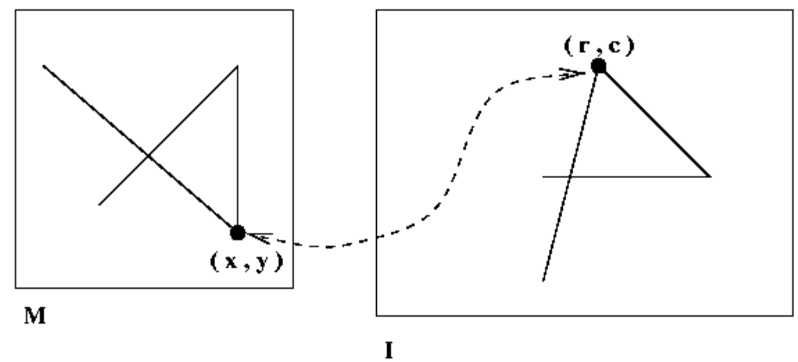
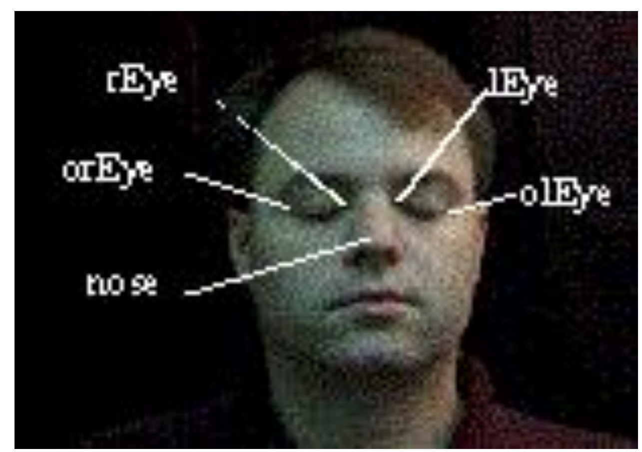
Point/feature matching methods 2

George Stockman
Computer Science and Engineering
Michigan State University

Problems (review)

- How to find landmarks to match across two images?
- How to distinguish one landmark from another?
- How to match N landmarks from I1 to M landmarks from I2, assuming that I2 has more and less than I1 ?

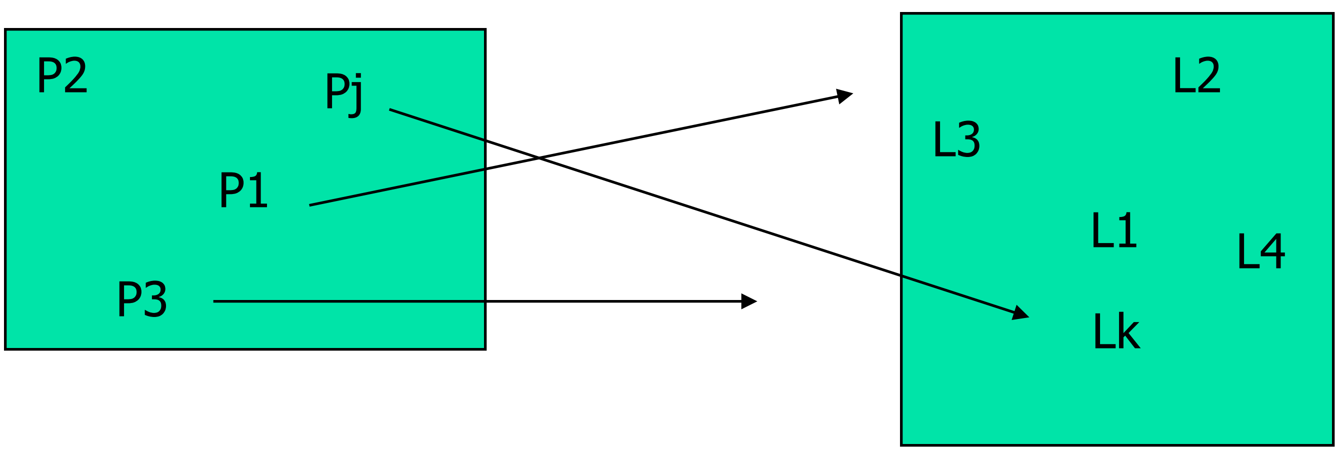




General matching notions

- Pj are points (or “parts”) in first image
- Lk are labels in second image or model; these could be points, parts, or interpretations of them
- Need to match some pairs (Pj , Lk): can be combinatorically expensive

There are many mappings from Image 1 pts to Image 2 labels.

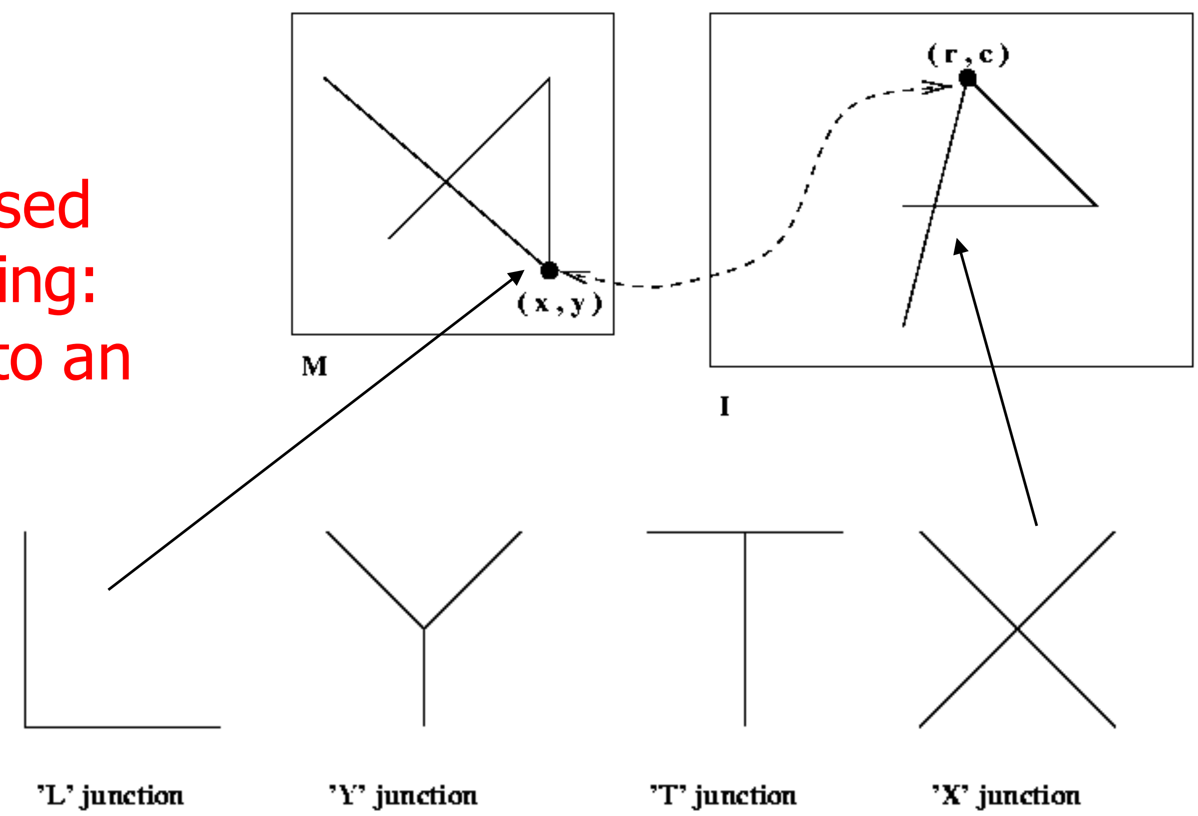


Constraining the mapping

- Points may have distinguishing features
- Points may be distinguished by relations with neighbor points or regions
- Points may be distinguished by distance relations with distant points
- Points or corners might be connected to others

