vertex, with the cost of π_0 being zero.

4.1 Simplification

Simplification of a triangulation $\mathcal{T}_k$ is performed through a series of half-edge collapses. Collapsing an edge changes a triangulation $\mathcal{T}_k$ to a triangulation $\mathcal{T}_{k+1}$, and thus changes the cost of the transport plan by $\delta_k = W_2(\pi_{k+1}) - W_2(\pi_k)$. Since the goal is to minimize increase in total cost, edge collapses are performed in increasing order of δ. To this end, all feasible collapses are initially simulated, and their associated δ is added to a dynamic priority queue sorted in increasing order. Each edge collapse is done by repeatedly popping from the queue the next edge to collapse, performing the collapse, updating the transport plan and cost, and updating the priority queue. We note that updating the transport plan involves only the edges in the one-ring of the removed vertex and updating the priority queue is required only for edges incident to the modified one-ring.

4.2 Collapsing half-edges

We note that collapsing a half-edge (x_i, x_j) can result in the edges incident to x_i intersecting the remaining edges in points different from the input points. This is called a fold-over in the triangulation. We do not want this to happen, since we want the vertices of the triangulation to be from the input points. Thus, we say that a half-edge is *collapsible* if its collapse creates neither overlaps nor fold-overs in the triangulation ([7], Figure 7). Every half-edge is made

collapsible by the following procedure: let (x_i, x_j) denote the half-edge we want to collapse. Let P_{x_i} denote the counter-clockwise oriented polygon formed by the one-ring of x_i, and let K_{x_i} denote its kernel. We say that an edge $(a, b) \in P_{x_i}$ is *blocking* x_j if the triangle (x_j, a, b) has clockwise orientation. We note that blocking edges are exactly the edges that result in the creation of fold-overs when collapsing (x_i, x_j). A blocking edge (a, b) is removed from P_{x_i} by flipping either (a, x_i) or (b, x_i). We observe that one of these edges is always flippable. For every blocking edge, flipping continues until there are no blocking edges in P_{x_i}.

4.3 Vertex Relocation

The triangulations obtained by edge collapses have their vertices on the input points. However, the presence of noise and missing data make interpolated triangulations poorly adapted to recover features. In order to overcome this problem, vertex relocation is performed after every edge collapse. The square of the transport cost associated with a vertex v of $\mathcal{T}$, and the normal cost associated with edges adjacent to v is given by

$$\sum_{p_i \in \mathcal{S}_v} m_i ||p_i - v||^2 + \sum_{b \in \mathcal{N}_1(v)} \sum_{p_i \in \mathcal{S}_{(v,b)}} m_i ||p_i - q_i||^2.$$

Here, $\mathcal{N}_{1,v}$ denotes the one-ring of v. The optimal position v^* of v is computed by equating the gradient of the above expression to zero. Precisely, if $q_i = (1 - \lambda_i)v + \lambda_i b$, $0 \leqslant \lambda_i \leqslant 1$, then

$$v^* = \frac{\sum_{p_i \in \mathcal{S}_v} m_i p_i + \sum_{b \in \mathcal{N}_{1,v}} \sum_{p_i \in \mathcal{S}_{(v,b)}} m_i (1 - \lambda_i)(p_i - \lambda_i b)}{M_v + \sum_{b \in \mathcal{N}_{1,v}} \sum_{p_i \in \mathcal{S}_{(v,b)}} m_i (1 - \lambda_i)^2}.$$

After relocating v, a new transport plan π_{k+1}, and its associated cost are computed.

4.4 Edge Filtering

The presence of noise and outliers can lead to a few undesirable edges e with positive M_e in the triangulation. Therefore, a notion of relevance r_e is defined for such edges as

$$r_e = \frac{M_e |e|^2}{N(e, \mathcal{S}_e)^2 + T(e, \mathcal{S}_e)^2}.$$

In the triangulation, only the edges with relevance above a certain threshold are kept. Performing edge filtering is optional, and depends on the application.

5 Numerical Experiments

The authors use CGAL [8] library to implement the shape reconstruction algorithm. The algorithm takes as input point sets with mass attributes, and a desired vertex count for the output. In order to speed up the computations, instead of using the entire input point set, a random sample of the point set is used and a Delaunay triangulation of this sample is constructed. In addition, for the simplification process, initially half-edge collapses for a random set of around 10 edges are simulated, and the collapse with the smallest δ is performed. When the number of vertices reached is 5 times the vertices in the final count, the exhaustive search for the best half-edge collapse is resumed. Vertex relocation is helpful only when a sufficient number of input points are assigned to the edges of the triangulation. Since this number is small in the beginning of the algorithm, vertex relocation is performed only after the last 100 edge collapses.

Please refer to Section 4 of [7] for details of the numerical experiments. The algorithm performs well in recovering boundaries and sharp features. The running time of the algorithm is small, for example for 20,000 input points, using the speed up methods described in the last paragraph, it takes 51 seconds to obtain the final triangulation.

References

[1] G. Leach. "Improving Worst-Case Optimal Delaunay Triangulation Algorithms.". http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.56.2323&rep=rep1&type=pdf

[2] In-Kwon Lee. "Curve reconstruction from unorganized points." http://www.sciencedirect.com/science/article/pii/S0167839699000448.

[3] S-W Cheng et al. "Curve reconstruction from noisy samples." https://www.sciencedirect.com/science/article/pii/S0925772104001075.

[4] S. Fleishman et al. "Robust moving least-squares fitting with sharp features." http://www.sci.utah.edu/~shachar/Publications/rmls.pdf.

[5] Y. Song. "Boundary fitting for 2D curve reconstruction." https://link.springer.com/article/10.1007%2Fs00371-009-0395-4.

[6] C. Villani. Topics in Optimal Transportation. http://bookstore.ams.org/gsm-58.

[7] De Goes et al. "An Optimal Transport Approach to Robust Reconstruction and Simplification of 2D Shapes". http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8659.2011.02033.x/full.

[8] CGAL, Computational Geometry Algorithms Library. http://www.cgal.org