

**WORKING PAPER 52**

**TRACKING WIRES ON PRINTED CIRCUIT BOARDS**

by

**TIM FININ**

**Massachusetts Institute of Technology**

**Artificial Intelligence Laboratory**

**October, 1973**

**Abstract**

This working paper describes a collection of LISP programs written to examine the backs of printed circuit boards. These programs find and trace the conductive wires plated on the insulating material. The "pads", or solder connections between these plated wires and leads from components on the front of the board, are also recognized and located by these programs.

Work reported herein was conducted at the Artificial Intelligence Laboratory, a Massachusetts Institute of Technology research program supported in part by the Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under Contract Number N00014-70-A-0362-0005.

Working Papers are informal papers intended for internal use.

## I. INTRODUCTION

This working paper describes a collection of LISP programs written to examine the backs of printed circuit boards. These programs find and trace the plated wires as well as detect "pads" on them. I see this as a part of preliminary work toward a number of goals:

### 1. The mini-robot

The A.I. Lab's Mini-Robot project has chosen electronics inspection and repair as a task area for new research in vision and robotics. A system such as this one could be used as a specialist in a number of applications. For example, suppose the mini-robot wishes to place a probe on a rather inaccessible transistor lead. A solution might be to turn the board over, find the pad corresponding to that lead, and trace the circuit leading from that pad to another, more accessible spot.

### 2. Industrial applications

The time is ripe for the application of sophisticated vision to industrial problems, particularly in such areas as visual inspection. Already work has been done in developing an automatic inspector to examine plated wires on printed circuit boards <1>. These programs might form the basis for a system which could inspect solder joints on such boards.

### 3. Vision and higher level knowledge

It is generally felt that the mini-world of white-faced blocks has been exhausted, at least visually. This work is a step in the exploration of a new one - the mini-world of circuit boards and

electrical components. This world should allow for, and require, a richer base of higher-level knowledge. At the same time, it is still narrow enough to make it tractable. Although these programs themselves do not require a great deal of higher-level knowledge, they could be used in a larger system which knows about the front and back of circuit boards and how they interact. For example, examining the back of a circuit board might be a good way to suggest where other programs should look for components on the front.

## II. THE PROBLEM

### 2.1 Goals And Strategies

Several different strategies might be used to examine the backs of circuit boards. We might, for example, use a multiple pass procedure in which "feature points" are first detected by local predicates, and then later grouped together to form wires. The Horn-Binford line finder <2> might be modified to perform reasonably well at this task. Another approach would center around a wire tracking system, similar to more recent line finders <3,4>.

The approach selected must depend on the particular goals we wish to accomplish. In this work, several applications were envisioned:

1. Find a particular pad and trace the wires leading from it to another pad.
2. Find a particular pad and inspect the soldering.
3. Find all the pads in a specified area.
4. Find all the pads on the board.
5. Develop a complete description of the board, including the location of pads, paths followed by wires, etc.

Our strategy must also depend on what the image of the back of a printed circuit board looks like under our input device. Several boards have been examined, and the programs tested on them. Figure 2.1 shows the result of scans across typical wires on the board. Note that there is good contrast between the dark insulating material and the bright plated wires.

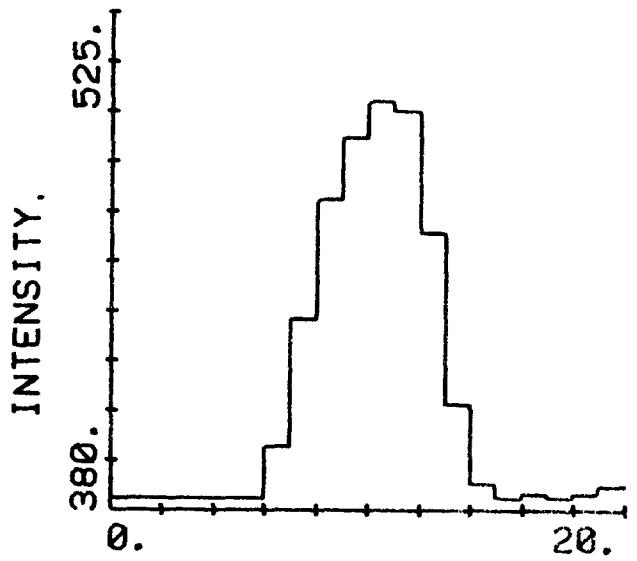
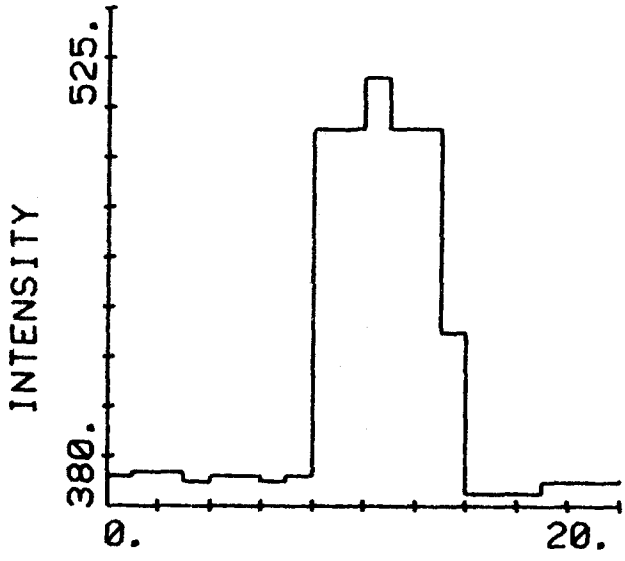