Point salience by topology/geometry

Junctions of a network can be used for reliable matching: never map an ‘L’ to an ‘X’, for example.



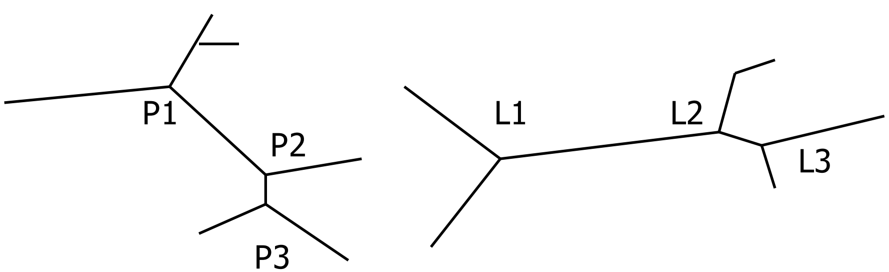
Might also use subtended angles or gradients across edges

Other constraints on mapping

- Match centroids of similar regions, e.g. of dark regions in IR of similar area
- Match holes of same diameter
- Match corners of same angle and same contrast (gradient)
- Match points with similar RMTs

Can match minimal spanning trees (C. Zahn 1975)

- Extract minimal spanning trees from points of I1 and of I2
- Assumption is that spanning tree structure is robust to some errors in point extraction
- Select tree nodes with high degree
- Below example: match 3 nodes of same degree and verify a consistent RS&T mapping



Local focus feature (Bolles)

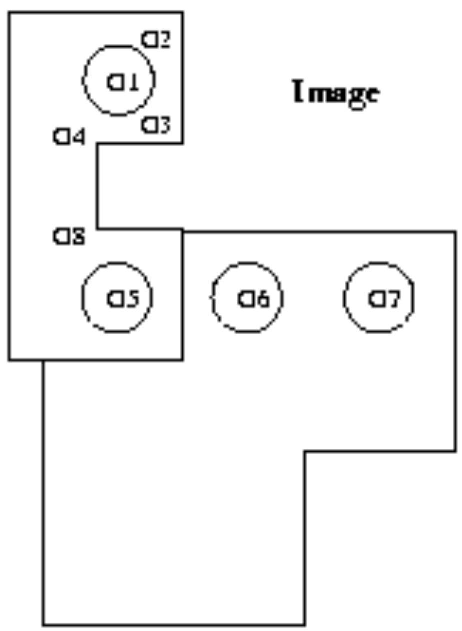
- Identify several “focus features” that are close together
- Match only the focus features
- Can a consistent RS&T be derived from them?
- If so, can find more distant features to refine the RS&T
- Method robust against occlusions/errors

Local focus feature matching

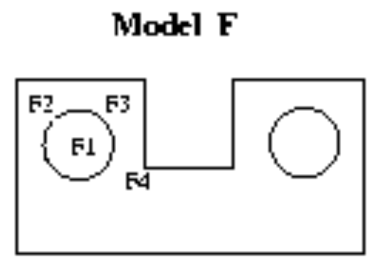
Let $G_i, i = 1, I$ be the detected image features.
Let $F_m, m = 1, M$ be the focus features of the model.
Let $S(f)$ be the set of nearby features for any feature f.

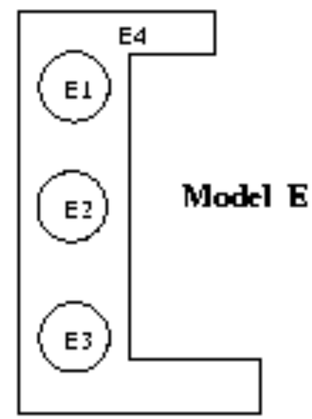
procedure local_feature_focus(G,F);
{
for each focus feature F_m
  for each image feature G_i of the same type as F_m
    {
      Find the maximal subgraph S_m of $S(F_m)$ that
        matches a subgraph S_i of $S(G_i)$;
      Compute the transformation T that maps the points of
        each feature of S_m to the corresponding feature of S_i;
      Apply T to the boundary segments of the model;
      if enough of the transformed boundary segments find
        evidence in the image **then return**(T);
    };
}

Try matching model to image



Model feature F1 matches image hole a5. When rotated 90 degrees, two edges will align, but global match still wrong





Model E will match image fairly well also.

Using distance constraints

Correct labeling is

{ (H1, E), (H2, A), (H3,B) }

(Left) Model object and (right) three holes detected in an image.
Model Point Locations and Interpoint Distances

point	coordinates †	to A	to B	to C	to D	to E
A	(8,17)	0	12	15	37	21
B	(16,26)	12	0	12	30	26
C	(23,16)	15	12	0	22	15
D	(45,20)	37	30	22	0	30
E	(22,1)	21	26	15	30	0

