2-D Shape Analysis

Ming Zhang

COS598b Geometric Modeling
Spring 2000

Comparing polygonal shapes using turning function

• A standard method is represent polygon by a list of vertices
• An alternative way is to define the turning function $\theta_A(s)$
• The turning function measures the angle $v$ of tangent as a function of arc length $s$
• The turning function increases with left hand turns and decrease with right hand turns
Turning function (continued)

- For convex polygon A, $\theta_A(1) = \theta_A(0) + 2\pi$
- It is piecewise constant for polygons
- It is invariant under translation and scaling
- It may be unstable under non-uniform noise
A polygon distance function

• Two polygons A and B with turning function $\theta_A(s)$ and $\theta_B(s)$, define L_p Distance between $\theta_A(s)$ and $\theta_B(s)$

$$\delta_p(A, B) = \| \theta_A - \theta_B \|_p = (\int_0^1 |\theta_A(s) - \theta_B(s)|^p \, ds)^{\frac{1}{p}}$$

• The distance function is sensitive to both the rotation of polygons and choice of reference point

Distance function (continued)

• It makes more sense to consider the minimum distance over all the choices

$$d_p(A, B) = (\min_{t \in [0, 1]} \int_0^1 |\theta_A(s + t) - \theta_B(s) + \theta|^p \, ds)^{\frac{1}{p}}$$

$$= (\min_{t \in [0, 1]} D_{p}^{A, B}(t, \theta))^{\frac{1}{p}}$$

$$D_{p}^{A, B}(t, \theta) = \int_0^1 |\theta_A(s + t) - \theta_B(s) + \theta|^p \, ds$$
Distance function (continued)

- To minimize, \( h(t, \theta) = D_2^{A,B}(t, \theta) \) the best value of \( \theta \) is given by

\[
\theta(t) = \int_0^1 (g(s) - f(s + t))ds = \alpha - 2\pi t
\]

- where \( \alpha = \int_0^1 g(s)ds - \int_0^1 f(s)ds \)

and \( f(s) = \Theta_A(s), g(s) = \Theta_B(s) \)

Critical events

- Critical event is a value of \( t \) where breakpoints of \( f \) collides with breakpoints of \( g \)
- There are \( mn \) critical events for \( m \) breakpoints in \( f \) and \( n \) breakpoints in \( g \)
Complexity of the algorithm

- For one-variable minimization problem
  \[
  d_2(A, B) = \min_{t \in [0, 1]} \int_0^1 [f(s + t) - g(s)]^2 ds - [\theta(t)]^2
  \]

- The basic algorithm compute the minimum value in O(mn(m+n)) time
- A refined version of algorithm runs in O(mnlog(mn)) time

Comparing polygons with graph matching

- Divide contour K by straight-lines which are parallel to X-axis
- The divided graph is called the segmentation E(K) of contour K
The areas restricted by lines and contours are called lumps.

Lumps with more than one point in common are called adjacent.

Among adjacent lumps, parent lump is higher in $Y$ direction.

$E(K)$ induces an associated graph $G(K)$, which is called $G$-graph.

---

**G-graph**

- The areas restricted by lines and contours are called lumps.
- Lumps with more than one point in common are called adjacent.
- Among adjacent lumps, parent lump is higher in $Y$ direction.
- $E(K)$ induces an associated graph $G(K)$, which is called $G$-graph.

---

**Fig. 1b**

Graph $G(K)$.  

---

6
G-graph (continued)

- Siblings are in the relation ‘left of’ or ‘right of’
- A weight function \( w(x) \) is defined on vertices
- There are no pair of vertices \( x, y \) such that \( x \) is the unique parent of \( y \) and \( y \) is the unique parent of \( x \)
- 2 G-graph \( G_1 \) and \( G_2 \) are isomorphic are denoted as \( G_1 \cong G_2 \)

Simplified contours

- Choose an arbitrary leaf in \( K' \) and cut it off
- Repeat the cut-off and get a family of simplified contours
Reduction to simplified contour similarity