FIGURE 2.1

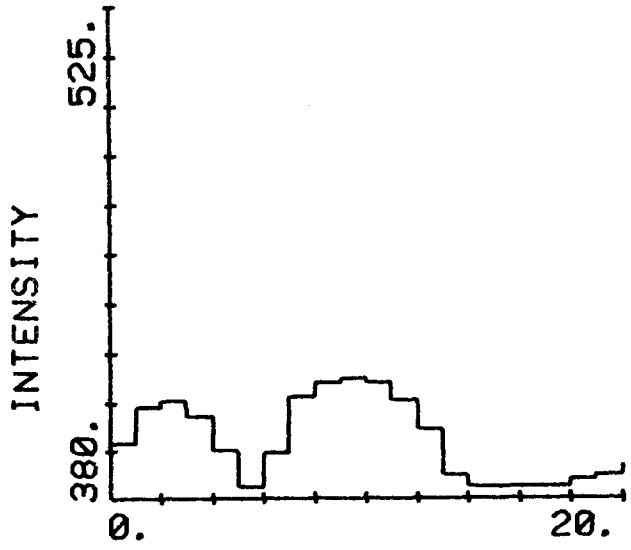


FIGURE 2.2

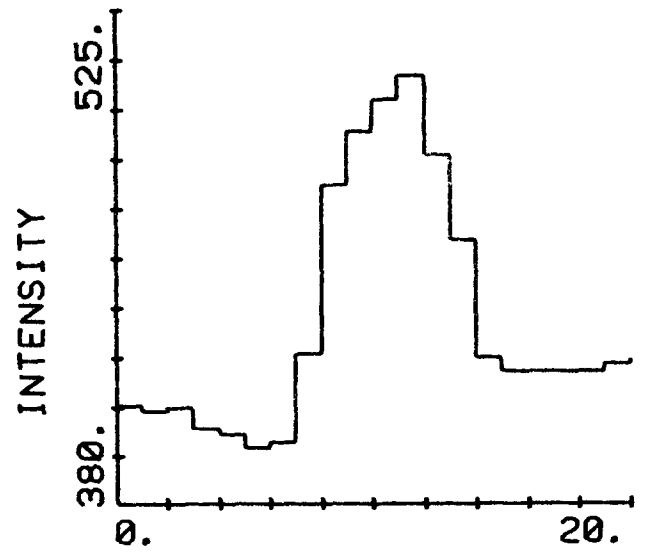


FIGURE 2.3

Were contrast this good throughout the picture, the problem would be considerably simplified to that of dealing with a binary image. Contrast is significantly reduced, however, in certain cases. Figure 2.2 shows another scan across a wire. Here, the scan has fallen very close to a pad. The blob of solder on the pad casts a shadow which has greatly reduced the intensity. Additional variations in the contrast can occur when the board is constructed of a semi-translucent material. Figure 2.3 shows a scan where the intensity of the background insulator is raised by printing on the front of the board.

Because of a number of factors, a tracking type approach was taken for this work. Tracking is the natural choice for the first of the previously mentioned goals, that of following a wire from one pad to another. This task is included in proposed scenarios for the Micro-Automation project. The other goals should also be amenable to a tracking approach.

Recognizing pads is also more easily done by a tracking procedure because of their non-local nature. A pad can be easily recognized as a "bulge" in a wire. Pads can thus be detected by monitoring the width of the wire as we track it. If the width suddenly increases, and then decreases, a pad is hypothesized. A tracking process will also allow us to make use of local context in other ways. For example, if we hypothesize a pad when tracking a wire, we can anticipate shadows and adjust our thresholds accordingly.

### **2.3 Representation - The Data Base**

For the purposes of this work, the following conventions are used.

A circuit board is made up of one or more CIRCUITS, each of which is comprised of a connected (and possibly branching) path of plated wire. NODEs divide a circuit into one or more SEGMENTS, which are sections of a circuit. A NODE is either :

1. The end of a branch of the circuit.
2. A place where the circuit branches.
3. A PAD, or place where a wire comes thru from the front of the board and is soldered down.

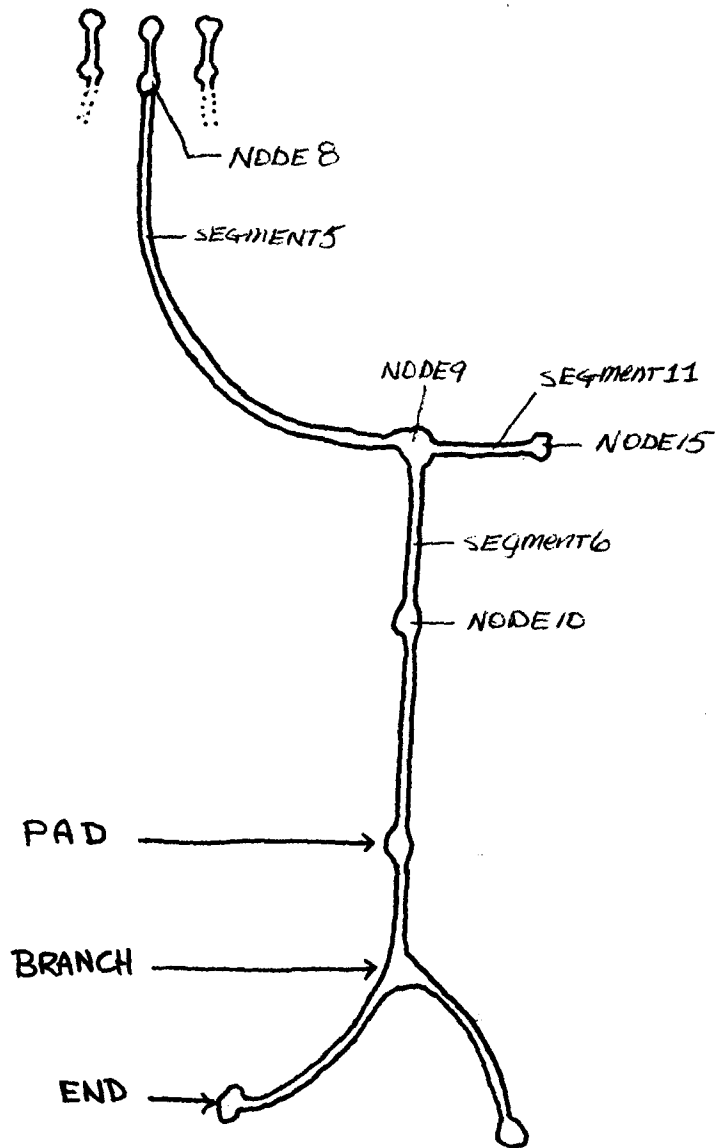
Figure 2.4 shows a typical circuit and its parts.