$12(P_1, P_2)$ **means that** $d(P_1, P_2) = 12$

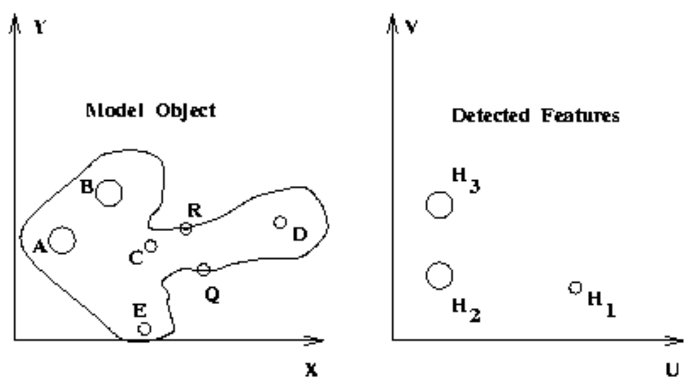


Image Point Locations and Interpoint Distances

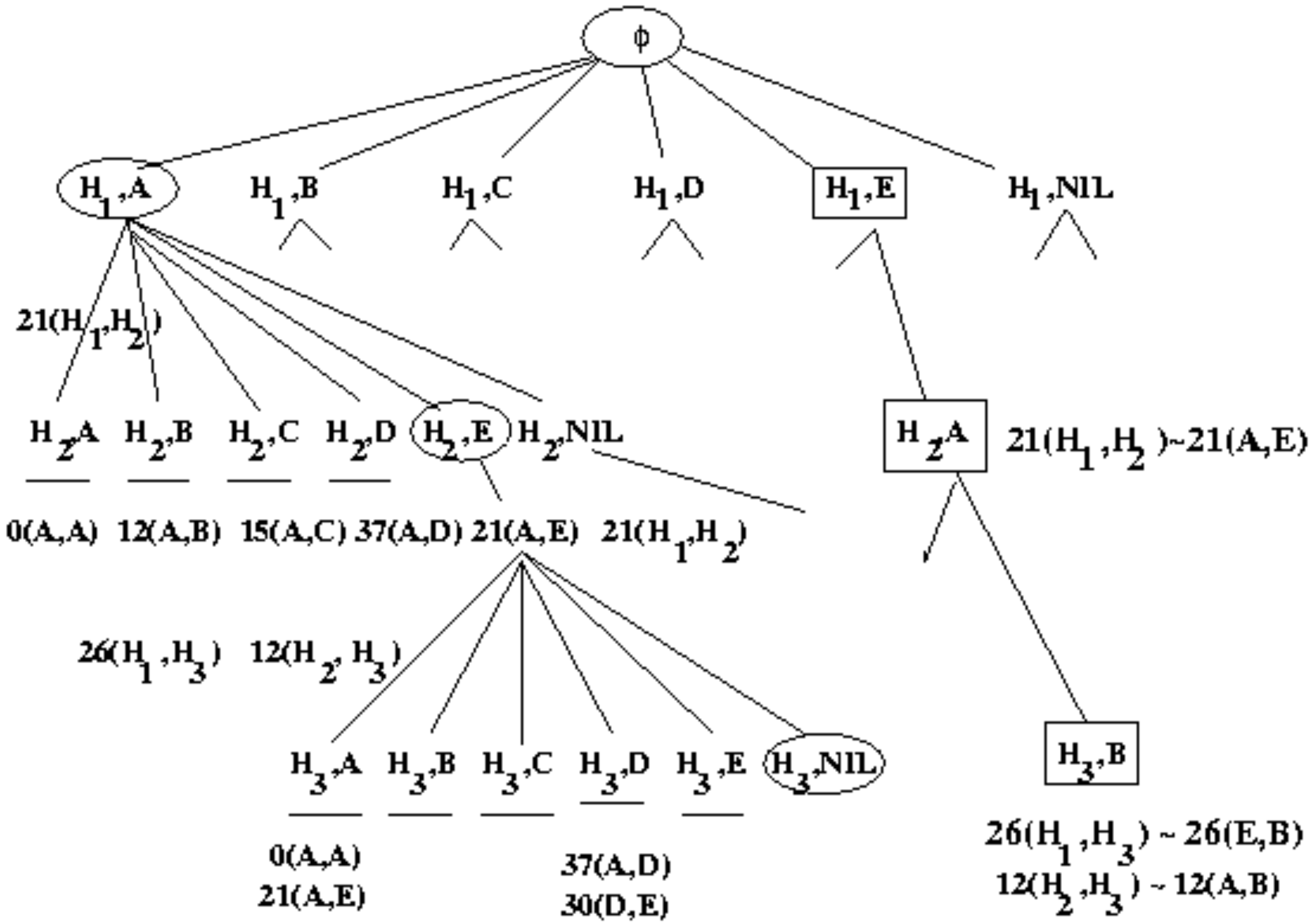
point	coordinates †	to H1	to H2	to H3
H1	(31,9)	0	21	26
H2	(10,12)	21	0	12
H3	(10,24)	26	12	0

Distances observed in the image must be explained in the model.

Backtracking search for consistent labeling of {H1, H2, H3}

If H1 is A, then H2 must be E to explain distance 21. Then, there is no label to explain H3.

If H1 is E, then H2 and H3 can both be consistently labeled.



Let’s call features of image 1 “parts” and features of image 2 “labels”

1 Definition *Given a set of parts P, a set of labels for those parts L, a relation R_P over P, and a second relation R_L over L, a consistent labeling f is an assignment of labels to parts that satisfies:*

$$\text{If } (p_i, p_{i'}) \in R_P, \text{ then } (f(p_i), f(p_{i'})) \in R_L.$$

Labels of related parts should be related

I1: S1 above S4

I2: Sj above Sn

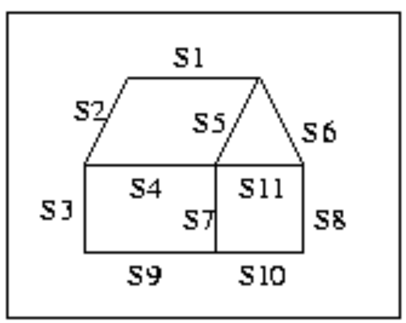


Image 1

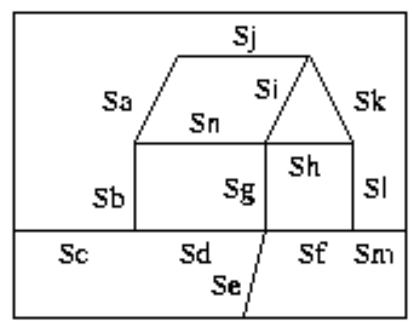
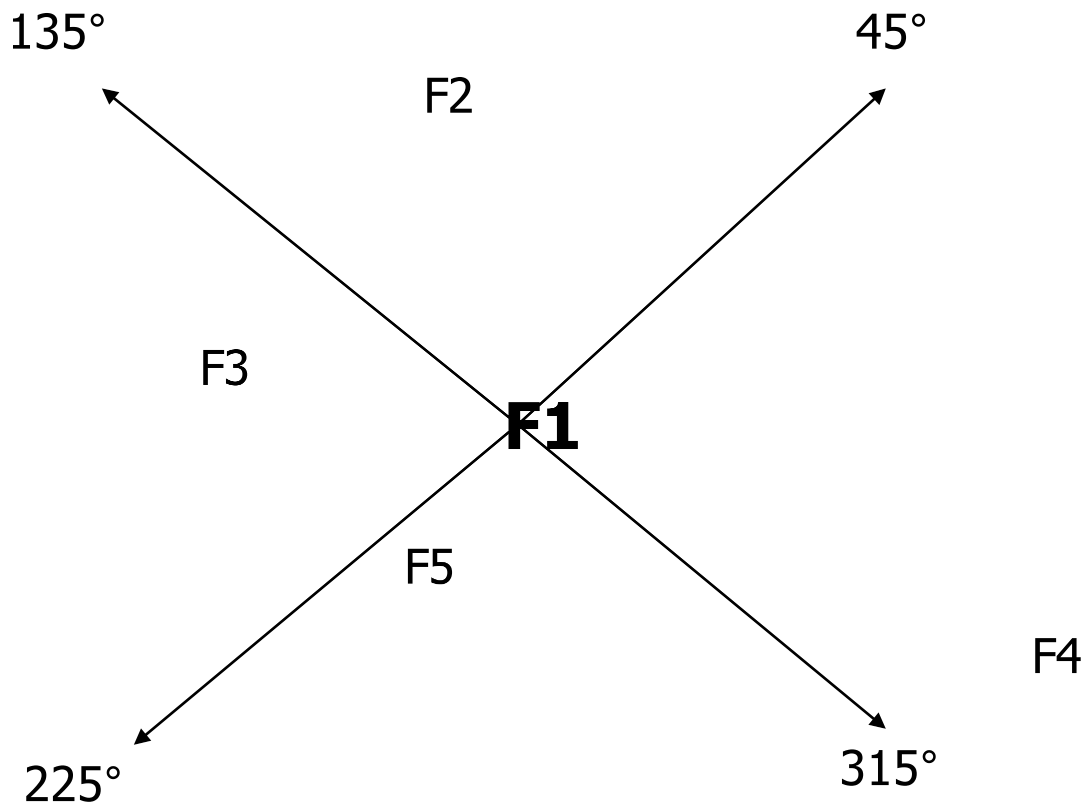


