Extensions

Modifications to the clique-finding algorithm extend it to finding maximal cliques and finding largest cliques. To find largest cliques, perform an additional test to stop the recursion in *Cliques* if the size of X plus the number of nodes in $Y-X$ connected to all of X becomes less than k, which is initially set to the size of the largest possible clique. If no cliques of size k are found, decrement k and run *Cliques* with the new k.

To find maximal cliques, at each stage of *Cliques*, compute the set

$$Y' = \{z \in \text{Nodes: } z \text{ is connected to each node of Y}\}.$$

Since any maximal clique must include Y', searching a branch may be terminated should Y' not be contained in Y, since Y can then contain no maximal cliques.

The association graph may be searched not for cliques, but for r-connected components. An *r-connected* component is a set of nodes such that each node is connected to at least r other nodes of the set. A clique of size n is an $n-1$-connected component. Fig. 11.9 shows some examples.

The r-connected components generalize the notion of clique. An r-connected component of N nodes in the association graph indicates a match of N pairs of nodes from the input and reference structures, as does an N-clique. Each matching pair has similar properties, and each pair is compatible with at least r other matches in the component.

Whether or not the r-connected component definition of a match between two structures is useful depends on the semantics of "compatibility." For instance, if all relations are either compulsory or prohibited, clearly a clique is called for. If the relations merely give some degree of mutual support, perhaps an r-connected component is the better definition of a match.

11.4 MATCHING IN PRACTICE

This section illustrates some principles of matching with examples from the computer vision literature.

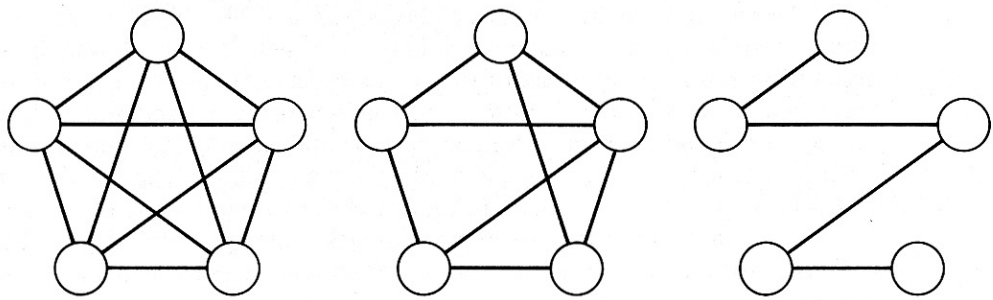


Fig. 11.9 r-connected components. (a) A 5-clique (which is 4-connected). (b) A 3-connected set of 5 nodes. (c) A 1-connected set of 5 nodes.

11.4.1 Decision Trees

Hierarchical decision-tree matching with ad hoc metrics is a popular way to identify input data structures as instances of reference models and thus classify the input instances [Nevatia 1974; Ambler et al. 1975; Winston 1975]. Decision trees are indicated when it is predictable that certain features are more reliably extracted than others and that certain relations are either easier to sense or more necessary to the success of a match.

Winston and Nevatia both compare matches with a "weighted sums of difference" metric that reflects cumulative differences between the parameters of corresponding elements and relations in the reference and input data. In addition, Nevatia does parameter fitting; his reference information includes geometrical information.

Matching Structural Descriptions

Winston is interested in matching such structures as appear in Fig. 11.10B. The idea is to recognize instances of structural concepts such as "arch" or "house," which are relational structures of primitive blocks (Fig.11.10A) [Winston 1975]. An important part of the program learns the concept in the first place—only the matching aspect of the program is discussed here. His system has the pleasant property of representational uniqueness: reference and input data structures that are identical up to the resolution of the descriptors used by the program have identical representations. Matching is easy in this case. Reflections of block structures can be recognized because the information available about relations (such as LEFT-OF and IN-FRONT-OF) includes their OPPOSITE (i.e., RIGHT-OF and BEHIND). The program thus can recognize various sorts of symmetry by replacing all input data structure relations by their relevant opposite, then comparing with the reference.

The next most complicated matching task after exact or symmetric matches is to match structures in isolation. Here the method is sequentially to match the input data against the whole reference data catalog of structures and determine which match is best (which difference description is most inconsequential). Easily computed scene characteristics can rule out some candidate models immediately.

The models contain arcs such as MUST-BE and MUST-NOT, expressing relations mandatory or forbidden relations. A match is not allowed between a description and a model if one of the strictures is violated. For instance, the program may reject a "house" immediately as not being a "pedestal," "tent," or "arch," since the pedestal top must be a parallelepiped, both tent components must be wedges, and the house is missing a component to support the top piece that is needed in the arch. These outright rejections are in a sense easy cases; it can also happen that more than one model matches some scene description. To determine the best match in this case, a weighted sum of differences is made to express the amount of difference.