The data base is made up of a network of information which is stored on property lists. The network character is achieved by the judicious use of "back pointers". For example, NODE9 in figure 2.4 would look like this:

```

                                NODE9
COORDINATES      : (512 432)
CIRCUIT          : circuit3
SEGMENTS         : (segment5 segment6 segment11)
TYPE             : branch
SUSPECTS        : (51.3 178.5 268.7 355.4)
FAILED-SUSPECTS : (51.3)
HYPOTHESIZED-TYPE : pad
NEIGHBOOR-NODES : (node8 node10 node11)

```



A TYPICAL CIRCUIT

FIGURE 2.4



### III. THE PROGRAMS

This section contains a description of the general strategies used in examining the backs of circuit boards and a more detailed explanation of some of the algorithms. Figure 3.1 shows the major higher level functions and the flow of control between them.

The top level function, TRACE-CIRCUIT, is given the coordinates of a point on or very near a piece of conductor. It centers itself on the conducting wire and calls TRACE-FROM-NODE to begin tracking the circuit.

The action of the system revolves around two main routines: TRACE-FROM-NODE and TRACE-SEGMENT. Starting at a node, the function TRACE-FROM-NODE finds all segments leaving that node and applies TRACE-SEGMENT to each of them in turn. TRACE-SEGMENT tracks the segment until it ends, or until another node is hypothesized. It then applies TRACE-FROM-NODE to this point. Thus a circuit is explored in a sort of "depth first tree search" manner.

#### 3.1 Tracing from a Node

Finding and tracking all the segments leaving a node is done using a suspect-generation/verification technique. The basic procedure is outlined as follows:

1. Suspect segments leaving the node are proposed.
2. If any segments are known to end at this node, the suspects which correspond to them are deleted from the suspect-list.

Unless the node is the initial one created by TRACE-CIRCUIT,

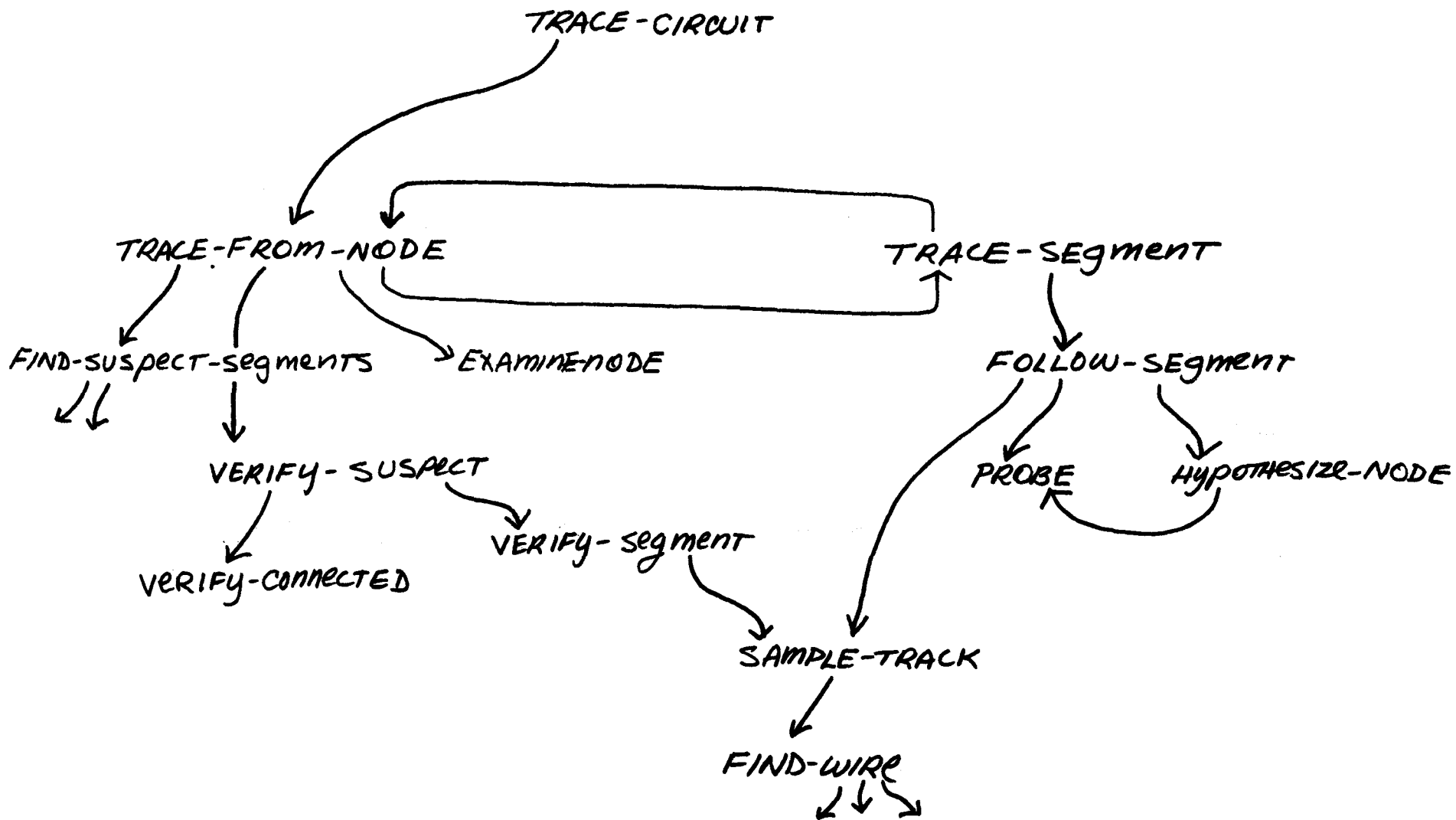


FIGURE 3.1

there will be at least one such known subject.

3. A suspect is either verified or rejected.
4. If verified, the segment is passed to the function TRACE-SEGMENT for tracking.
5. Steps (2) and (3) are repeated until all the suspects found in (1) have been checked.
6. The node is then re-examined to determine its type.

### Finding Suspect Segments

Circular search is the basic technique used for generating suspects. A circle of points around the node is scanned and examined to find suspect segments. The radius of this circle is determined by trading off several factors. Since nodes often appear as large blobs which cast shadows, we want to sample points far enough away to assure a good view of the segments leading into it. Similarly, a large radius will be more accurate if we start with a poor initial estimate of the node's center. On the other hand, a large circle will intersect unconnected neighboring circuits, resulting in many spurious suspects. A good value for this radius (eFIND-SUSPECTS-RADIUS) was found to be about two to three times the width of a typical wire (see figure 3.2).