Image 2

What relations?

- We have been using rigid geometry – distance and angle
- Topological: F1 connects to F2; F1 inside of F2; F1 crosses F2; …
- Other relations are useful when rigid mapping does not exist: F1 left of F2; F1 above F2; F1 between F2

Simple definitions of …



F2 above F1

F1 below F2

F5 below F1

F1 above F5

F1 left_of F4

F1 right_of F3

F4 right_of F1

F3 left_of F1

IT backtracks over part-label pairs to get consistent set

- A stack holds all pairings {(P1, L1), … , (Pj, Lk)}
- check all relations on Pj and Lk
- if relation is violated, retract pairing (Pj, Lk) and try a new one

In general, IT Search is exponential in effort; however, in 2D or 3D alignment, Grimson and Lozano Perez have shown the bound to be $O(N^3)$

procedure Interpretation_Tree_Search(P, L, R_P, R_L, f);
{
p := first(P);
for each l in L
{
$f' = f \cup \{(p, l)\}$; /* add part-label to interpretation */
OK = true;
for each N-tuple $(p_1, \ldots, p_N)$ in R_P containing component p
and whose other components are all in domain(f)
/* check on relations */
if $(f(p_1), \ldots, f(p_N))$ is not in R_L **then**
{
OK: = false;
break;
}
if OK **then**
{
$P' = $ rest(P);
if isempty(P') **then** output(f');
else Interpretation_Tree_Search(P', L, R_P, R_L, f');
}
}
}

Discrete relaxation deletes part labels that are inconsistent with observations

(Left) Model object and (right) three holes detected in an image.

Model Point Locations and Interpoint Distances

Kleep model distances

point	coordinates †	to A	to B	to C	to D	to E
A	(8,17)	0	12	15	37	21
B	(16,26)	12	0	12	30	26
C	(23,16)	15	12	0	22	15
D	(45,20)	37	30	22	0	30
E	(22,1)	21	26	15	30	0

Image Point Locations and Interpoint Distances

Features and distances observed in the image

point	coordinates †	to H1	to H2	to H3
H1	(31,9)	0	21	26
H2	(10,12)	21	0	12
H3	(10,24)	26	12	0

Label A, B, C cannot match H1 since observed distances 26, 21 have not been observed. Also, H2 cannot be B.

Label Sets Midway through first pass of relaxation labeling.

	A	B	C	D	E
H_1	no $N \ni$ $d(A,N) = 26$	no $N \ni$ $d(B,N) = 21$	no $N \ni$ $d(C,N) = 26$	possible	possible
H_2	$21(H_2, H_1)$ $E \in L(H_1)$ $12(H_2, H_3)$ $B \in L(H_3)$	no $N \ni$ $d(B,N) = 21$	$21(H_2, H_1)$ $D \in L(H_1)$ $12(H_2, H_3)$ $B \in L(H_3)$		
H_1	no $N \ni$ $d(A,N) = 26$	$12(H_3, H_2)$ $A \in L(H_2)$ $26(H_3, H_1)$ $E \in L(H_1)$			

Impossible labels cause additional "filtering" out of possible labels

After completion of the second pass of relaxation labeling

Failed relationships cause labels to be dropped

	A	B	C	D	E
H_1	no	no	no	no	possible
H_2	possible	no	possible	no	no
H_3	no	possible	no	no	no

After completion of the third pass of relaxation labeling

During any pass filtering can be done in parallel; creating a separate output matrix for the next pass

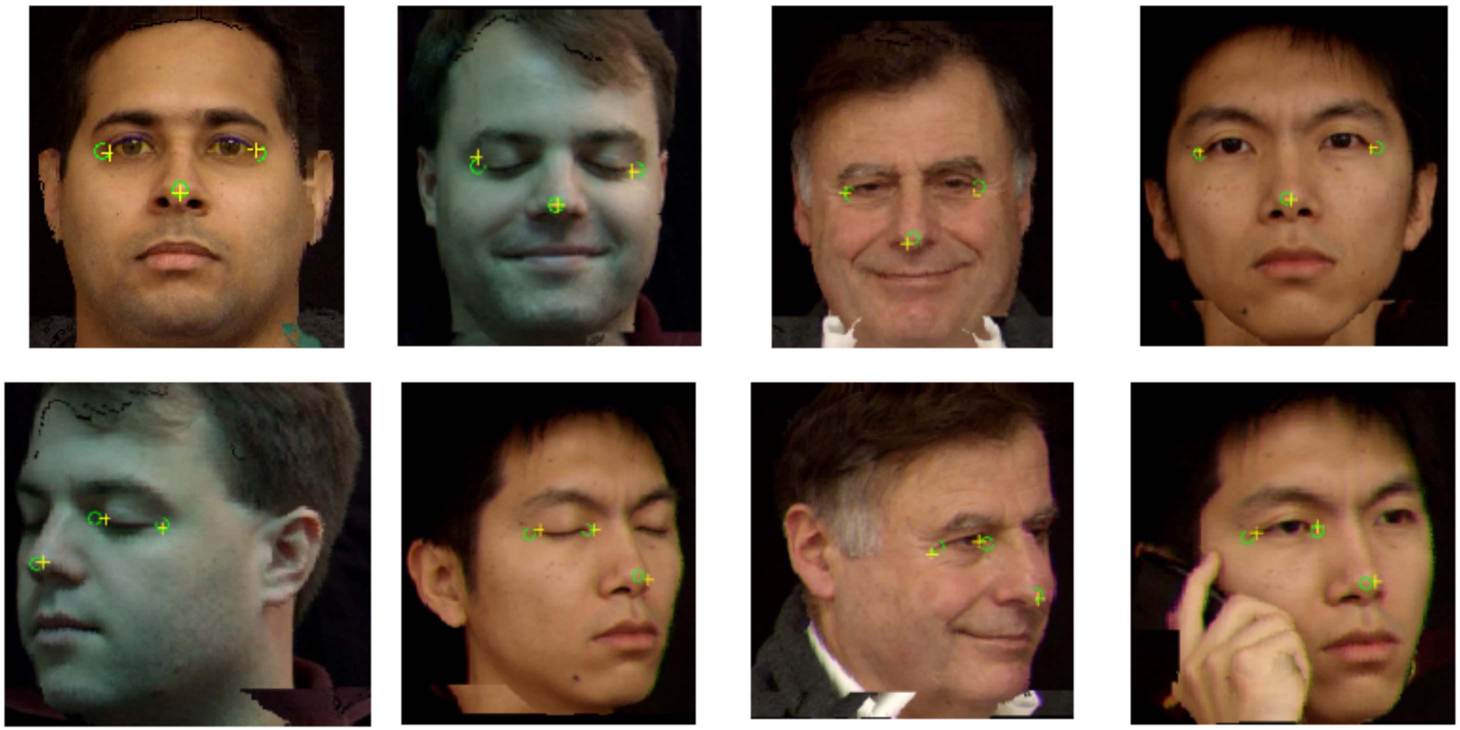
	A	B	C	D	E
H_1	no	no	no	no	possible
H_2	possible	no	no	no	no
H_3	no	possible	no	no	no