The next harder case is to match structures in a complex scene. The issue here is what to do about evidence that is missing through obscuration. Two heuristics help:

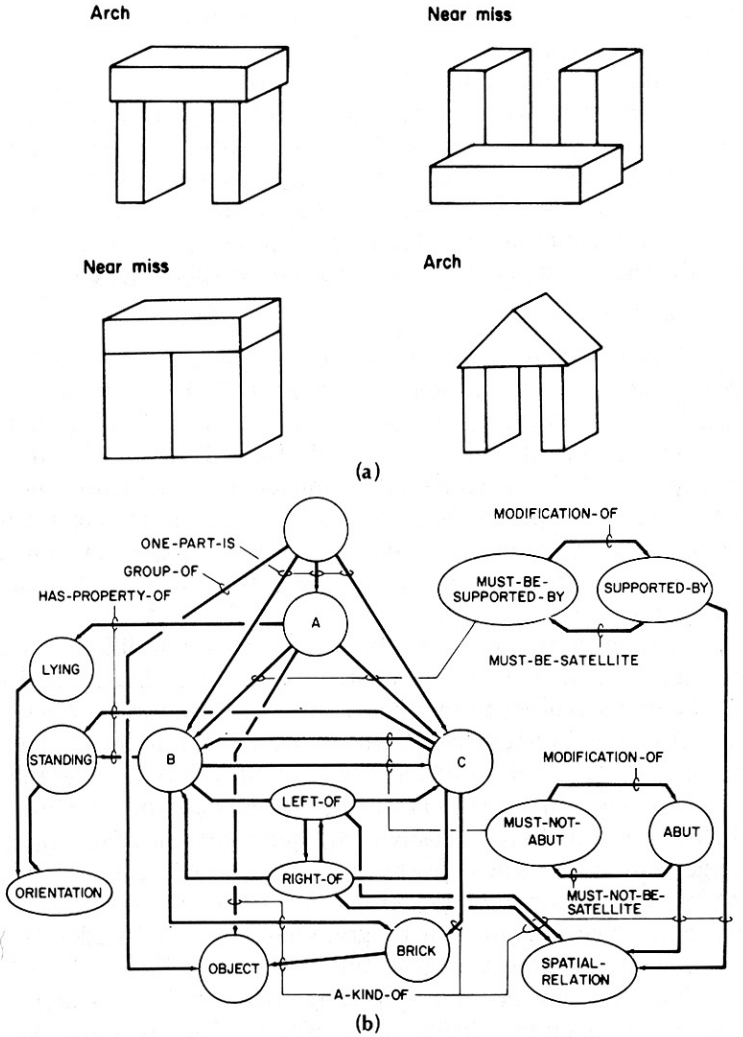


Fig. 11.10 (a) Several arches and non-arches. (b) The computer-generated arch description to be used for matching.

1. Objects that seem to have been stacked and could be the same type are of the same type.
2. Essential model properties may be hidden in the scene, so the match should not be aborted because of missing essential properties (though the presence of forbidden properties is enough to abort a match).

This latter rule is equivalent to Nevatia's rules about connectivity difference and missing input instance parts (see below). In terms of the general structure metric introduced earlier, neither Winston or Nevatia penalize the match for missing elements or relations in the reference data. One result of this is that the best match is sometimes missed in favor of the first possible match. Winston suggests that com-

plex scenes be analyzed by identifying subscenes and subtracting out the identified parts, as was done by Roberts.

Winston's program can learn shortcuts in matching strategy by itself; it builds for itself a similarity network relating models whose differences are slight. If a reference model does not quite fit an input structure, the program can make an intelligent choice of the next model to try. A good choice is a model that has only minor differences with the first. This self-organization and cataloging of the models according to their mutual differences is a powerful way to use matching work that is already performed to guide further search for a good match.

Backtrack Search

Nevatia addresses a domain of complex articulated biological-like forms (hands, horses, dolls, snakes) [Nevatia 1974]. His strategy is to segment the objects into parts with central axes and "cross section" (not unlike generalized cylinders, except that they are largely treated in two dimensions). The derived descriptions of objects contain the connectivity of subparts, and descriptions of the shape and joint types of the parts. Matching is needed to compare descriptions and find differences, which can then be explained or used to abort the match. Partial matches are important (as in most real-world domains) because of occlusions, noise, and so on.

A priori ideas as to the relative importance of different aspects of structures are used to impose a hierarchical order on the matching decision tree. Nevatia finds this heuristic approach more appealing than a uniform, domain-independent one such as clique finding. His system knows that certain parts of a structure are more important than others, and uses them to index into the reference data catalog containing known structures. Thus relevant models for matching may be retrieved efficiently on the basis of easily-computed functions of the input data. The models are generated by the machine by the same process that later extracts descriptions of the image for recognition. Several different models are stored for the same view of the same object, because his program has no idea of model equivalence, and cannot always extract the same description.

The matching process is basically a depth-first tree search, with initial choices being constrained by "distinguished pieces." These are important pieces of image which first dictate the models to be tried in the match, and then constrain the possible other matches of other parts.