Depending on various conditions, one of two procedures is used in examining the points. If the picture is noisy, the contrast poor, or there is a good deal of low-frequency intensity drift across the scene, these points are filtered by a first differencing operation and averaged over a distance eFIRST-DIFFERENCE-DELTA (about one half the typical wire width). Maximum and minimum peaks in these values are then found using

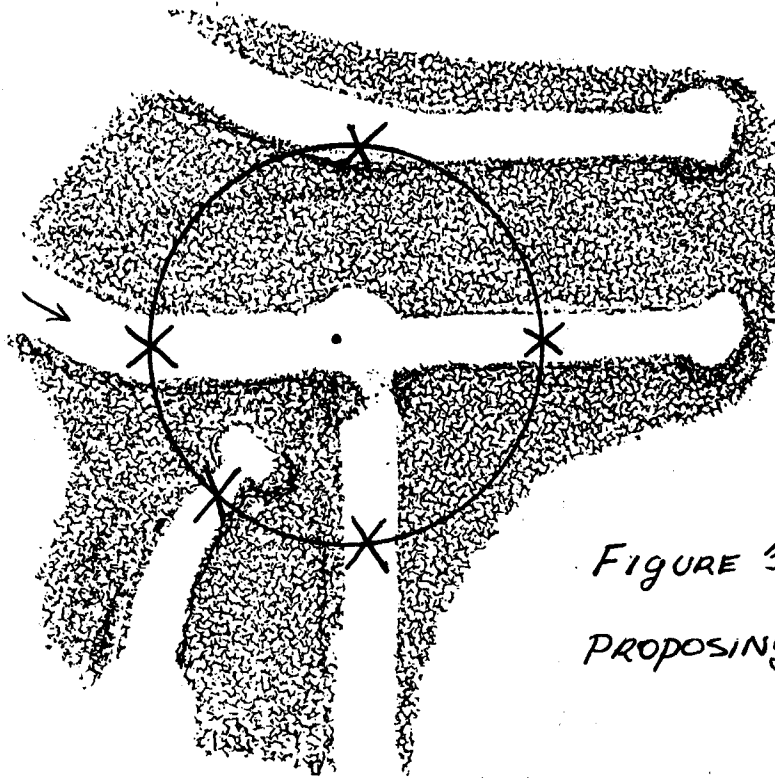


FIGURE 3.2

PROPOSING SUSPECTS

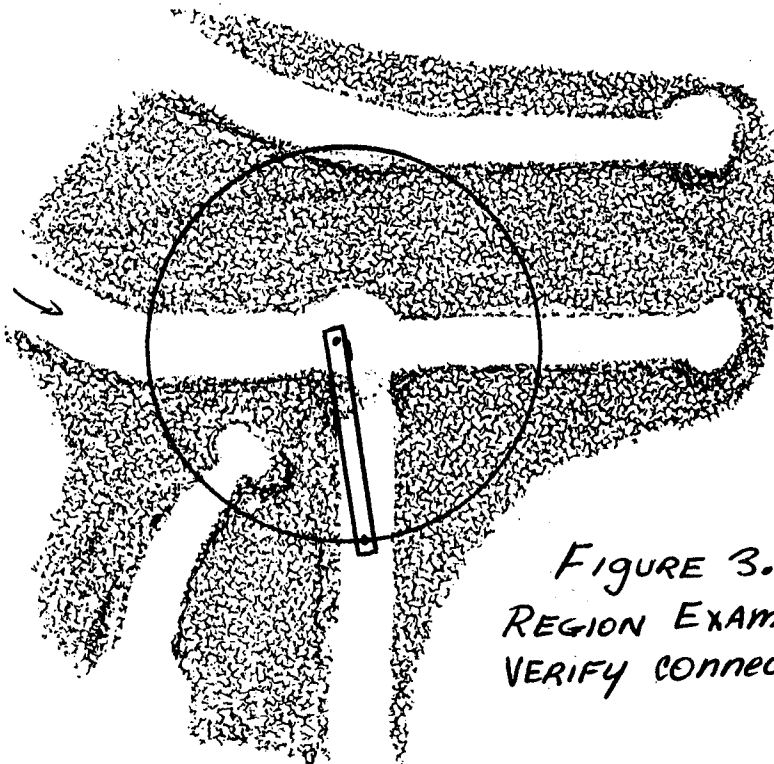


FIGURE 3.3

REGION EXAMINED TO  
VERIFY CONNECTEDNESS.

an algorithm from Sharai <3>. The peaks are then matched to form positive and negative pairs, which correspond to cross-sections of the wire. Criteria for pairing are:

1. The height of the two peaks must be within 75% of one another.
2. The separation between them must be between certain minimum and maximum values (eMINIMUM-WIRE-WIDTH and eMAXIMUM-WIRE-WIDTH).

When two peaks are so matched, they are merged to form a "wire", and its width, maximum intensity, average intensity, and position are noted. In addition, the average intensity of the "background" around the wire is noted. This, together with the wire's average intensity, is later used to calculate intensity thresholds which can be applied to detect the wire.

If the picture is of better quality, the filtering-averaging step is eliminated. In this case, the circular intensity array is examined directly for positive peaks, again using an algorithm similar to Sharai's. Any peak above the threshold eWIRE-TH which is within the minimum and maximum allowable widths is taken as a possible wire. Again the widths, intensities, and positions of these peaks are recorded.

### Suspect Verification

Once a suspect has been found, a two step verification procedure is applied. We first determine if the suspect segment actually runs into the node, or if it is part of a separate unconnected circuit. This is done by scanning a band of points along a line between the center of the node and the location of the suspect and comparing their intensities to a threshold.

A node is typically a pad where a wire from the front of the circuit board comes thru and is soldered down. Under all but the most diffused lighting, the lead and solder blob cast shadows which can obscure part of the node and segments leading into it. Another problem arises when we are not situated on the center of the node. In this case the band of points may skirt along the edge of the background insulating material. For these reasons, the following criteria for connectedness are used.

The sampled points are examined for values less than the threshold `eCONNECTED-TH`. This threshold is chosen to be slightly less than the current value of `eWIRE-TH` to offset the problem of weakly shadowed regions. To handle the problem of small darkly shadowed patches and slight excursions into the background, we will accept points below the threshold if they are isolated or occur in short sequences. The maximum length of such a sequence, `eCONNECTED-GAP`, is chosen to be about one-third the typical wire width. If no long dark stretches are found, then the segment is taken to be connected to the node (see figure 3.3).

