Recall

**Triangulations of point sets/polygons.** Recall what we’ve seen:

- Delaunay triangulation of point set in $\mathbb{R}^d$, $O(n \log n)$ algorithm in $\mathbb{R}^2$.

- $O(n)$ algorithm to triangulate any polygon in $\mathbb{R}^2$ (Chazelle’s hard algorithm)

**Applications and criteria** (this is the outline for the next lectures)

- angle criteria - for meshing

- length criteria: minimum weight triangulation

- constrained triangulations (when certain edge must be included)

- meshing - triangulations with Steiner points

- flip distance

**today**

- morphing

- curve and surface reconstruction

- medial axis and straight skeleton
Application of Triangulations: Morphing

500 Years of Female Portraits in Western Art

https://www.youtube.com/watch?v=nUDIoN-_Hxs

Choose corresponding points, and make the “same” triangulation on both. Then morph the triangles.

Alexei Efros
In this homework, we are going to morph an image into another. We also create a morph video sequence for the morph processing. To morph an image A to image B, we need to define some corresponding feature points on two images, then we morph a face to another face and we can also morph an object to another.

The feature points define correspondence between two images. Generally, the more points, the better the morph. We create a morphed hand by morph left hand and right hand to align and warp the sampled points, the morphed images are aligned.

The morphing algorithm is composed of the 3 steps: 1. Defining correspondences 2. Computing a triangulation 3. Create the morph.

For many faces, then an average face is calculated by applying the morphing algorithm. In this part, we also masculinize and feminize a face with the mean man and woman faces.

Overview

PART A

Result

Once we got the triangulation for two faces, we will define the mapping between two triangles. The triangulation of different faces should be the same, we compute the triangulation for the mean of the two point sets to decrease the potential deformations.

The morphing algorithm is composed of the 3 steps: 1. Defining correspondences 2. Computing a triangulation 3. Create the morph.

For each triangle, we compute the inverse affine transform, use it to look up the color associated to each pixel in both images, and compute their weighted average. The feature points define correspondence between two images. Generally, the more points, the better the morph. We use a 6-DOF affine matrix for triangulation transformation. These matrices must be computed independently for each pair of triangles.

To smoothly generate the morphing animation, we use a sigmoid function to control the level of dissolution according to the warp degree.

In part A we manually select facial features, then warp the image shape and dissolve the colors. In part B we automatically select facial features to caricature a person's face.

http://vision.gel.ulaval.ca/~jflalonde/cours/4105/h16/tps/results/ecrb/index.html

More result

The more result
Two aspects to this morphing approach:

1. how to triangulate “compatibly”
2. how to morph compatible triangulations

Compatible triangulations

Given two (unlabelled) point sets, triangulate them the “same” way.

Two triangulations are *compatible* if we can map the points $p$ of the first set to points $f(p)$ of the second set (one-to-one, onto) s.t. $pqr$ is a clockwise triangle iff $f(p)f(q)f(r)$ is a clockwise triangle.
Compatible triangulations

an interesting open side question:

**Conjecture**: Given two points sets each with n points total, and h points on the convex hull, they have a compatible triangulation.

This assumes no 3 points collinear (otherwise false).


[https://doi.org/10.1016/S0304-3975(02)00428-5](https://doi.org/10.1016/S0304-3975(02)00428-5)

also see Devadoss O'Rourke book

back to what’s relevant for morphing:

**Theorem.** Two simple polygons on n vertices can be compatibly triangulated with Theta(n^2) Steiner points.


[https://doi.org/10.1016/0925-7721(93)90028-5](https://doi.org/10.1016/0925-7721(93)90028-5)
compatible triangulations of polygons

(a) (c) (e)

(b) (d) (f)

Craig Gotsman, Vitaly Surazhsky

compatible triangulations using 1 Steiner point inside and 1 Steiner point outside
Morphing compatible triangulations

The face morphing projects just use a linear mapping of each triangle.

such morphs do not preserve planarity in general

Planarity preserving morphs

- existence first proved by Cairns, 1944

- solution by Floater, Gotsman, Surazhky 2000, using Tutte’s graph drawing algorithm. No explicit vertex trajectories.

- piecewise linear solution


https://doi.org/10.1137/16M1069171
morphing using Floater, Gotsman, Surazhky method

\[ t=0 \quad t=0.25 \quad t=0.5 \quad t=1 \]

Their algorithm takes a time \( t \in [0, 1] \) and produces the polygon at time \( t \).
They use many values of \( t \) to make it look continuous.

