

Shape Matching  
 ASCI course A20, feb 2003, part 3

Remco Veltkamp

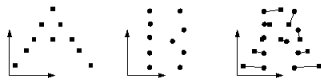


## Geometric Patterns Matching

- point set - point set
- curve - curve
- region - region

## Point Sets

- point set  $A = \{a_1, \dots, a_n\}$ , query/data
- point set  $B = \{b_1, \dots, b_m\}$ , model/target
- one-to-one or many-to-many
- graph theoretical approach:  
 bipartite graph  $(V, E)$ ,  $V = A \cup B$ ,  $E$  weighted



- geometric approach more efficient:  
 "Geometry Helps in Matching" [Vaidya 89]

point sets  
 1-1

## Bottleneck Distance 1

- 1-1 matching of sets  $A, B$  of  $n$  points
- min of max distance  $d(a, b)$  over all 1-1 matchings  $(a, b)$
- using parametric search:  $O(n^{1.5} \log n)$
- approximate matching  $d(A, B) < (1 + \epsilon) d^*(A, B)$   
 using approximate nearest neighbors  $O(n^{1.5} \log n)$

point sets  
 1-1

## Bottleneck Distance 2

- decision problem for translation:  
 decide whether  $t$  exists such that  $d(A+t, B) < \epsilon$   
 $O(n^5 \log n)$
- approximation: compute  $t$  such that  
 $d(A+t, B) < (1 + \epsilon) d(A+t^*, B)$   
 $O(n^{2.5} \log n)$
- optimization for rigid motion:  $O(n^6 \log n)$
- optimization for translations:  $O(n^5 \log n)$

point sets  
 1-1

## Minimum Weight

- minimum total sum of the distances  $d(a, b)$   
 over all 1-1 matches  $(a, b)$
- computation:  $O(n^{2+\epsilon})$   
 but for  $L_\infty$ :  $O(n^2 \log^3 n)$

point sets  
1-1

## Most Uniform

- most uniform/balanced/fair
- minimum difference between maximum and minimum  $d(a,b)$  over all 1-1 matches (a,b)
- $O(n^{10/3} \log n)$

point sets  
1-1

## Minimum Deviation

- minimum difference between maximum and average  $d(a,b)$  over all 1-1 matches (a,b)
- $O(n^{10/3} \log n)$

point sets  
n-m

## Voting Schemes

- alignment [Huttenlocher 87]
- generalized Hough transform, pose clustering [Balard 81, Stockman 87]
- geometric hashing [Lamdan & Wolfson 88]
- query point set  $A = \{a_1, \dots, a_n\}$
- model point set  $B_i = \{b_1, \dots, b_m\}, i=1, \dots, N$

point sets  
n-m

## Alignment

- for each triplet of points in query set  
for each triplet from model set  
compute trafo between them
- with this trafo, transform all other points from model set
- if they match with query points, trafo receives a vote
- if number of votes is above threshold, trafo is accepted
- time complexity for matching:  $O(Nm^4n^3)$

point sets  
n-m

## Pose Clustering

- affine trafo represented by 6 coefficients
- quantize trafo space into 6D table
- for all model point sets  
for each triplet of points in query set  
for each triplet of points model set  
compute trafo between two triplets  
tally a vote in corresponding entry
- highest score gives trafo between query and model
- time complexity for matching:  $O(Nm^3n^3)$

point sets  
n-m

## Geometric Hashing 1

- choose 3 points  $e_0, e_1, e_2$  from  $B_i$
- $b_j = e_0 + \alpha(e_1 - e_0) + \beta(e_2 - e_0)$
- quantize  $(\alpha, \beta)$ -plane into 2D table
- each  $(\alpha, \beta)$  is mapped to an index  $(u, v)$
- for each 3 non-collinear points  $e_0, e_1, e_2$ ,  
for each  $b_i = e_0 + \alpha(e_1 - e_0) + \beta(e_2 - e_0)$ ,  
append  $(i, e_0, e_1, e_2)$  to entry  $(u, v)$
- time complexity for N models:  $O(Nm^4)$

point sets  
n-m

## Geometric Hashing 1.5

point sets  
n-m

## Geometric Hashing 2

- choose 3 points  $e'_0, e'_1, e'_2$  from A
- for each  $a_i = e_0 + \alpha(e'_1 - e'_0) + \beta(e'_2 - e'_0)$ , tally a vote for each  $(i, e_0, e_1, e_2)$  in entry  $(u, v)$
- winner  $(i, e_0, e_1, e_2)$  indicates that point set  $B_i$  contains query set A
- affine trafo between  $(e'_0, e'_1, e'_2)$  and winner  $(e_0, e_1, e_2)$
- time complexity for matching:  $O(n)$