In the second part of the verification, we determine that the suspect is indeed a segment, and get a better estimate on its width, average brightness, and local direction. This is done by the function `SAMPLE-TRACK` which applies `FIND-WIRE` at several points along the hypothesized direction of the segment (see figure 3.4). The function `FIND-WIRE` scans a band of points perpendicular to the hypothesized direction of the segment and uses procedures similar to those in the circular search to detect the wires. If more than one is found, the one

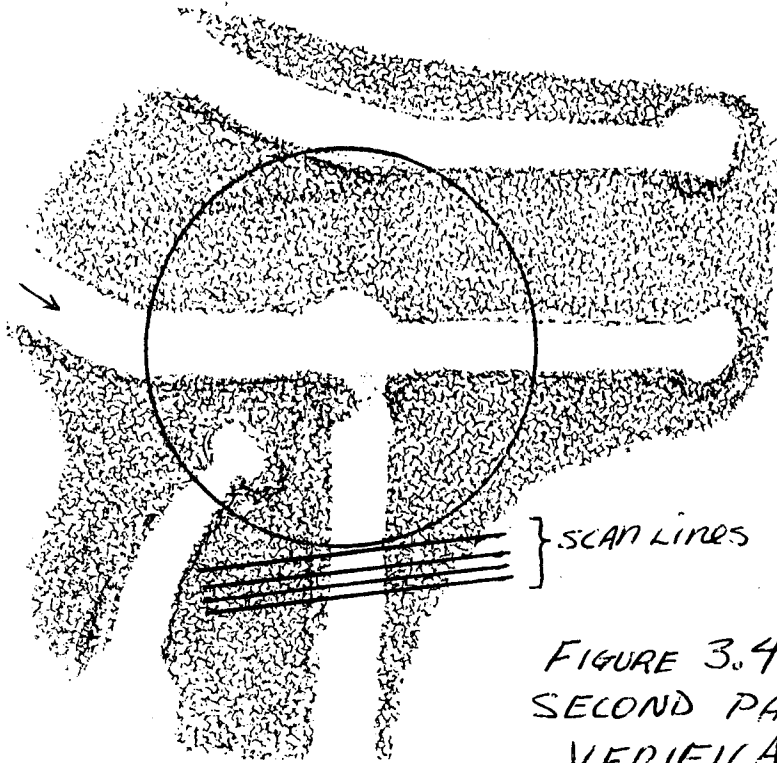


FIGURE 3.4  
SECOND PART OF  
VERIFICATION

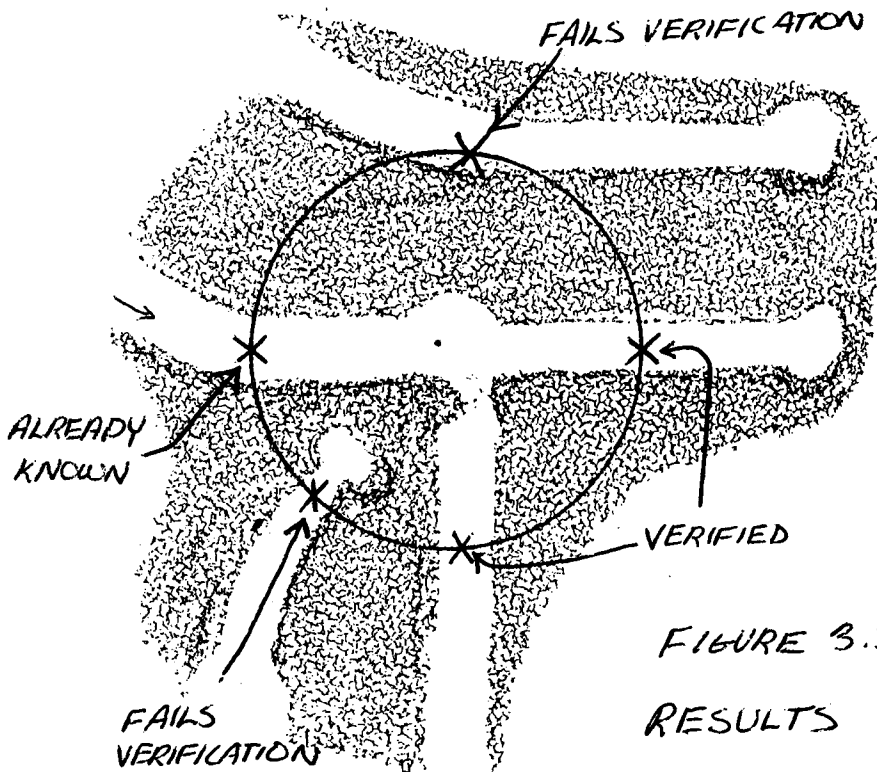


FIGURE 3.5  
RESULTS

closest to the center of the scan is returned.

If wires are found in three out of four attempts, **SAMPLE-TRACK** fits a line to their centers to determine the local direction of the segment. This direction, along with the wire's average width and brightness and the average brightness of the background around the wire is passed to the function **FOLLOW-SEGMENT**. If fewer than the requisite number are found, the suspect segment fails verification. Figure 3.5 shows the results of **TRACE-FROM-NODE** on the previous figure.

### 3.2 Tracking a Segment

My first attempt at tracking the segments was a procedure which applied the function **FIND-WIRE** along the hypothesized direction. This is the standard sort of "line following" technique used by Lerman <4> and Sharai <3>. Additions were made in the form of a servoing mechanism to follow the curves, and a step size (distance between scans) which varied with the local curvature of the segment. Tho this worked well, it left several problems:

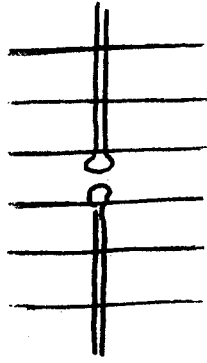
#### 1. Insuring continuity

The back of a circuit board has many wires which may run very close together. We want to be sure that we do not jump from one to another (figure 3.6a). This forces us to keep the separation between scan lines small.