Discrete relaxation deletes inconsistent labels in stages

Let $P_i, i = 1, D$ be the detected image features.
Let $S(P_i), i = 1, D$ be the set of initially compatible labels.

```
procedure Relaxation_Labeling(P, S );
{
repeat
   for each (P_i, S(P_i)) do in parallel
   {
      for each label L_k ∈ S(P_i)
         for each relation R(P_i, P_j) in the image parts
            if ∃ L_m ∈ S(P_j) with R(L_k, L_m) in model
               then keep L_k in S(P_i)
               else delete L_k from S(P_i)
   }
until no change in any set S(P_i)
return(S);
}
```

Face points filtered by known L-R, up-down, and distance relationships



Points identified by surface curvature in neighborhood and filtered by location relative to other salient points. These 3 points are then used for iterative 3D alignment (ICP algorithm) of the scan to a 3D model face.

Face relationship filtering

TABLE 3
Relaxation Matrix after First Relaxation

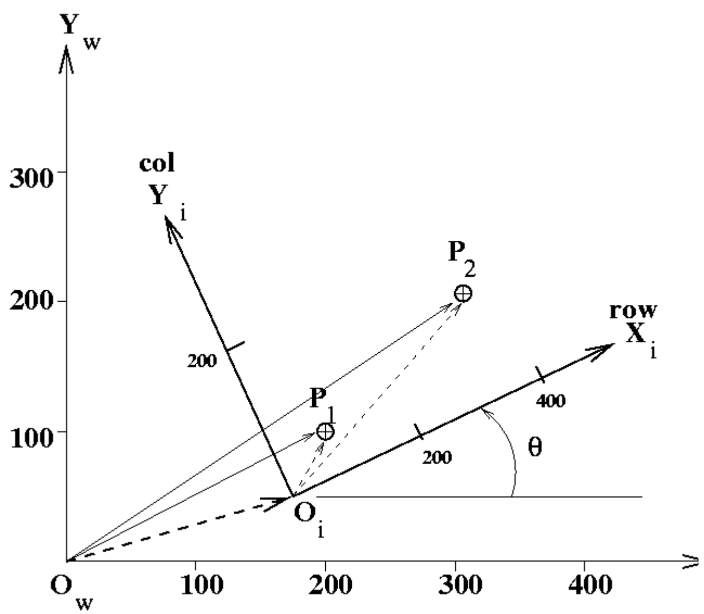
		Interpretation						
		nose	rEye	lEye	orEye	olEye	centroid	null
Observation	Pt 1	1	0	0	1	0	0	1
	Pt 2	0	0	0	0	0	0	1
	Pt 3	0	1	1	1	0	0	1

Notes on deriving a rigid mapping of Pj to Pk

- For 2D to 2D, match 2 points
- For 3D to 3D, match 3 points
- compute transform, then refine it to a best fit using least squares and many more matching points

Match of 2 points in 2D can determine R, S, and T

Rotation, Scaling and Translation



Assume two corresponding control points ^iP_j and ^wP_j

1. determine rotation θ
   (a) direction of the vector $\mathbf{P_1P_2}$ in $\mathbf{I}$ is determined as $\theta_i = \arctan((^iy_2 - {}^iy_1)/(^ix_2 - {}^ix_1))$
   (b) direction of the vector in $\mathbf{W}$ is determined as $\theta_w = \arctan((^wy_2 - {}^wy_1)/(^wx_2 - {}^wx_1))$.
   (c) $\theta = \theta_w - \theta_i$.
2. θ is determined: all *sin* and *cos* elements are known
3. solve 3 equations and 3 unknowns for s and x_0, y_0.

$$\begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

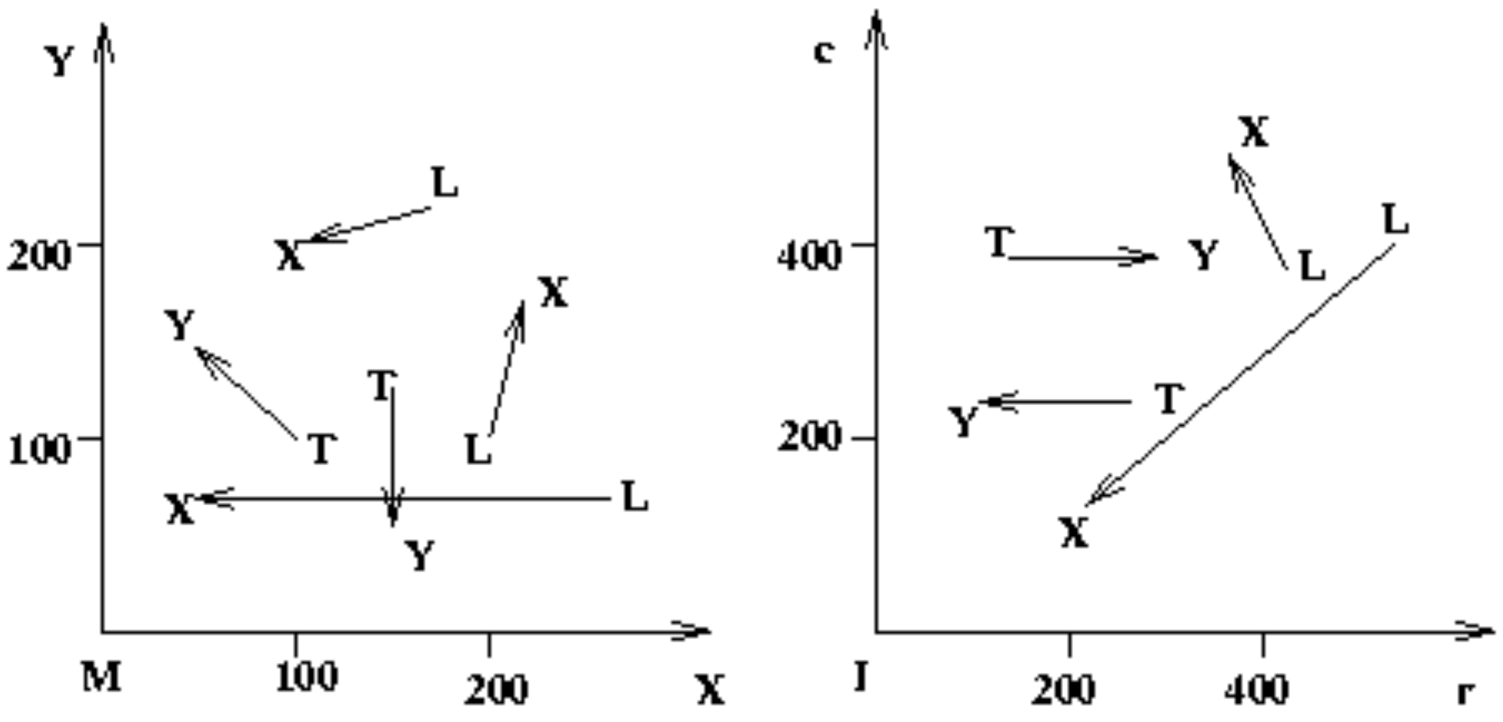
$$x_w = x_i s\cos\theta - y_i s\sin\theta + x_0$$