There is a topological and a geometrical component to the match. The topological part is based on the connectivity of the "stick figure" that underlies the representation. The geometrical part matches the more local characteristics of individual pieces. Consider Nevatia's example of matching a doll with stored reference descriptions, including those of a doll and a horse.

By a process not concerning us here, the doll image is segmented into pieces as shown in Fig. 11.11. From this, before any matching is done, a connection graph of pieces is formed, as shown in Fig. 11.12.

This connection graph is topologically the same as the reference connection graph for the doll, which looks as one would expect. In both reference and input, "distinguished pieces" are identified by their large size. During reference forma-

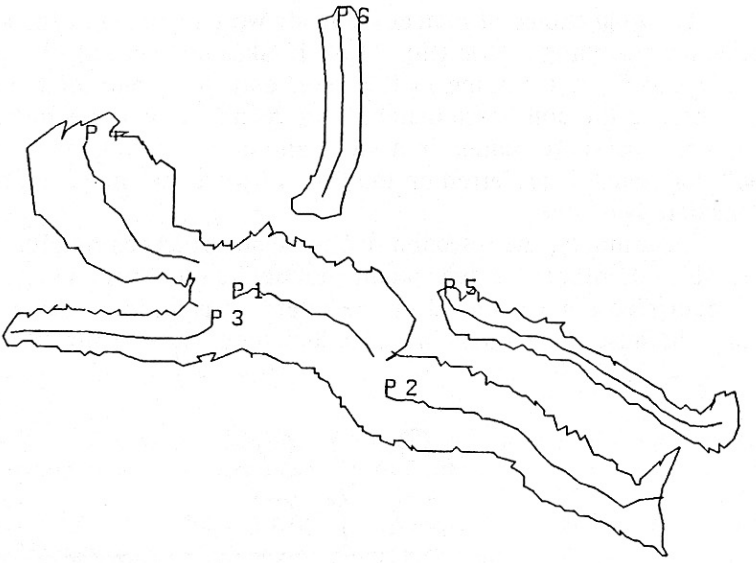


Fig. 11.11 A view of a doll, with derived structure.

tion time, the two largest pieces were the head and the trunk, and these are the distinguished pieces in the reference. There are the same pieces picked as distinguished in the instance to be matched consistent with the hierarchical decision-tree style, distinguished pieces are matched first.

Because of noise, connections at joints may be missed; because of the nature of the objects, extra joints are hardly ever produced. Thus there is a domain-dependent rule that an input piece with two other pieces connected at a single joint (a "two-ended piece") cannot match a one-ended reference piece, although the reverse is possible.

On the basis of the distinguished pieces in the input instance, the program decides that the instance could be a doll or a horse. Both these possibilities are evaluated carefully; Fig. 11.13 shows a schematic view of the process. Piece-match evaluation must be performed at the nodes of the tree to determine which pieces at a joint should be made to correspond.

The final best match between the doll input and the horse reference model is diagrammed in Fig. 11.14. This match is not as good as the match between the doll input and the doll reference.

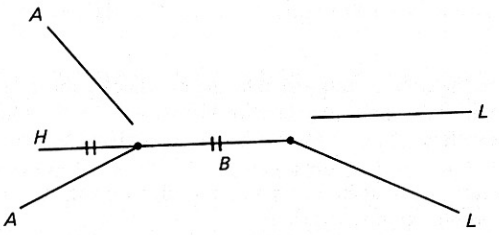


Fig. 11.12 Connection graph of the doll.

The final choice of matches is made with a version of the general relational structure matching metric (Eq. 11.3). It takes into account the connectivity relations, which are the same in this case, and the quality of the individual piece matches. In the doll-horse match, more reference parts are missing, but this can happen if parts are hidden in a view, and do not count against the match. The doll-doll match is preferred on the basis of piece matching, but both matches are considered possible.

In summary, the selection of best match proceeds roughly as follows: unacceptable differences are first sought (not unlike the Winston system). The number of input pieces not matched by the reference is an important number (not vice versa, because of the possibility of hidden input parts). Only elongated, large parts