#### 2. Detecting branches

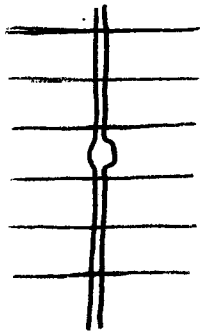
Unless a scan line falls exactly on the place where a circuit branches, that branch will be missed (figure 3.6b). To detect these branches, we must either keep the scan separation small or





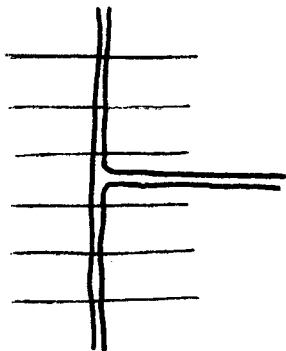
A.

SKIPPING



B

MISSED PAD



C.

MISSED BRANCH

FIGURE 3.6

scan along additional lines parallel to the segment.

### 3. Detecting pads

Pads are detected by noticing a "bulge" in the width of the wire. Pads may be missed unless we use closely set scan lines (figure 3.6c).

Tho keeping the scan separation to a minimum solves all three problems, it forces us to sample and examine a large number of points, many more than are necessary to merely follow the segment. For this reason, a different approach was developed. The basic idea is to scan along lines parallel to the direction of the segment. Points inside the wire and on either side are examined. The inside track insures that we are still on the wire and that it is continuous, while the side scans are used to servo the direction and propose pads and branches. Unless the picture is very bad, a simple threshold can be used to distinguish the conducting wire from the insulating background. This threshold is allowed to "float" so that it will be good for at least a local area of the picture. This procedure can be viewed as a special case of scanning along lines perpendicular to the segment - one in which we sample a minimum number of points.

### 3.3 Probing a Segment

In following a segment, the basic procedure is to "step" the function PROBE along the hypothesized direction of the wire. This function samples points within and on either side of the wire, thresholds them, and determines whether we should go forward, turn right

or left, or stop tracking. If PROBE's advice is to turn or go forward, a new point is calculated and the process continued. If PROBE suggests stopping, additional procedures are invoked which make a closer examination of the situation.

### The PROBE

Figure 3.7 shows one application of PROBE along a segment. The point C, the hypothesized center of the wire, is first sampled. If its intensity is below the threshold  $\epsilon$ PROBE-TH, 'STOP' is returned, indicating the end of the segment, a shadow, or noise. If it is above this threshold, then the two points L and R are sampled. The points are situated at a distance  $D_1$  from the point C in a direction perpendicular to the hypothesized segment direction. This distance is chosen to be 80% of the segment's typical width. If both these points are below the threshold, then we are roughly centered on the segment and PROBE returns 'FORWARD'. If both points are above the threshold, a pad is hypothesized, PROBE returning 'PAD'.

The other cases, in which one of the points is above the threshold and the other below, are ambiguous. We may simply have wandered from the center of the segment, or we may have reached a branch in the circuit. To differentiate these cases, an auxiliary point is sampled on the side which was above the threshold. This point is located a distance  $D_2$  from C, this distance being 1.75 times the width of the segment. If the intensity at this point is greater than the threshold, PROBE hypothesizes a branch toward that side (returning 'SUSPECT-LEFT', or -RIGHT'). If the point is below the threshold, then PROBE reports that we have wandered from the center of the wire, returning 'TURN-