$$y_w = x_i s\sin\theta + y_i s\cos\theta + y_0$$

Transform derived from 2 points can fit badly for other points

- Better if original 2 points are far apart
- An error of $\Delta\theta$ in the rotation implies an error in point location of $r\Delta\theta$ where r is the distance to the point from the center of rotation.
- Can use crude transformation to pair points and then refit the transformation using least squares

Synthetic example of map to image correspondence



Assuming point labels must match, derive the RS&T matching each pair of similar vectors → 10 possible mappings.

Clusters in pose space show correct transform & error!

Cluster space formed from 10 pose computations.

Model Pair	Image Pair	θ	s	u_0	v_0	
L(170,220),X(100,200)	L(545,400),X(200,120)	0.403	6.10	118	-1240	
L(170,220),X(100,200)	L(420,370),X(360,500)	5.14	2.05	-97	514	
T(100,100),Y(40,150)	T(260,240),Y(100,245)	0.663	2.05	225	-48	*
T(100,100),Y(40,150)	T(140,380),Y(300,380)	3.87	2.05	166	669	
L(200,100),X(220,170)	L(545,400),X(200,120)	2.53	6.10	1895	200	
L(200,100),X(220,170)	L(420,370),X(360,500)	0.711	1.97	250	-36	*
L(260, 70),X(40, 70)	L(545,400),X(200,120)	0.682	2.02	226	-41	*
L(260, 70),X(40, 70)	L(420,370),X(360,500)	5.14	0.651	308	505	
T(150,125),Y(150, 50)	T(260,240),Y(100,245)	4.68	2.13	3	568	
T(150,125),Y(150, 50)	T(140,380),Y(300,380)	1.57	2.13	407	60	

3 of the 10 vector pairings produce approximately the same RS&T transformation: examine the variance of parameters.

Pose-clustering concept

- Match minimal set of (points) features Fs between I and M
- Compute alignment transformation ω from Fs and contribute to cluster space
- Repeat this for "all' or "many" corresponding feature sets
- Identify the best cluster centers ωi and attempt to verify those on other features
- * minimal basis means that ωi will have error; cluster center is better, but iterative refinement of ωi using many more features is better yet

Pose clustering in 2D and 3D

- 2D: Can get 4 parameter RS&T ω from only two matching points (as above)
- 3D: can get 6 parameter rigid trans. from 3 matching points (S&S text 13.2.6)
- 3D: or 2 lines and one point, or 3 lines

RANSAC: random sample and consensus (Fischler and Bolles)

- Randomly choose minimal set of matching points {Pj} and {Lj}
- Compute aligning transformation T from these points so that T(Pj) = Lj for all j
- Check that this transformation maps other points correctly: T(Pi) = Li for more i   &
- Repeat until some transformation is verified (or no more choices remain)

& best to refine T on all correspondences as they are checked.

Corner features matched between aerial images

Registering Image ${}^{1}\mathbf{I}_{\mathbf{t1}}$ to ${}^{2}\mathbf{I}_{\mathbf{t2}}$

General affine transform

ANY COMBINATION of *scaling, rotation, translation, shearing, reflection* is modeled by matrix multiplication—with the combined form:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad (8)$$

The transformation is rigid when the 2 x 2 matrix has orthonormal rows and columns.

$$\begin{matrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{matrix}$$

Derivation of least squares constraints on coefficients ajk

Taking the six partial derivatives of the error function with respect to each of the six variables and setting this expression to zero gives us the six equations represented in matrix form.

$$\begin{bmatrix} \Sigma x_j{}^2 & \Sigma x_j y_j & \Sigma x_j & 0 & 0 & 0 \\ \Sigma x_j y_j & \Sigma y_j^2 & \Sigma y_j & 0 & 0 & 0 \\ \Sigma x_j & \Sigma y_j & \Sigma 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Sigma x_j{}^2 & \Sigma x_j y_j & \Sigma x_j \\ 0 & 0 & 0 & \Sigma x_j y_j & \Sigma y_j^2 & \Sigma y a_j \\ 0 & 0 & 0 & \Sigma x_j & \Sigma y_j & \Sigma 1 \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \end{bmatrix} = \begin{bmatrix} \Sigma u_j x_j \\ \Sigma u_j y_j \\ \Sigma u_j \\ \Sigma v_j x_j \\ \Sigma v_j y_j \\ \Sigma v_j \end{bmatrix} \qquad (16)$$

Best affine transform matching the town images

11 matching point pairs

Matching control point pairs are:

288	210	31	160	232	288	95	205	195	372	161	229	269	314	112	159
203	424	199	209	230	336	130	196	284	401	180	124	327	428	198	69
284	299	100	146	337	231	45	101	369	223	38	64				

The Transformation Matrix is:

Ajk from the least squares procedure

$$\begin{bmatrix} -0.0414 , & 0.773 , & -119 \\ -1.120 , & -0.213 , & 526 \\ 0.0 , & 0.0 , & 1.0 \end{bmatrix}$$

Residuals (in pixels) for 22 equations are as follows:

Δx	0.18	-0.68	-1.22	0.47	-0.77	0.06	0.34	-0.51	1.09	0.04	0.96
Δy	1.51	-1.04	-0.81	0.05	0.27	0.13	-1.12	0.39	-1.04	-0.12	1.81