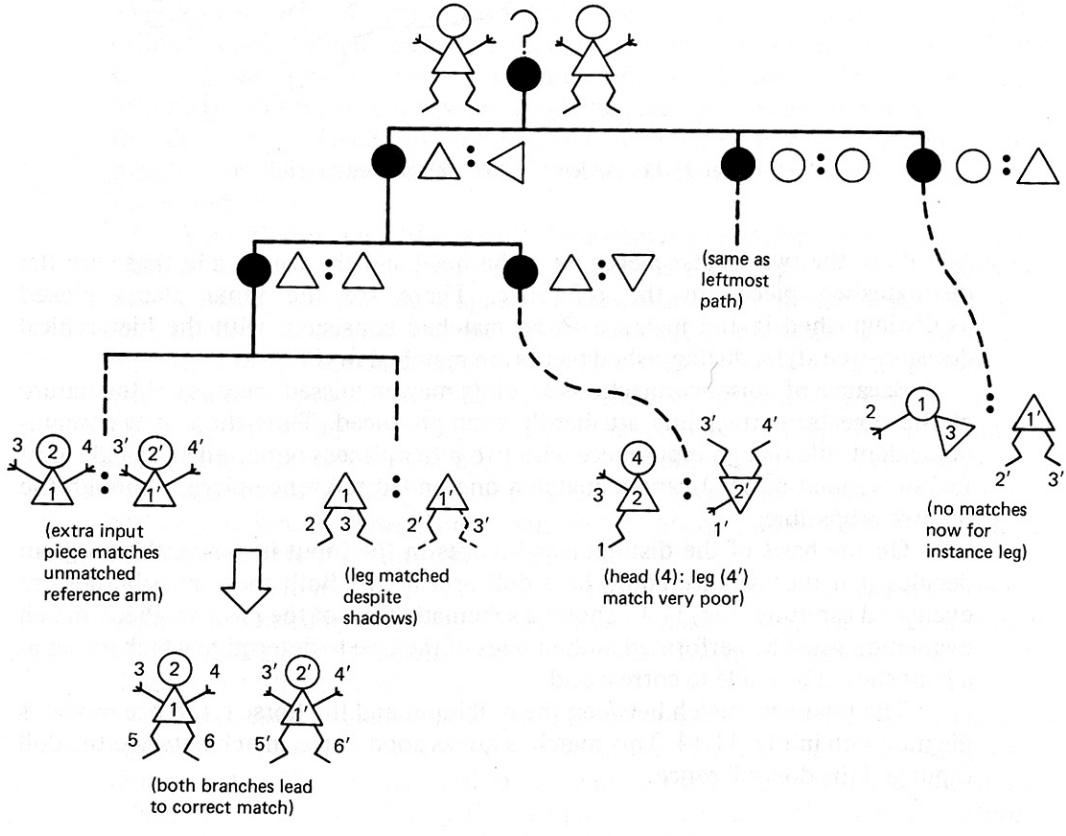


Fig. 11.13 A pictorial guide to the combinations tried by the matcher establishing the best correspondence of the doll input with the doll reference. The graphic shapes are purely pedagogical: the program deals with symbolic connectivity information and geometric measurements. Inferences and discoveries made by the program while matching are given in the diagram. A:B means that structure A is matched with structure B, with the numbered substructures of A matching their primed counterparts in B.

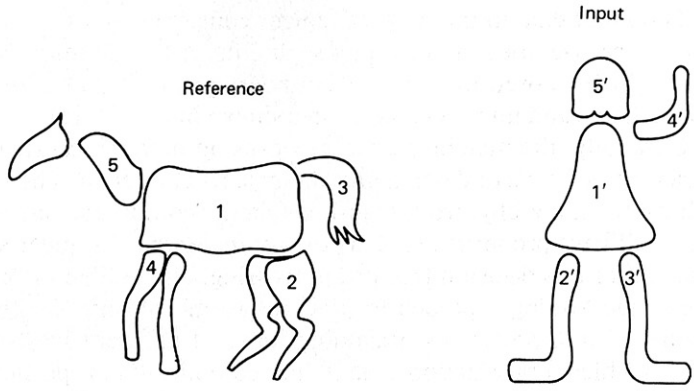


Fig. 11.14 The best match of the doll input with the horse reference model. One doll arm is unmatched, as is the horse head and two legs.

are considered for this determination, to eliminate small "noise" patches. The match with fewest unmatched input pieces is chosen.

If no deciding structural differences exist, the quality of piece matches determines the quality of the match. These correspond to the template cost term in Eq. (11.3). If a "significant" difference in match error exists, the better match is exclusively selected; if the difference is not so great as that, the better match is merely preferred.

Piece matching is a subprocess of joint matching. The difference in the number of pieces attached at the ends of the piece to be matched is the *connectivity difference.* If the object piece has more pieces connected to it than the model piece, the match is a poor one; since pieces may not be visible in a view, the converse is not true. If one match gives fewer excess input pieces, it is accepted at this point. If not, the goodness of the match is computed as a weighted sum of width difference, length-to-width ratio difference, and difference in acuteness of the generalized cylinders (Chapter 9) forming the pieces. The weighted sum is thresholded to yield a final "yes or no" match result. Shadowing phenomena are accommodated by allowing the input piece to be narrower than the reference model piece with no penalty. The error function weights are derived empirically; one would not expect the viewing angle to affect seriously the width of a piece, for example, but it could affect its length. Piece axis shapes (what sort of space curve they are) are not used for domain-dependent reasons, nor are cross section functions (aside from a measure of "acuteness" for cone-like generalized cylinders).

11.4.2 Decision Tree and Subgraph Isomorphism

A robotics program for versatile assembly [Ambler et al. 1975] uses matching to identify individual objects on the basis of their boundaries, and to match several individual blobs on a screen with a reference model containing the known location of multiple objects in the field of view. In both cases the best subgraph isomorphism between input and reference data structures is found when necessary by the clique-finding technique (Algorithm 11.2).