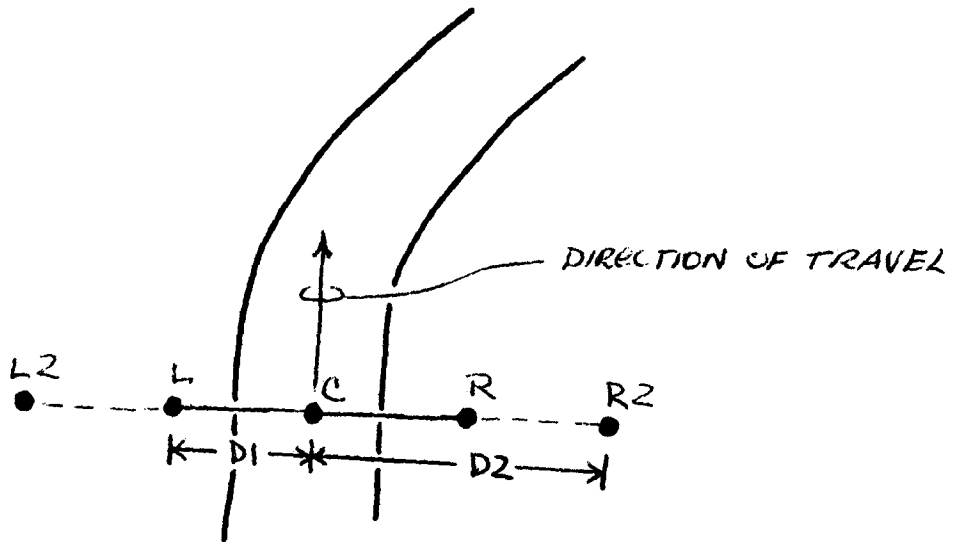


FIGURE 3.7

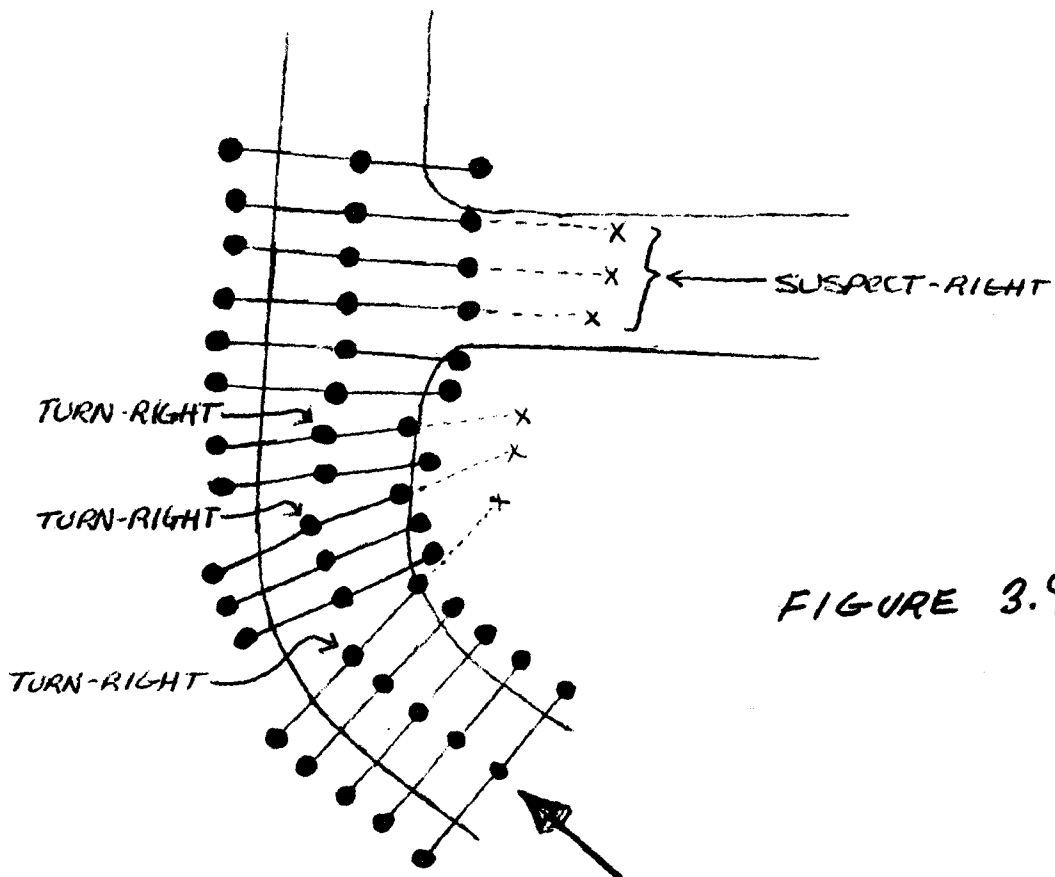


FIGURE 3.9

RIGHT' or 'TURN-LEFT'. Figure 3.8 shows a flow chart for PROBE.

It is important to sample these extra points only when the closer side scans suggest it. This is because the circuits may run close to one another, resulting in many false branch hypotheses if the outermost side scans were continually checked.

#### FOLLOW-SEGMENT

The function FOLLOW-SEGMENT does the work of guiding PROBE, stepping it along the wire (using the feedback to servo the direction), and adjusting the various parameters PROBE uses. FOLLOW-SEGMENT is initially given the following description of the segment (obtained from the call to SAMPLE-TRACK when the segment was verified):

1. An initial point on the wire.
2. An initial direction.
3. The wire's average width (W).
4. The average brightness of the wire (B1).
5. The average brightness of the background around the sampled part of the wire (B2).

From these values, various parameters are calculated for the tracking process. The major parameters are:

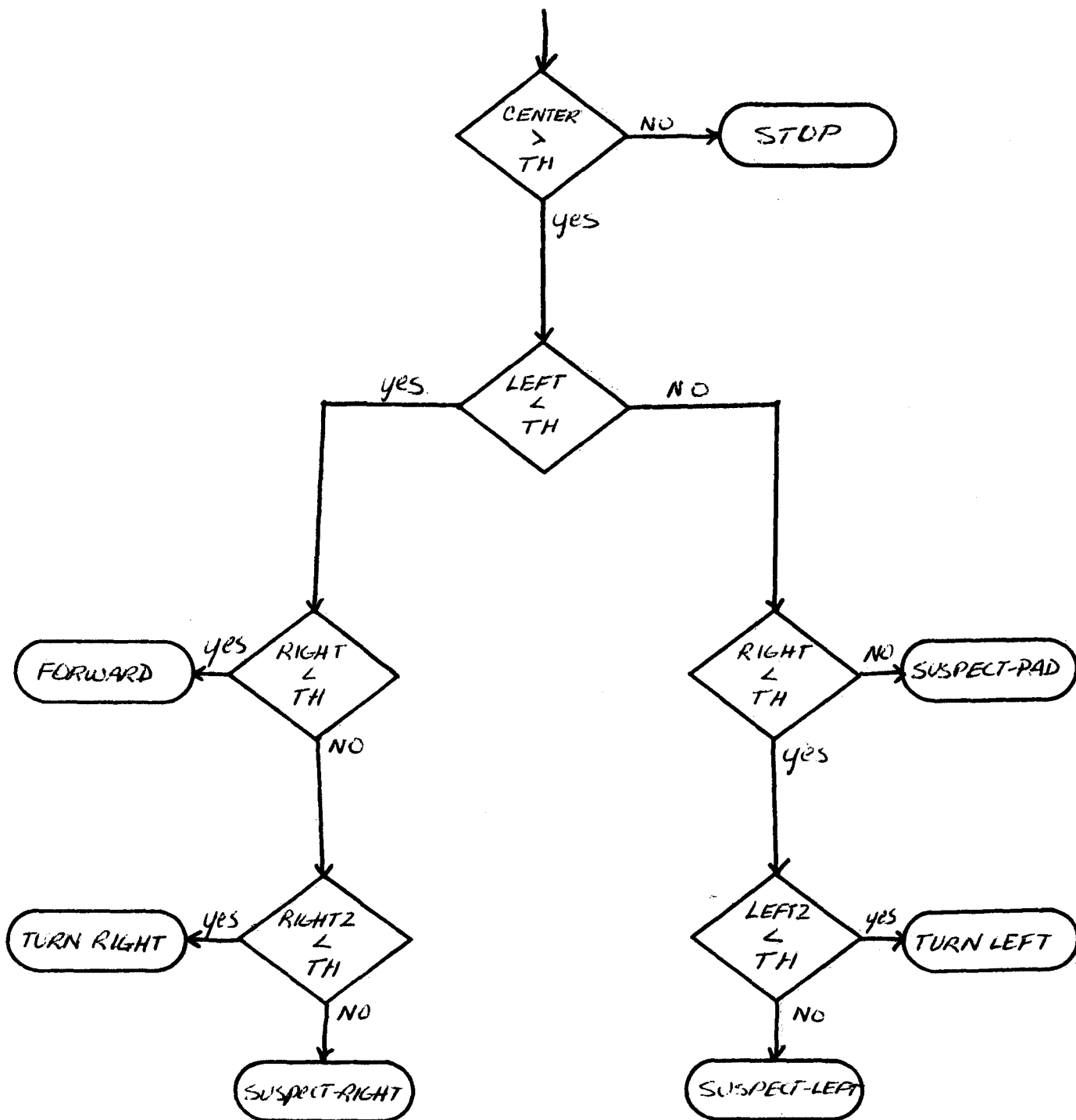
$$\bullet \text{PROBE-TH} \leftarrow B2 + 0.4*(B1 - B2)$$

$$\bullet \text{PROBE-STRIDE} \leftarrow 0.33*W$$

$$\bullet \text{PROBE-SIDE-STEP} \leftarrow 0.25*W$$

Once these have been calculated, Probe is applied to the initial point. If a node or the end of the segment is not proposed, a new point is calculated and the process repeated. The coordinates of this new point depend on PROBE's return as follows:

FIGURE 3.8



**1. FORWARD**

Step @PROBE-STEP units forward.