point sets  
n-m

## Geometric Hashing 3

- robust against occlusion of models:  yes
- robust against noise in query:  no
- variations:
  - balance the hashing table
  - avoid taking all  $O(m^3)$  tuples

point sets  
n-m

## Hausdorff Distance 1

- $d(a, B) = \min_{b \in B} d(a, b)$
- directed Hausdorff distance:  
 $h(A, B) = \max_{a \in A} d(a, B)$
- Hausdorff distance:  
 $H(A, B) = \max \{h(A, B), h(B, A)\}$

point sets  
n-m

## Hausdorff Distance 2

- computation in time  $O((n+m)\log(n+m))$  using Voronoi diagrams
- optimization under translation:  
 $O(mn(m+n)\alpha(mn)\log(m+n))$   
with  $\alpha()$  the inverse Ackermann function, a very slowly increasing function
- optimization under rigid motion:  
 $O((m+n)^6 \log(mn))$



point sets  
n-m

## Hausdorff Distance 2

- partial directed Hausdorff distance:  
 $h_k(A, B) = k^{\text{th}}_{a \in A} d(a, B)$
- not a metric

point sets  
n-m

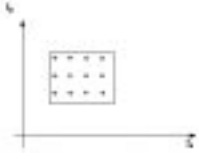
## Hausdorff Distance 3

- application (Huttenlocher)
- model points: 
- data points+match: 

point sets  
n-m

## Template Matching

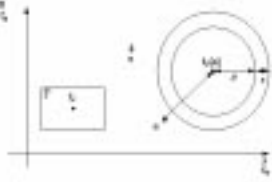
- because of high complexity of Hausdorff distance optimization problem under translation or rigid motions, probe transformation space:



- variations:
  - hierarchical template matching
  - adaptive template matching

point sets  
n-m

## Trafo Space Subdivision 1

- translation in 2D: 
- $\rho(T) = \max_{t \in T} |(t-t_c)|$
- if  $d(t_c(a), B) > \rho(T) + \epsilon$  then no  $t \in T$  with  $d(t(a), B) \leq \epsilon$

point sets  
n-m

## Trafo Space Subdivision 2

- [Huttenlocher et al 93]
- $U =$  translation + scaling:
- $\rho(U) = \max_{u \in U} |(s-s_c)a + (t-t_c)|$
- $a \geq a$  for all  $a \in A$
- if  $d(u_c(a), B) > \rho(U) + \epsilon$  then no  $u \in U$  with  $d(u(a), B) \leq \epsilon$

point sets  
n-m

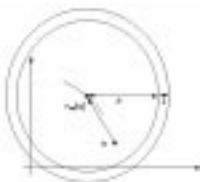
## Trafo Space Subdivision 3

algorithm:

- if  $h_k(u_c(A), B) > \epsilon + \rho(U)$
- then return  $\phi$
- else if  $h_k(u_c(A), B) \leq \epsilon$
- then return  $U$
- else split  $U$  and recur

point sets  
n-m

## Trafo Space Subdivision 4

- rotation: 
- $Rot_{r_1}(a) - Rot_{r_2}(a) \leq |a| |r_1 - r_2|$
- $M =$  rotation + translation
- spherical volume:
- $\rho(M) = \max_{m \in M} |r - r_c| |a| + |(t - t_c)|$
- if  $d(m_c(a), B) > \rho(M) + \epsilon$
- then no  $m \in M$  with  $d(m(a), B) \leq \epsilon$

point sets  
n-m

## Trafo Space Subdivision 5

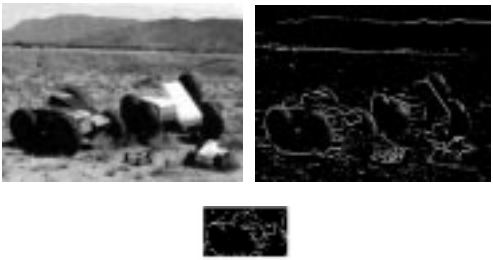
- [Hagedoorn & Velkamp 97], traced volumes:
- $\tau(M,a) \supseteq \{x | x=m(a), m \in M\}$
- if  $\#(a: \tau(M,a) \oplus \text{cube}(\epsilon)) \cap B = \phi > |A| - k$
- then no  $m \in M$  with  $h_k(m(A), B) \leq \epsilon$

Algorithm:

- if  $\#(a: \tau(M,a) \oplus \text{cube}(\epsilon)) \cap B = \phi > |A| - k$
- then return  $\phi$
- else if for all  $a$ ,  $\text{diam}(\tau(M,a)) \leq \epsilon$
- then return  $M$
- else split  $M$

point sets  
n-m

## Trafo Space Subdivision 6



point sets  
n-m

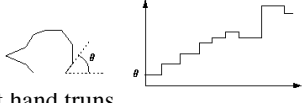
## Trafo Space Subdivision 7

- model: 379 points
- test image: 15,520 points
- transformation: 2D translation + scaling
- $\epsilon$ : 1 pixel width
- $k$ : 360
- spherical volume: 5,786 cells, 9,375 seconds
- traced volume: 783 cells, 950 seconds

curves

## Turning Function 1

- cumulative angle function  $\Theta_A(s)$ : angle between ccw tangent and x-axis as a function of the arc length  $s$
- for polylines: piecewise constant
- increasing with left hand turns
- decreasing with right hand turns
- invariant under translation
- rotation over angle  $\theta$ : vertical shift with  $\theta$



curves

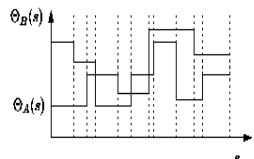
## Turning Function 2

- dissimilarity function:  $L_p$  on functions
- $d_{A,B} = (\int |\Theta_A(s) - \Theta_B(s)|^p ds)^{1/p}$
- rotating by  $\theta$ :  $\Theta_A + \theta$
- invariance for rotation: minimize  $d_{A,B}(\theta) = (\int |\Theta_A(s) - \Theta_B(s) + \theta|^p ds)^{1/p}$  for  $\theta$

curves

## Turning Function 3

- matching: for  $p=2$ ,  $d_{A,B}(\theta)$  minimal for  $\theta = \int \Theta_B(s) ds - \int \Theta_A(s) ds$
- polyline of unequal length: shift one along the other
- variation of [Arkin et al 91]



curves

## Signature Function

- less discriminative than turning function
- at every point: arc length to the left or on the tangent line

- invariant under translation, rotation, scaling
- convex curves: 1 everywhere
- constructed in  $O(n^2)$  time [O'Rourke 85]

curves

## Signature Function

- matching two signature functions: 'time warps', pairing elements of A to elements of B
- using dynamic programming:  $O(nm)$  time

curves

## Fréchet Distance 1

- while walking forward along curves A and B, minimum over all walking speeds of max of distance between corresponding points
- can be larger than Hausdorff distance:

- more formally: min over all monotone increasing  $\alpha(t), \beta(t)$  of max distance  $d(A(\alpha(t)), B(\beta(t)))$

curves

## Fréchet Distance 2

- polylines special case
- deciding  $d_{A,B} < \epsilon$ :  $O(mn)$  time
- computing  $d_{A,B}$ :
  - $O(mn \log(mn))$  time (parametric search)
  - $O(mn (\log(mn))^3)$  (simpler)
  - $O(mn(m+n) \log(mn))$  (still simpler)
- [Alt et al 95], [Godau 91]

curves

## Fréchet Distance 3

- variation: drop monotonicity condition
  - resulting distance  $\delta_{A,B}$  is a semimetric
  - deciding  $\delta_{A,B} < \epsilon$ :  $O(mn)$  time
  - computing  $\delta_{A,B}$ :  $O(mn \log(mn))$  time
- variation: partial matching, shifting one polyline along the other
  - deciding  $\delta_{A,B} < \epsilon$ :  $O(mn \log(mn))$  time
  - computing  $\delta_{A,B}$ :  $O(mn(\log(mn))^2)$  time

curves

## Size Function

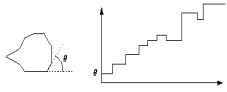
- use measuring function  $f$ , e.g. normalized ordinate of vertex
- size function  $z_f(a,b)$ : number of connected components with  $f \leq b$  that have at least one point with  $f \leq a$ :

- matching: use corner points and multiplicity
- does not uniquely define a shape, but classes

### Turning Function 1

regions

- cumulative angle function  $\Theta_A(s)$ : angle between ccw tangent and x-axis as a function of the arc length  $s$
- for polygons: piecewise constant
- increasing with left hand turns
- decreasing with right hand turns
- invariant under translation
- rotation over angle  $\theta$ : vertical shift with  $\theta$

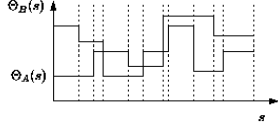


### Turning Function 2

- dissimilarity function:  $L_p$  on functions
- $d_{A,B} = (\int |\Theta_A(s) - \Theta_B(s)|^p ds)^{1/p}$
- rescale polygons to have perimeter length 1
- shifting starting point by  $t$ :  $\Theta_A(s+t)$
- rotating by  $\theta$ :  $\Theta_A + \theta$
- invariance for starting point and rotation: minimize  $d_{A,B}(t,\theta) = (\int |\Theta_A(s+t) - \Theta_B(s) + \theta|^p ds)^{1/p}$  for  $t$  and  $\theta$
- unevenly spread noise is problematic