Craig Gotsman, Vitaly Surazhsky
Curve and surface reconstruction

Approximately 12 Minutes
Hundreds of Thousands Points Connected

Original > Point Cloud > 3D Model
Curve and surface reconstruction

digital Michaelangelo project
Curve and surface reconstruction

alpha-shapes and alpha-hulls

pushing lines against a point set gives the convex hull
line = infinite radius circle

pushing discs of smaller radius gives more refined “shape” and detects holes

the alpha-hull, alpha = disc radius
When alpha is small, the points remain isolated; when alpha is large the alpha-hull approaches the convex hull.
alpha-shapes and alpha-hulls

issues:
- what is the “right” value of alpha?
- if points are not uniform then no single value of alpha will work.

cited by 1939

d https://doi.org/10.1145/174462.156635
Crust Algorithm for surface reconstruction

in 2D this is curve reconstruction

figures from Devadoss, O’Rourke

points on the curve must be sufficiently dense in order to reconstruct the curve


[https://doi.org/10.1006/gmip.1998.0465](https://doi.org/10.1006/gmip.1998.0465)
A. Lubiw, U. Waterloo

Lecture 13: Triangulations, continued

Voronoi diagram

input points

Delaunay triangulation of original points $S$ + Voronoi vertices

edges with both endpoints in $S$
Medial axis of a convex polygon = Voronoi diagram of edges of polygon

= locus of centers of circles inside polygon that touch boundary at 2 or more points (centers of maximal inscribed discs)
Medial axis of a convex polygon = Voronoi diagram of edges of polygon

= grow the vertex angle bisectors by shrinking the polygon. The trajectories of the vertices form the medial axis.

every edge of the medial axis is a bisector of two polygon edges
Medial axis of a convex polygon = Voronoi diagram of edges of polygon

There is an $O(n)$ time algorithm.
Here is a simpler $O(n \log n)$ algorithm:

Maintain
- for each pair of consecutive edges their angle bisector ray
- for each pair of consecutive rays, their meeting point and time $t$ when they meet
- a priority queue of these times.

Update:
- find & delete min time, output Voronoi vertex
- delete $e_i$ and 2 rays of
- find new ray bisector of $e_{i-1}$ and $e_{i+1}$
- compute its meeting point with its two neighbour rays.
Medial axis of a non-convex polygon = locus of centers of maximal inscribed discs

Figure 5.6: The central arc lies on the parabola determined by the vertex \( v \) and the edge \( e \), where the maximal disks centered on that arc touch \( e \) and \( v \).

can be found in time \( O(n) \)
A physical model for medial axis

- Imagine the polygon is drawn on the prairie, and you light fires along the boundary. Medial axis = points where fire is quenched (fire meets other fire)

- pouring sand

Voronoi diagram
A physical model for medial axis
Applications of medial axis

Blum transform for shape recognition

character recognition shape matching

**Straight Skeleton** — similar to medial axis but avoids curved sections

Grow the vertex angle bisectors by shrinking the polygon. The trajectories of the vertices form the straight skeleton.

For a convex polygon, this is the same as the medial axis

But for a non-convex polygon, it is not the same:
**Straight Skeleton** — similar to medial axis but avoids curved sections

Difference between medial axis and straight skeleton — only for non-convex polygons:

offset curve with mitred caps
Straight skeleton algorithms

idea of previous algorithm gives $O(n^2 \log n)$ because the next ray intersection need not be between consecutive rays

improvements:

$O(n^{8/5} + \varepsilon)$ for any fixed $\varepsilon > 0$


https://doi.org/10.1007/PL00009479

$O(n^{4/3} + \varepsilon)$ time for any $\varepsilon > 0$


https://doi.org/10.1007/s00454-014-9625-2
Straight skeleton applications: designing roofs

How to fit a roof to these walls?
Straight skeleton application: fold and cut problem

Fold and Cut Theorem. For any (slightly perturbed) polygon on a piece of paper there is a flat folding of the paper that puts all the polygon edges on one line and puts the inside and outside of the polygon on opposite sides of the line.

solution for triangle:
What does a solution to fold and cut look like at the polygon boundary?

MUST use bisector at each vertex. MAY use perpendiculars on any edge.

Thus, use straight skeleton. MAY use perpendiculars on any edge and we need some of these to get flat folding.
Example.

All folds except the pink ones are straight skeleton folds.

In degenerate cases, this bouncing can be infinite. This is why we may need to perturb the input polygon slightly.
fold-and-cut examples


[http://erikdemaine.org/foldcut/](http://erikdemaine.org/foldcut/)
Summary

- compatible triangulations and morphing
- curve and surface reconstruction
- medial axis (Voronoï diagram of edges)
- straight skeleton

References

- papers and books listed throughout