The input data to the part recognizer consist of silhouettes of parts with outlines of piecewise linear and circular segments. A typical set of shapes to be recognized might be stored in terms of boundary primitives as shown in Fig. 11.15a, with matchable and unmatchable scenes shown in Fig. 11.15b.

Generally, the matching process works on hierarchical structures which capture increasing levels of detail about the objects of interest. The matching works its way down the hierarchy, from high-level, easily computable properties such as size down to difficult properties such as the arrangements of linear segments in a part outline. After this decision tree pre-processing, all possible matches are computed by the clique-finding approach to subgraph isomorphism. A scene can be assigned a number of interpretations, including those of different views of the same part. The hierarchical organization means that complicated properties of the scene are not computed unless they are needed by the matcher. Once computed they are never recomputed, since they are stored in accessible places for later retrieval if needed. Each matching level produces multiple interpretations; ambiguity is treated with backtracking. The system recognizes rotational and translational invariance, but must be taught different views of the same object in its different gravitationally stable states. It treats these different states basically as different objects.

11.4.3 Informal Feature Classification

The domain of this work is one of small, curved tabletop objects, such as a teacup (Fig. 11.16) [Barrow and Popplestone 1971]. The primitives in models and image descriptions are regions which are found by a process irrelevant here. The regions have certain properties (such as shape or brightness), and they have certain parameterized relations with other regions (such as distance, adjacency, "aboveness"). The input and reference data are both relational structures. The properties and relations are the following:

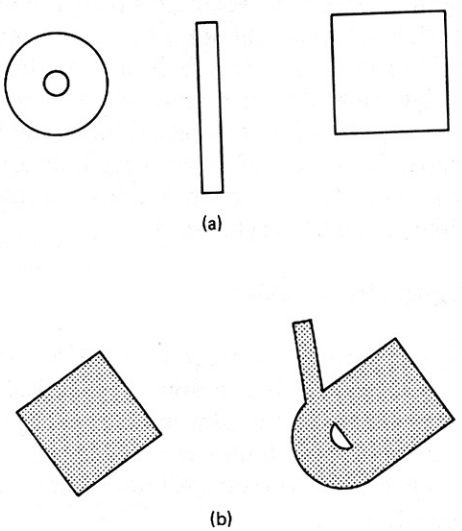


Fig. 11.15 A small catalog of part boundaries (a) and some sample silhouettes (b). The "heap" will not match any part very well, while the square can match the square model in four different ways, through rotations. Gross properties such as area may be used cheaply to reject matches such as the square with the axle.

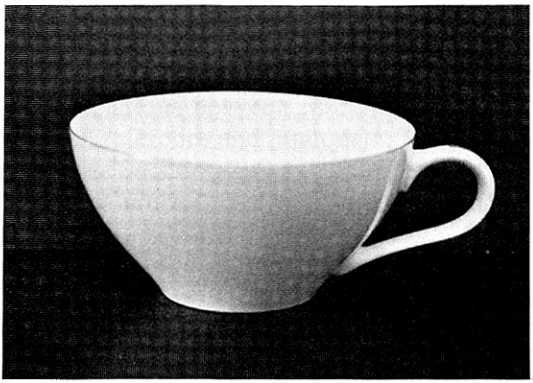


Fig. 11.16 An object for recognition by relational matching.

1. *Region Properties*

   Shape 1–Shape 6: the first six root mean square amplitudes of the Fourier components of the $\phi(s)$ curve [Chapter 8].

2. *Relations between Regions A and B*

   Bigger: Area(A)/Area(B)

   Adjacency: Fraction of A's boundary which also is a boundary of B.

   Distance: Distance between centroids divided by the geometric mean of average radii. The average radius is twice the area over the perimeter. Distance is scale, rotation, translation, reflection invariant.

   Compactness: $4*\pi*area/perimeter^2$

   Above, Beside: Vertical and horizontal distance between centroids, normalized by average radius. Not rotation invariant.

The model that might be derived for the cup of Fig. 11.16 is shown in Fig. 11.17.

The program works on objects such as spectacles, pen, cup, or ball. During training, views and their identifications are given to the program, and the program forms a relational structure with information about the mean and variance of the values of the relations and properties. After training, the program is presented with a scene containing one of the learned objects. A relational structure is built describing the scene; the problem is then to match this input description with a reference description from the set of models.

One approximation to the goodness of a match is the number of successes provided by a region correspondence. A one-region object description has 7 relations to check, a two-region object has 28, a three-region one has 63. Therefore, the "successes" criterion could imply the choice of a terrible three-region interpretation over a perfect one-region match. The solution adapted in the matching evaluation is first to grade failures. A failure weight is assigned to a trial match according to how many standard deviations σ from the model mean the relevant

