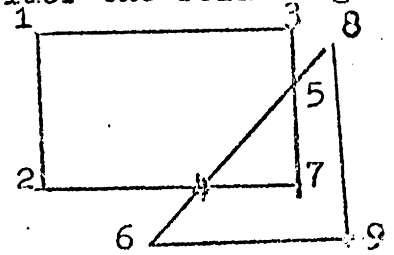
Artificial Intelligence Project--RLE and MIT Computation Center
Memo 18--Some Results from a Pattern Recognition Program Using LISP

by

Louis Hodes

This paper describes some aspects of an elaborate pattern recognition system being programmed by the author under the supervision of Marvin Minsky. A more detailed discussion is forthcoming as a Lincoln Laboratory group report.

Consider the following drawing



The drawing is represented by a LISP expression which just lists vertices and lines adjacent to the vertices

(1, ((2, ...), (3, ...)), 2, ((1, ...), (4, ...)), 3, ((1, ...), (5, ...)),
4, ((2, ...), (6, ...) , (7, ...), (5, ...)), 5, ((4, ...), (7, ...), (8, ...),
(3, ...)), 6, ((9, ...), (4, ...)), 7, ((4, ...), (5, ...)), 8, ((5, ...),
(9, ...)), 9, ((6, ...), (8, ...)))

Occurences of ... in the descriptions of lines stand for information about the lines themselves. For example, numbers representing beginning angle, end angle, length and curvature were used.

If p is a list structure description of a pattern then thru (p) is a list of triplets defining vertices which lines pass through. Let p be the above description. Then thru (p) = ((2,4,7), (3,5,7), (4,5,8), (5,4,6), (6,4,5), (7,4,2), (7,5,3), (8,5,4))
Check with the figure.

From thru a list of all lines going through vertices can be obtained as n-tuples

allines (thru(p)) = ((2,4,7), (3,5,7), (4,5,8), (5,4,6),
(6,4,5), (7,4,2), (7,5,3), (8,5,4),
(6,4,5,8), (8,5,4,6))

There is also a program "triangle" which, if p again is the above description, yields

triangle (p) = ((4,5,7), (5,4,7), (4,7,5), (5,7,4), (6,8,9),
(6,9,8), (7,4,5), (7,5,4), (8,6,9), (8,9,6),
(9,6,8), (9,8,6))

i.e. triangle finds all triplets whose vertices form triangles

There is a program construct (m,p) where p is a pattern description and m is a list of n-tuples $n \geq 3$. Construct has as output a new pattern description p^1 which contains p and has just those lines added which are represented by n-tuples of m.

Again with p as the above description, construct (allines (thru(p)),p)=

(1, ((2,...), (3,...)), 2, ((1,...), (4,...), $\underline{(7,\ldots)}$),
3, ((1,...), (5,...), $\underline{(7,\ldots)}$), 4, ((2,...), (6,...),
(7,...), (5,...), $\underline{(8,\ldots)}$), 5, ((4,...), (7,...),
(8,...), (3,...), $\underline{(6,\ldots)}$), 6, ((9,...), (4,...),
$\underline{(5,\ldots)}$, $\underline{(8,\ldots)}$), 7, ((4,...), (5,...) $\underline{(2,\ldots)}$,
$\underline{(3,\ldots)}$), 8, ((5,...), (9,...), $\underline{(4,\ldots)}$, $\underline{(6,\ldots)}$),
9, ((6,...), (8,...)))

The added lines were underlined.

"Midverts" just extracts the centers from a list of triplets and gives all distinct centers as output. So

```
        midverts (thru(p)) = (4,5)
```

for the above p.

In the program extract (v,p) v is a list of numbers and p is a pattern. The output of extract is a new pattern that has none of the lines adjacent to vertices mentioned in v.

```
        So extract (midverts(thru(p)), construct (allines (thru
    (p)),p))
        = (1, ((2,...), (3,...)), 2, ((1,...), (7,...)),
          3, ((1,...), (7,...)), 6, ((9,...), (8,...)),
          7, ((2,...), (3,...)), 8, ((9,...), (6,...)),
          9, ((6,...), (8,...))
    and triangle (extract----- )
        = ((6,8,9), (6,9,8), (8,6,9), (8,9,6), (9,6,8), (9,8,6))
```

This last function is a general method for finding triangles in overlapping patterns.

My purpose in writing this memo was to make some comments on the form of the program which was used to find triangles in overlapping patterns. This program can be considered as a sequence of three steps - construction, extraction and detection.

The first comment I would like to make is that a construction-extraction sequence may be a useful way of working non-visual pattern recognition problems.

The other two comments deal with human pattern recognition, and they are really two reasons why I think construction-extraction processes are a good way to simulate human pattern recognition. First a sequence of alternating constructions and extractions can be looked on as a way to get higher levels of abstraction. It has a hierarchial structure. Second, one can possibly account for ambiguous pictures as being the result of an extraction preceded by different constructions.

CS-TR Scanning Project
Document Control Form

Date : 11 / 30 / 95

Report # AIM - 18

Each of the following should be identified by a checkmark:
Originating Department:

☒ Artificial Intellegence Laboratory (AI)
☐ Laboratory for Computer Science (LCS)

Document Type:

☐ Technical Report (TR) ☒ Technical Memo (TM)
☐ Other:

Document Information

Number of pages: 3 (7 - images)
Not to include DOD forms, printer intstructions, etc... original pages only.

Originals are:
☒ Single-sided or
☐ Double-sided

Intended to be printed as :
☒ Single-sided or
☐ Double-sided

Print type:

☐ Typewriter ☐ Offset Press ☐ Laser Print
☐ InkJet Printer ☐ Unknown ☒ Other:

Check each if included with document:

☐ DOD Form ☐ Funding Agent Form ☐ Cover Page
☐ Spine ☐ Printers Notes ☐ Photo negatives
☐ Other:

Page Data:

Blank Pages(by page number):

Photographs/Tonal Material (by page number):

Other (note description/page number):

Description :	Page Number:
IMAGE MAP! (1-3)	1-3
(4-7) SCANCONTROL, TRGT'S (3)	

Scanning Agent Signoff:
Date Received: 11 / 30 / 95 Date Scanned: 12 / 13 / 95 Date Returned: 12 / 14 / 95

Scanning Agent Signature: Michael W. Cook

Rev 9/94 DS/LCS Document Control Form cstrform.vsd