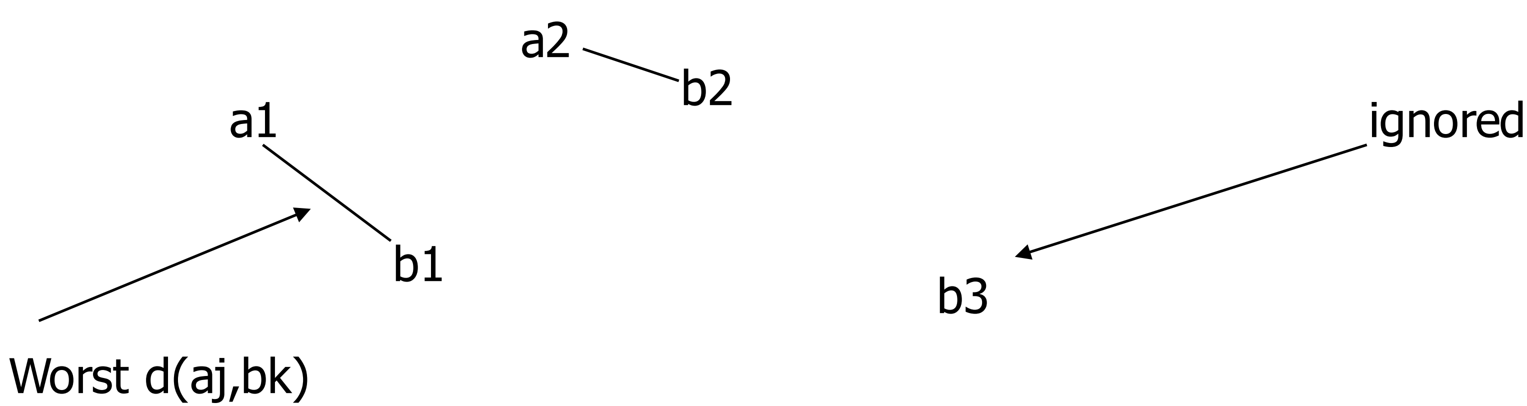
P1

Residuals of mapping are all less than 2 pixels in the right image space.

P11

Euclidean distance between point sets measures their match: 2D or 3D

- set A has points from image 1
- set B has points from image 2
- assume every point from A should be observed in B also
- match measure can be the worst distance from any point in A to some point in B, or the root mean square of all such distances

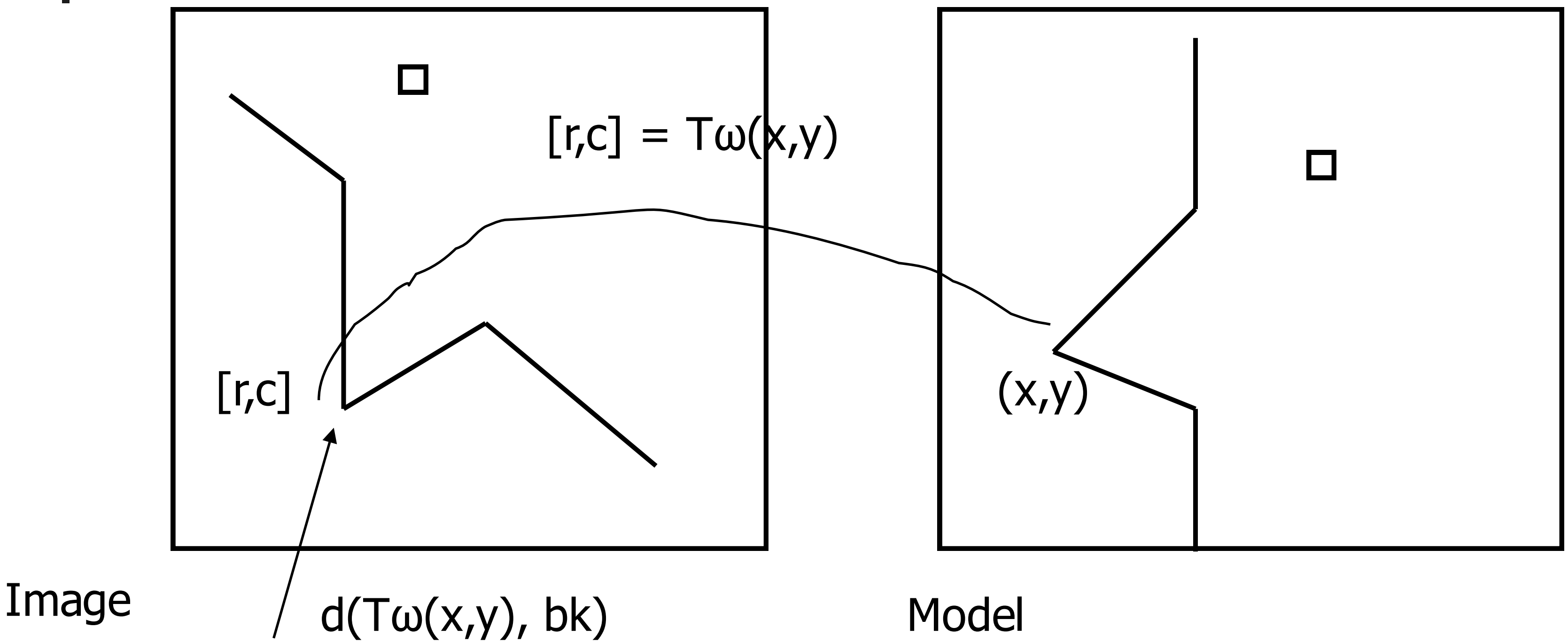


Example application

- Observed data A is 3D surface scan of face
- Model data B is 3D surface model of face
- Best match of A to B probably requires points of A to be rotated and translated at least slightly
- So, we have to search over a 6-parameter space ω
- For each parameter set, we need to transform points of A into the space of B and **then** find the best point matches.

$$compute \quad d(T_\omega(a_j), b_k)$$

Example 2D case



To evaluate the value of the match of the Model in the Image for the pose parameters ω model points must be transformed and best matching image points located – how to do it?