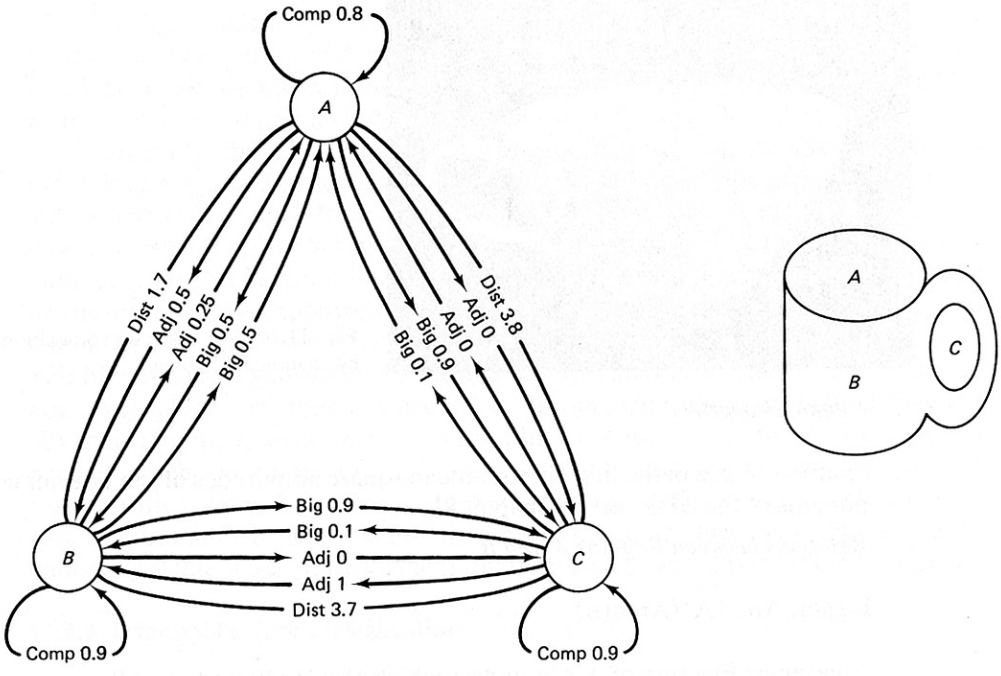


Fig. 11.17 Relational model for cups such as that of Fig. 11.16.

parameter is. From zero to three σ imply a success, or a failure weight of 0; from three to six σ, a failure weight of 1; from six to nine σ, failure weight of 2, and so on. Then the measure "trials–cumulative failure weight" is an improvement on just "successes." On the other hand, simple objects are often found as subparts of complex ones, and one does not want to reject a good interpretation as a complex object in favor of a less explanatory one as a simple object. The final evaluation function adapted is

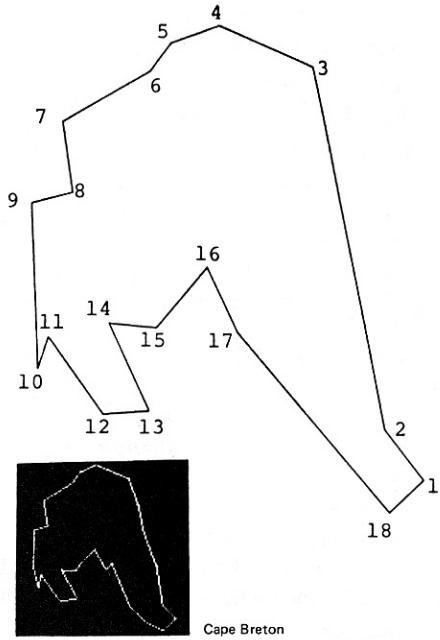
$$\text{Cost of Match} = \frac{1 - (\text{tries-failure weight})}{\text{number of relations}} + \frac{K}{\text{number of regions in view description}} \tag{11.5}$$

As in Eq. (11.4), the first term measures the average badness of matches between properties (unary relations) and relations between regions. The second term is inversely proportional to the number of regions that are matched, effectively increasing the cost of matches that explain less of the input.

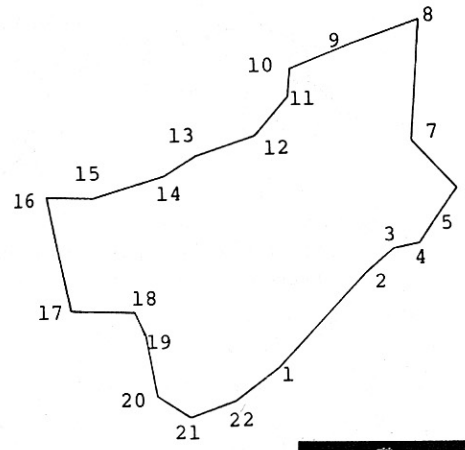
11.4.4 A Complex Matcher

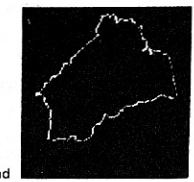
A program to match linear structures like those of Fig. 11.18 is described in [Davis 1976]. This matcher presents quite a diversity of matching techniques incorporated into one domain-dependent program.