**2. TURN-RIGHT**

Decrement the current direction by @PROBE-ANGLE-INCREMENT , step towards the left (perpendicular to the direction) @PROBE-SIDE-STEP units, and step forward in the new direction.

**3. TURN-LEFT**

Similar to above, but in the opposite direction.

If PROBE returns SUSPECT-RIGHT, SUSPECT-LEFT or PAD, FOLLOW-SEGMENT defers to HYPOTHESIZE-NODE. This function continues to step PROBE along the current direction until the nature of the segment changes. It then creates a hypothesized node midway between the first and last suspect reported by PROBE.

If PROBE reports the end of a segment, a closer look must be taken. A number of situations might cause the hypothesis of the end of a segment besides its actual termination. We may have run into a shadow near a node, some noise, or a corner too sharp to navigate with our simple curve following mechanism. It might also be the case that the threshold (@PROBE-TH) we originally calculated is no longer suitable to detect the wire.

FOLLOW-SEGMENT attempts to find a continuation of the wire calling SAMPLE-TRACK to take a more detailed look. As described before, this function applies FIND-WIRE to points along the hypothesized direction of the segment. If wires are detected, the description of these wires is used to recalculate the "probing" parameters. The process is then resumed with the new initial point and direction. If no wires are

found, the end of the segment is hypothesized, and control returned to TRACE-SEGMENT. Figure 3.9 attempts to give a graphical description of this process, showing PROBE rounding a curve and stopping at a branch.



## IV. RESULTS AND CONCLUSIONS

### 4.1 Results

These programs, together with an additional module TRACE-FROM-PINS were tested on several printed circuit boards with good results. A common type of circuit board has most of its circuits terminating at a strip of pins on one side of the board. TRACE-FROM-PINS, is given the endpoints of a line which crosses this strip and scans along this line. Whenever it discovers a wire, TRACE-CIRCUIT is applied to this point.

Figure 4.1 shows the results of this process applied to a typical board. TRACE-FROM-PINS was given the coordinates of the points A and B and correctly found all of the pins. Note that some of the pins are no more than solder blobs, with no circuits attached. The simple mechanism for following segments worked surprisingly well, all of the segments being properly tracked.

The programs are currently somewhat liberal in hypothesizing nodes. Pads are sometimes falsely proposed at points of large local curvature on a segment. This could easily be fixed by the addition of a pad verifier which takes a finer look when a pad is proposed. In certain situations, pads can also be missed. In these cases, two pads are so close together that they appear as one large "blob", and only one is proposed.

### 4.2 Problems and Future Work

One problem not addressed is that of forming a complete representation of the board. The difficulty, particularly with a

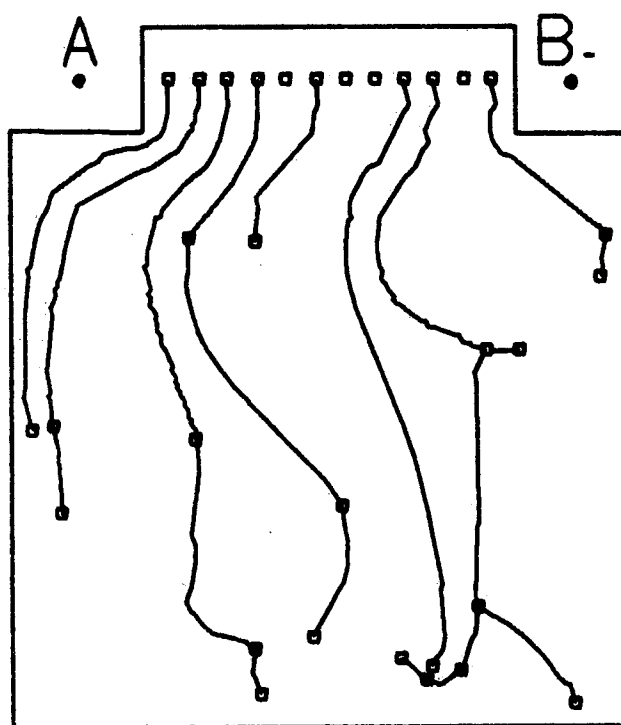


FIGURE 4.1

tracking type strategy, lies in insuring that all the circuits have been discovered and none missed. An allied problem is recognizing an already explored circuit when we stumble upon it in another place. To solve these problems, the entire scene could be divided into an array of regions, and some sort of multi-entry bucket coding procedure <6> used to keep track of known segments and unexplored suspects.

Pad specialists should be written which closely examine a node once it has been proposed. This is necessary to differentiate branches which are also pads from those that are not. Similarly a closer look at merged pads might detect evidence of two leads, or a "dumb-bell" shape.

Work on a pad inspector can also be done. Several different kinds of bad solder joints are easily detected by the trained human eye. Dewetted or over-heated solder connections, for example, appear to have a rough, spotty texture instead of a smooth, glossy one. Insufficient solder can be recognized by detecting a depression or pit in the solder above the lead hole.

Some additional higher level knowledge will also be necessary for some applications. For example, the nature of some components on the front of the board can be guessed by the pattern of pads on the back (e.g. transistors, ICs). This would be very useful if we are using the back of the circuit board to propose locations of components on the front.

## REFERENCES

- <1> Ejiri, M. , et al.  
A Process for Detecting Defects in Complicated Patterns
- <2> Horn, B.K.P. and Binford, T.  
The Binford-Horn Linefinder  
A.I. Memo 285 ; July 1971
- <3> Sharai, Y.  
A Heterarchical Program for Recognition of Polyhedra  
A.I. Memo 263 ; June 1972
- <4> Lerman, J. and Woodham, B.  
The Track Program Package  
Vision Flash 49 ; August 1973
- <5> Lozano-Perez, T.  
Finding Components on a Circuit Board  
Vision Flash 51; September 1973
- <6> Horn, B.K.P.  
VISMEM: a bag of "robotics formulas"  
Vision Flash 34; December 1972