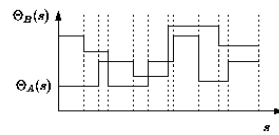
### Turning Function 3

- for fixed  $t$ , and  $p=2$ ,  $d_{A,B}(t,\theta)$  minimal for  $\theta = \int \Theta_B(s) ds - \int \Theta_A(s) ds - 2\pi t$
- for polygons,  $\Theta$  is step function
- $d_{A,B}(t, \theta)$  is sum of  $O(m+n)$  terms:



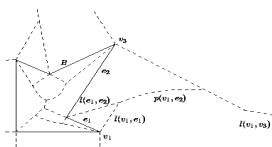
### Turning Function 4

- minimum  $d_{A,B}(t, \theta)$  when two steps coincide:  $O(mn)$  possibilities
- giving  $O(mn(m+n))$  algorithm
- incremental evaluation:  $O(mn \log(mn))$
- [Arkin et al 91]



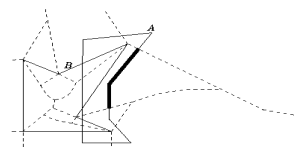
### Hausdorff Distance 1

- computed with Voronoi diagram
- each vertex/edge of polygon B has region of points closer to that vertex/edge than to any other:
- line segments and parabolic segments
- Voronoi diagram of B has  $O(n)$  edges
- computed in time  $O(n \log n)$



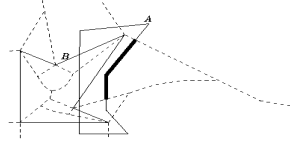
### Hausdorff Distance 2

- directed Hausdorff dist from A to B:  $h(A,B)$
- consider part of A falling in single region of VD(B):
- moving along the thick polyline,  $h(A,B)$  decreases, then increases
- max distance at intersection of thick segments with VD(B)



### Hausdorff Distance 3

- in general:  
max distance  
at vertex of A  
or at intersection  
A with VD(B)
- at those points, compute distance to B and take the maximum
- with sweep line algorithm:  $O((m+n) \log(m+n))$
- same for  $h(B,A)$ , take  $\max\{h(A,B), h(B,A)\}$
- [Alt et al 95]



### Hausdorff Distance 4

- minimal Hausdorff dist under translation:
  - $O((mn)^2(\log(m+n))^3)$  time (parametric search)
  - $O((mn)^3(m+n) \log(m+n))$  time (simpler)
- approximation:
  - $L_A$ : lower left corner of axis parallel bounding box
  - if  $f^*$  gives minimal  $h^*=h(f^*(A),B)$  then  $d(L_A, L_B) \leq h^*\sqrt{2}$
  - so if  $f$  maps  $L_A$  onto  $L_B$  then  $h(f(A),B) \leq (1+\sqrt{2})h^*=h(f^*(A),B)$
  - determining  $f$ :  $O(m+n)$  time
  - computing  $d(f(A),B)$  still  $O(m+n) \log(m+n)$

### Hausdorff Distance 5

- minimal Hausdorff distance under rigid motions:  
 $O((mn)^4(m+n) \log(m+n))$
- approximation:
  - $K_A$ : centroid of edges of convex hull of A
  - suppose  $f^*$  gives minimal  $h^*=h(f^*(A),B)$
  - many rigid motions of A map  $K_A$  onto  $K_B$
  - if  $g$  is the one that gives the smallest Hausdorff distance then  $h(g(A),B) \leq (4\pi+3)h^*=h(f^*(A),B)$
  - determining  $g$ :  $O((mn)\log(mn) \log^*(mn))$  time
  - $O(\log^*n)$  is number of times that log has been applied to get down from  $n$  to below 1, for example  $\log^*(2^{4294967296})$  is only 6

### Area of Overlap

- two simple polygons
- to compute area of overlap:
  - construct arrangement (combinatorial structure of point, edges, and facets of overlay of polygons)
  - time  $O(n \log^*n + C)$ , with  $C$  complexity of arrangement: number of vertices, edges, facets
  - after preprocessing of  $O((mn)^2)$  time
  - overlap computed in time  $O(\log(m+n))$
  - also for minimization under translation
- [Mount & Wu 96]

### Area of Overlap

- convex polygons:
- optimization under translations:  
 $O((m+n) \log(m+n))$  time [de Berg et al 97]
- making centroids coincide gives overlap of at least 9/25 of the optimal

### Area of Symmetric Difference

- making centroids coincide gives symmetric difference of at most 11/3 of the optimal under translations [Alt et al 96]
- also for trasfos  $F$  other than translations:
  - if the centroid of A,  $c(A)$ , is equivariant under transformations  $f$ :  $c(f(A))=f(c(A))$
  - and if  $F$  closed under composition with translation

## Concluding remarks

$d(f(A),B)$ , research issues:

- partial matching:  $d$
- matching two *sets* of curves, regions:  $A, B$
- matching articulated figures :  $A, B$
- matching under larger class of trafo:  $f$