The matching metric is very close to the general metric of Eg. (11.3). The match is characterized by a structural match of reference and input elements and a geometrical transformation (found by parameter fitting) which accounts for the spatial relations between reference and input. Davis forms an association graph between reference and input data. This graph is reduced by parallel-iterative relaxation (see Section 12.4) using the "spring functions" to determine which node associations are too costly. Eliminating one node–node match may render others

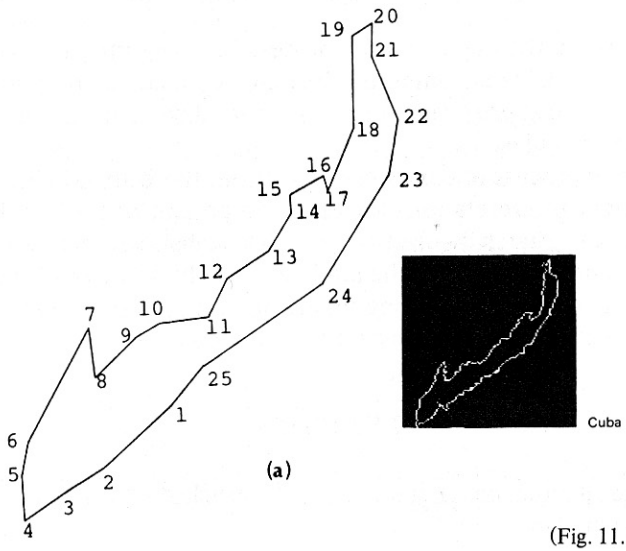


Cape Breton





Baffin Island



Cuba

(Fig. 11.18 continues on p. 380.)

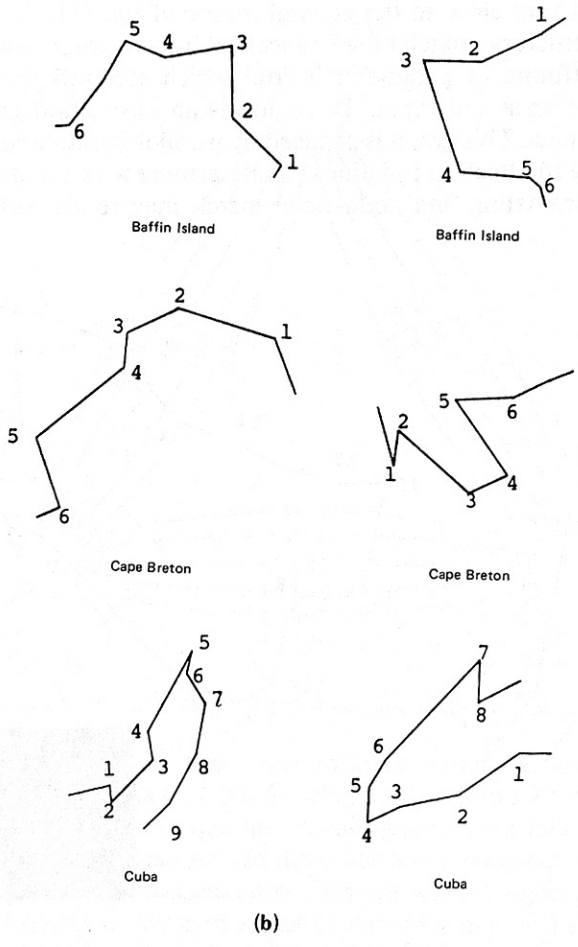


Fig. 11.18 (a) Reference and (b) input data for a complex shape matching program.

more unlikely, so the node-pruning process iterates until no more nodes are eliminated. What remains is something like an r-connected component of the graph, which specifies an approximate match supported by some amount of consistent relations between nodes.

After the process of constraint relaxation, there are still in general several locally consistent interpretations for each component of the input structure. Next, therefore, a tree search is used to establish global consistency and therefore the best match. The tree search is the familiar "best first" heuristic search through the partial match space, with pruning taking place between each stage of search again by using the parallel-iterative relaxation technique.

EXERCISES

11.1 Relational structures A and B are to be matched by the association-graph, clique-finding method.

Relational structure A: entities u, v, w, x, y, z.
relations $P(u), P(w), P(y), R(v), R(x), R(z)$,
$F(u, v), F(v, w), F(w, x), F(x, y), F(y, z), F(z, u)$

Relational structure B: entities a, b, c, d, e, f.
relations $P(a), P(b), P(d), Q(e), Q(f), R(c)$
$F(b, c), F(c, d), F(d, e), F(e, f), F(f, a)$.

(a) Construct graph structures corresponding to the structures A and B. Label the nodes and arcs.

(b) Construct the association graph of structures A and B.

(c) Visually find the largest maximal cliques in the association graph and thus the best matches between A and B. (There are three.)

11.2 Suppose in a geometric match that two input points on the xy plane are identified with two others taken to correspond with two reference points. It is known that the input data comes about only through rotation and translation of the reference data. Given the two input points (x_1, y_1) and (x_2, y_2) and the two reference points (x'_1, y'_1) and (x'_2, y'_2), one way to find the transformation from reference to input is to solve the equation

$$\sum_{i=1}^{2} [x_i - (ax'_i + by'_i + c)]^2 + [y_i - (bx'_i + ay'_i + d)]^2 = 0$$

The resulting values of a, b, c, and d represent the desired transformation. Solve the equation analytically to get expressions for a, b, c, and d in terms of the reference and input coordinates. What happens if the reference and input data are not related by simple rotation and translation?