- Define the depth DEP(K') of a node K' as the sum of weights of the leaves which have been cut off.
- If G(K1) ≡ G(K2), Define the simple distance
  \[ D(K1,K2) = \sum |w(v) - w(\lambda(v))| \quad v \in V(G(K)) \]
- The function DSIMy(K1,K2) measures the similarity between K1 and K2 in Y direction
  \[ DSIMy(K1,K2) = \min F(K1',K2') \]
  \[ F(K1',K2') = c(DEP(K1')+DEP(K2'))+D(K1',K2') \]

![Fig. 3. The distances between contours.](image-url)
Simplified contour similarity (continued)

• Similarity between contours should be independent of the Y direction

\[ DSIM(K_1, K_2) = \int_0^\pi DSIM_{\nu}(K_1 \odot \alpha, K_2 \odot \alpha) d\alpha \]

• Finally, similarity between K1 and K2 is defined as

\[ DSIM(K1, K2) = \min DSIM(K1, K2 \angle \beta) \]

\[ 0 \leq \beta < 2\pi \]

Elementary morphism

• An elementary morphism of G is defined as
• 1. The deletion of an edge \((z, z1)\)
• 2. If, after deletion, the property of G-graphs is violated, merge the pair of vertices\((z1, z2)\)
Lattice of morphisms

- The lattice of morphisms can be obtained by replacing simplified contours $K'$ by $G(K')$
- For $G'$, $G1'$, $G2' \in L(G)$, define $DEP(G')$ and $D(G1', G2')$ analogous to simplified contours
- Reduce this problem to $G$-graph pair resolution problem

$$DSIM_Y(K1, K2) = DSIM_G(G(K1), G(K2))$$

Fig. 2b. Morphism lattice $L(G(K))$. 
Top-down greedy algorithm

- The difficulty in evaluation of DSIM_g(G1,G2) is the exponential dependence of |V(L(G))| on |V(G)|
- The proposed algorithm replaces the lattices by slowest paths in lattices
- The slowest path P(G) in L(G) is defined as
  1. The first node G’ is the root G
  2. Choose leaf z ∈ G’ with minimal weight w(z)
  3. Execute elementary morphism G’/z

Comparing polygons by signature

- The method is based on shape deformation
- Enclose the polygon by some predetermined outer polygonal shape
- Shrink the outer polygon into the given shape
- Measure the deformation path length at sample points on perimeter of outer shape
- Compare the path lengths of two polygons at sample points
Shape comparison algorithm

• Normalize the target polygon so that it fits within a bounding box of pre-defined size
• Triangulate zone between the outer and inner polygons
• Eliminate the triangles according to some criteria (snapping)
• Among triangles which can be eliminated, pick the one with largest area

Snapping rules

• Rule 1. One edge $P_iP_{i+1}$ is on the outer loop and opposite vertex $Q_j$ is on Q, snap a point $K$ on $P_iP_{i+1}$ to $Q_j$
Snapping rules (continued)

- Rule 2. Two edges $P_{i+1}P_{i+2}$, $P_{i+2}P_{i+3}$ are on the outer loop, snap $P_{i+2}$ to a point $L$ on $P_{i+1}P_{i+3}$

Snapping rules (continued)

- Rule 3. Snap operations are to be done such that the resulting outer loop is never intersecting itself
Pathline

- Pathline is the trajectory of a point on outer polygon P to its corresponding position on inner polygon Q
- A pathline begins from or terminates at a vertex of either P or Q is called primary pathline
- Non-primary pathline are derived by interpolating primary pathlines
- All points in zone Z are expressed by $\pi_s(t)$

Postprocessing

- Pathlines can be straightened where possible

(f) Pathline

(g) Refined pathlines
Signature file

• A signature file for a polygon is defined as a vector \( S = (s_1, s_2, \ldots, s_n) \)

\[
s_i = \frac{1}{1 + |B|} \left( \text{PathLength}(a) + \sum_{b \in B} \text{PathLength}(b) \right)
\]

Comparison

Two polygons \( P \) and \( Q \) are compared by computing from their signature files

\[
\text{SimilarityIndex}_{p,q} = \frac{1}{n} \sum_{i=1}^{n} (S_P(i) - S_Q(i))^2
\]
Summary

• Turning function
• Graph matching
• Shape signature by deformation