Distance transform helps

- Identify image feature points bk;
- Set image feature points to 0
- Set all neighbors of 0 values to 1
- Set all unset neighbors of 1 values to 2
- Set all unset neighbors of 2 to 3, etc.

$$match_{\omega}(A) = \sum_{(x,y)\varepsilon A} I_B[T(x,y)]$$

Sum all the distance penalties over all transformed points of the model: called "chamfer-matching" by Barrow and Tenenbaum

Example distance transform

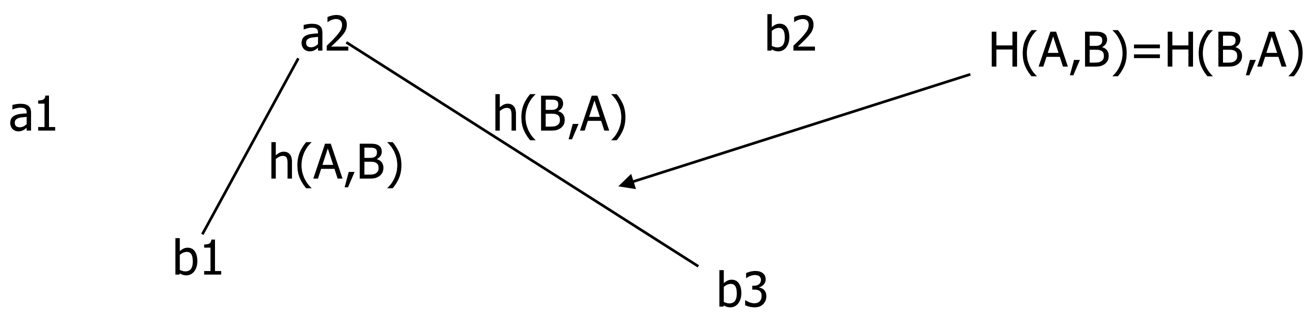
5	4	3	4	5	6	5	4	3	2	3	4	5
4	3	2	3	4	5	4	3	2	1	2	3	4
3	2	1	2	3	4	3	2	1	0	1	2	3
2	1	0	1	2	3	4	3	2	1	2	3	4
3	2	1	0	1	2	3	4	3	2	3	4	5
4	3	2	1	0	1	2	3	4	3	4	5	6
4	3	2	1	0	1	2	2	3	3	4	5	6
4	3	2	1	0	1	1	1	2	2	3	4	5
4	3	2	1	0	0	0	0	1	1	2	3	4
5	4	3	2	1	1	1	1	0	0	1	2	3
6	5	4	3	2	2	2	2	1	1	0	1	2
7	6	5	4	3	3	3	3	2	2	1	2	3

Set image feature points to 0
Set all neighbors of 0 values to 1
Set all unset neighbors of 1 values to 2
Set all unset neighbors of 2 to 3, etc.

Parallel computation: at each stage, every unassigned pixel P checks its neighbors; if any neighbor has a distance label of d, then pixel P becomes d+1

Manhatten distance used here. Can use scaled Euclidean distance, where 4-neighbors are distance 10 and diagonal neighbors are distance 14.

Hausdorff distance: take worst of the closest matches



$$h(A,B) = \max_{a \varepsilon A} \{ \min_{b \varepsilon B} \{ d(a,b) \} \}$$

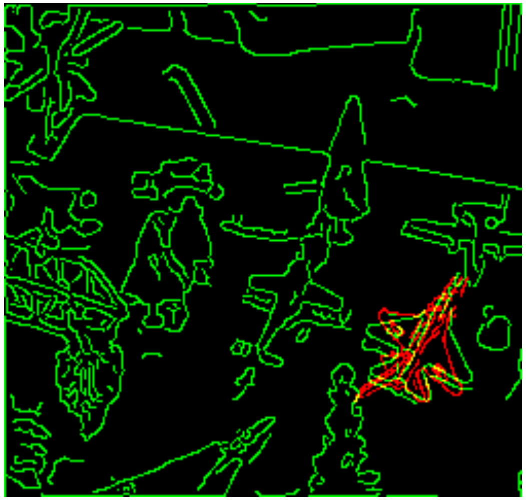
$$in\ general \quad h(A,B) \neq h(B,A)$$

$$a\ symmetric\ dist.\ is$$

$$H(A,B) = \max \{ h(A,B), h(B,A) \}$$

Example use from Leventon

Edge points of
model plane image



Left: Model scene
with objects

Images from Leventon web pages.

Best h(plane,scene)
over all sets of
translated plane edge
points

Symmetric distance not wanted here since
most scene points should not be matched.

Variations on Hausdorff dist.

- Create histogram of all individual point distances of h(A,B)
- If, say 80%, of these are suitably small, then accept the match.
- Can define the 80% Hausdorff distance as that distance D such that d(aj,B) <= D for 80% of the points aj of A
- Perhaps we can now match the outline of a given pickup truck with and without a refrigerator in the back.
- See papers of Huttenlocher, Leventon

Summary of matching, Part 2

- Brute force matching of points is computationally expensive
- Strong constraints on matching from feature point type and relations with other features
- Some matching methods: focus features, RANSAC, relaxation, interpretation tree, pose clustering
- Best affine (RT&T) transformation from N pairs of matching points can be done in 2D or 3D.
- Chamfer-matching enables faster match evaluation

references

- H.G. Barrow, J.M. Tenenbaum, R.C. Bolles, and H.C. Wolf. *Parametric correspondance and chamfer matching: two techniques for image matching*. In Proc. 5th International Joint Conference on Artificial Intelligence, pages 659-663, 1977.
- R. Bartak. *Theory and Practice of Constraint Propagation*, Proc. of the 3rd Workshop on Constraint Programming in Decision and Control, pp. 7-14, Gliwice, June 2001.
- G. Borgefors. *Distance transformations in digital images.* Computer Vision, Graphics, and Image Processing, 34:344-371, 1986.
- H. Breu, J. Gil, D. Kirkatrick, and M. Werman. *Linear time euclidean distance transform algorithm.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 17(5):529-533, 1995.
- H. Eggers. *Two fast euclidean distance transformations in z2 based on sufficient propagation*. *Computer Vision and Image Understanding*, 69(1):106-116, 1998.
- D. P Huttenlocher, G. A. Klanderman, W. J. Rucklidge (1993). *Comparing images using the Hausdorff distance*. IEEE Trans. on Pattern Analysis and Machine Intelligence, 15(9), pp. 850-863.

references

- M. Fischler and R. Bolles, (1981) *Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartrography*, Communications Assoc. for Computing Machinery, (June 1981),24(6):381-395.
- R. Haralick and L. Shapiro (1979) *The consistent labeling problem I*, IEEE-PAMI (1979)173-184.
- R. Hummel and S. Zucker, *On the foundations of relaxation labeling processes*, IEEE Trans. PAMI, vol. 5, pp. 267-287, 1983.
- X. Lu, D. Colbry and A. K. Jain. *Matching 2.5D Scans for Face Recognition*, Proc. International Conference on Biometric Authentication, 2004.
- A. Rosenfeld, R. Hummel, and S. Zucker (1976) *Scene labeling by relaxation operators*, IEEE-SMC (1976)420-453.
- L. Shapiro and G. Stockman (2001) Computer Vision, Prentice-Hall.
- G. Stockman (1987) *Object recognition and localization via pose clustering*, Computer Vision, Graphics and Image Proc. (1987)361-387.
- W. J. Rucklidge (1997). *Efficiently locating objects using the Hausdorff distance*. Int. J. of Computer Vision, 24(3), pp. 251-270.
- Z. Zhang, *Iterative point matching for registration of free-form curves and surfaces*, International Journal of Computer Vision, vol. 13, no. 1, pp. 119-152, 1994.