11.3 What are the advantages and disadvantages of a uniform method (such as subgraph isomorphism algorithm approach) to matching as compared to an ad hoc (such as a decision-tree approach with various empirically derived metrics) one?

11.4 In the worst case, for graphs of n nodes, how many partial solutions total will Algorithm 11.1 have to proceed through? Construct "worst case" graphs X and Y (label their nodes $1, \ldots, n$, of course), assuming that nodes of Y are selected in ascending order at any stage.

11.5 Find out something about the state of associative memories in computers. How do they work? How are they used? Would anything like this technology be useful for computer vision? Introspect about familiar phenomena of visual recall, recognition, and memory. Do you have a theory about how human visual memory could possibly work?

11.6 What graph of N nodes has the maximum number of maximal cliques? How many does it have?

11.7 Think about reasoning by analogy and find out something about programs that do analogical reasoning. In what sense can analogical process be used for computer vision, and technically do the matching techniques necessary provide any insight?

11.8 Compare Nevatia's structure matching with Hinton's relaxation-based puppet recognition (Chapter 12).

11.9 Verify the observation made in Section 11.4.3 about the number of relations that must be checked between regions (one region, 7; two regions, 28; three regions, 63; etc.).

REFERENCES

AHO, A. V., J. E. HOPCROFT and J. D. ULLMAN. *The Design and Analysis of Algorithms.* Reading, MA: Addison-Wesley, 1974.

AMBLER, A. P., H. G. BARROW, C. M. BROWN, R. M. BURSTALL, and R. J. POPPLESTONE. "A versatile computer-controlled assembly system." *Artificial Intelligence 6*, 2, 1975, 129–156.

BARROW, H. G. and R. J. POPPLESTONE. "Relational descriptions in picture processing." In *MI6*, 1971.

BARROW, H. G., J. M. TENENBAUM, R. C. BOLLES, and H. C. WOLF. "Parametric correspondence and chamfer matching: two new techniques for image matching." *Proc.*, DARPA IU Workshop, May 1978, 21-27.

BERGE, C. *Graphs and Hypergraphs* 2nd rev. ed.. New York: American Elsevier, 1976.

BERZTISS, A. T. "A backtrack procedure for isomorphism of directed graphs." *J. ACM 20*, 3, July 1973, 365-377.

BITTNER, J. R. and E. M. REINGOLD. "Backtrack programming techniques." *Comm. ACM 18*, 11, November 1975, 651-656.

BRON, C. and J. KERBOSCH. "Algorithm 457: finding all cliques in an undirected graph (H)." *Comm. ACM 16*, 9, September 1973, 575-577.

CORNEIL, D. G. and C. C. GOTLIEB. "An efficient algorithm for graph isomorphism." *J. ACM 17*, 1, January 1970, 51-64.

DAVIS, L. S. "Shape matching using relaxation techniques." *IEEE-PAMI 1*, 1, January 1979, 60-72.

FISCHLER, M. A. and R. A. ELSCHLAGER. "The representation and matching of pictorial structures." *IEEE Trans. Computers 22*, 1, January 1973, 67-92.

HARALICK, R. M. and G. L. ELLIOTT. "Increasing tree search efficiency for constraint satisfaction problems." *Proc.*, 6th IJCAI, August 1979, 356-364.

HARARY, F. *Graph Theory.* Reading, MA: Addison-Wesley, 1969.

KNODEL, W. "Bestimmung aller maximalen vollstandigen Teilgraphen eines Graphen G nach Stoffers." *Computing 3*, 3, 1968, 239–240 (and correction in *Computing 4*, p. 75).

NEVATIA, R. "Structured descriptions of complex curved objects for recognition and visual memory." AIM-250, Stanford AI Lab, October 1974.

NILLSON, N.J. *Principles of Artificial Intelligence.* Palo Alto, CA: Tioga, 1980.

OSTEEN, R. E. and J. T. TOU. "A clique-directed algorithm based on neighbourhoods in graphs." *International J. Computer and Information Science 2*, 4, December 1973, 257–268.

REINGOLD, E. M., J. NIEVERGELT, and N. DEO. *Combinatorial Algorithm Theory and Practice.* Englewood Cliffs,N. J.: Prentice-Hall, 1977.

SCHUDY, R. B. and D. H. BALLARD. "Model-directed detection of cardiac chambers in ultrasound images." TR12, Computer Science Dept., Univ. Rochester, November 1978.

SHAPIRO, L. G. and R. M. HARALICK. "Structural descriptions and inexact matching." Technical Report CS79011-R, Computer Science Dept., Virginia Polytechnic Institute, November 1979.

ULLMAN, J. R. "An algorithm for a subgraph isomorphism." *J. ACM 23*, 1, January 1976, 31-42.

WINSTON, P. H. "Learning structural descriptions from examples." In *PCV*, 1975.