FINDING COMPONENTS ON A CIRCUIT BOARD

VISION FLASH 51

by

Tomás Lozano-Pérez

September 1973

Massachusetts Institute of Technology

Artificial Intelligence Laboratory

Abstract

This paper describes a set of programs written in LISP that recognize resistors on circuit boards. The approach leans heavily on a thorough examination of the features found in representative intensity arrays and on representing the important points procedurally. The programs attempt to exploit evidence as it is gathered. The issues of hypothesis formation and change are considered. This paper represents a continuation of research described in a S. B. thesis of the same title submitted at M.I.T. on June, 1973.

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.

SECTION 1: Introduction

An attempt is currently under way at this lab to develop and test a complete robotics system in an environment of assembly, inspection and repair of discrete electronic circuits.(1,2) This effort has at least three closely related goals:

1. The development of a compact and inexpensive 'mini-robot' laboratory which will encourage participation in robotics research by other universities and industrial laboratories.
2. To test the mini-robot's competence as a laboratory tool.
3. To test the relevance of such machines in industrial applications.

The particular applications area chosen for initial attention is that of electronic assembly and repair. This application area, though well constrained, offers a rich variety of problems. These problems bear directly on the theoretical work which the laboratory has been pursuing for the past several years.

This paper presents some preliminary work done towards developing a vision system for the application. In particular, a set of programs has been implemented to locate resistors on a circuit board. This problem is simplified by the fact that components are limited to fairly simple shapes and are arranged flush on a board. The latter restriction limits the three dimensional interaction between objects. These characteristics make the necessary extension of visual skills from blocks to electronic components less painful. As will be seen later this extension requires a somewhat different approach, but many of the basic ideas remain valid in spirit if not in form. Moreover a circuit board presents a good micro-world intermediate, in some aspects, between blocks and the 'real world'. Large amounts of knowledge can be brought to bear on the recognition of components and thus the problem of organizing this knowledge is central. The issues of color, gloss, texture and curved objects can also be addressed in this domain.

The next section describes what things look like to our image input device. This is followed by a section describing my approach in general terms, followed in turn by a detailed description of the algorithms now implemented.

SECTION 2: The Circuit Board

The goal of the work reported here was to write a set of programs that would recognize resistors. It is hoped that the techniques used to accomplish this are applicable to other types of components. This chapter will deal with the visual characteristics of the board and the components on it. We will try to find some useful heuristics to recognize components by means of their intensity profiles along their crossections.

2.1 The Experimental Setup

An image dissector camera was used to obtain the intensity data on which the programs operate. Detailed information on the construction, operation and performance of this device can be found in (3). Only top views of the circuit board, with the light source near the camera are considered in this paper.

The Lab's facility for storing these pictures was used to ease debugging. Another invaluable debugging aid was a program to output the stored pictures on the line printer.

2.2 The Components

The purpose of this section is to show what typical components of circuit boards look like through the vidissector.

Resistors

1. Figure 2.1 shows the intensity profile through the dark

part of a resistor. Points labelled B correspond to board material. Points labelled R correspond to the dark material.

2. Resistors often have a highlight along the center. This is due to specular reflection along the line of maximum curvature with respect to the light source. The matting spray used to reduce the intensity of these highlights and thus protect the vidissector also tends to spread them out. Figure 2.2 shows a glossy resistor. The point labelled G indicates the gloss. B's indicate the board and R's the dark resistor masterial.

3. Figure 2.3 shows a typical color code band. Notice the bands become quite dark near the edges of the resistor. This is because the intensity of the reflected light is roughly proportional to the cosine of the angle between the incident angle (which in this case happens to equal the viewing angle) and the normal to the surface (see (4) for a discussion of this). B's indicate board brightness.

4. Leads are at least as bright as bands but they don't become as dark at the sides. Figure 2.4 shows the profile of a lead. The L indicates the lead, B's indicate board brightness and R's indicate resistor brightness. Notice that the width (the number of points between points equal to board intensity) of the peak is very close to the width of the resistor, making it difficult to identify a lead by its width. Instead the fact that it does not become quite as dark at the edges must be used.

5. Figure 2.5 shows the crossectional intensities along the

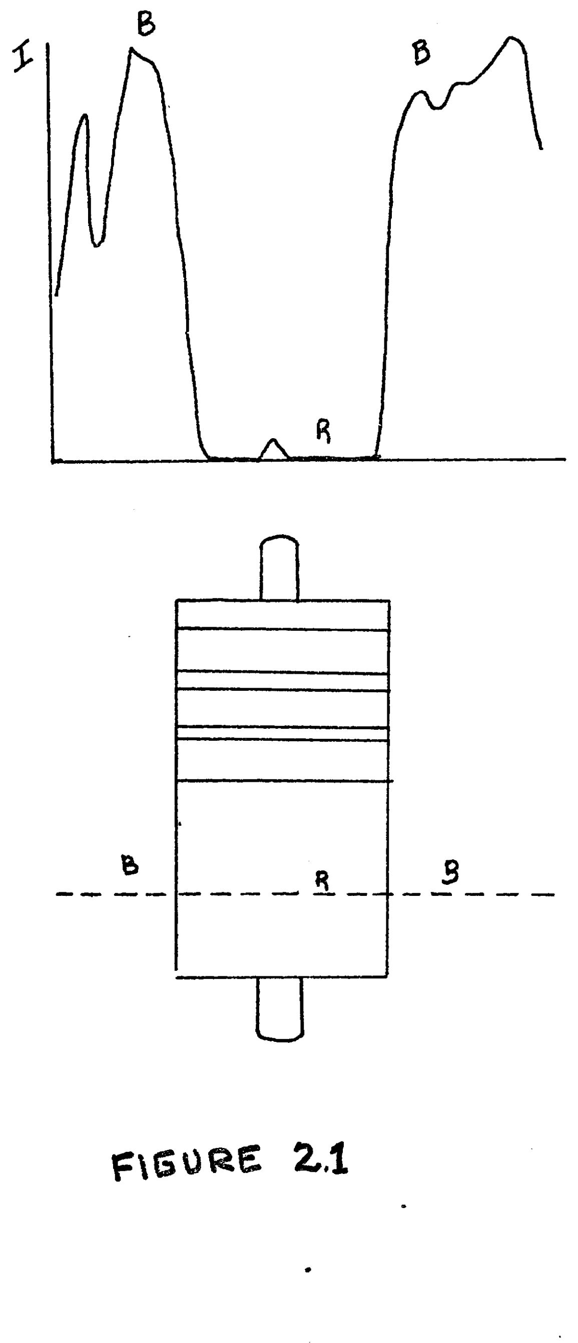

FIGURE 2.1

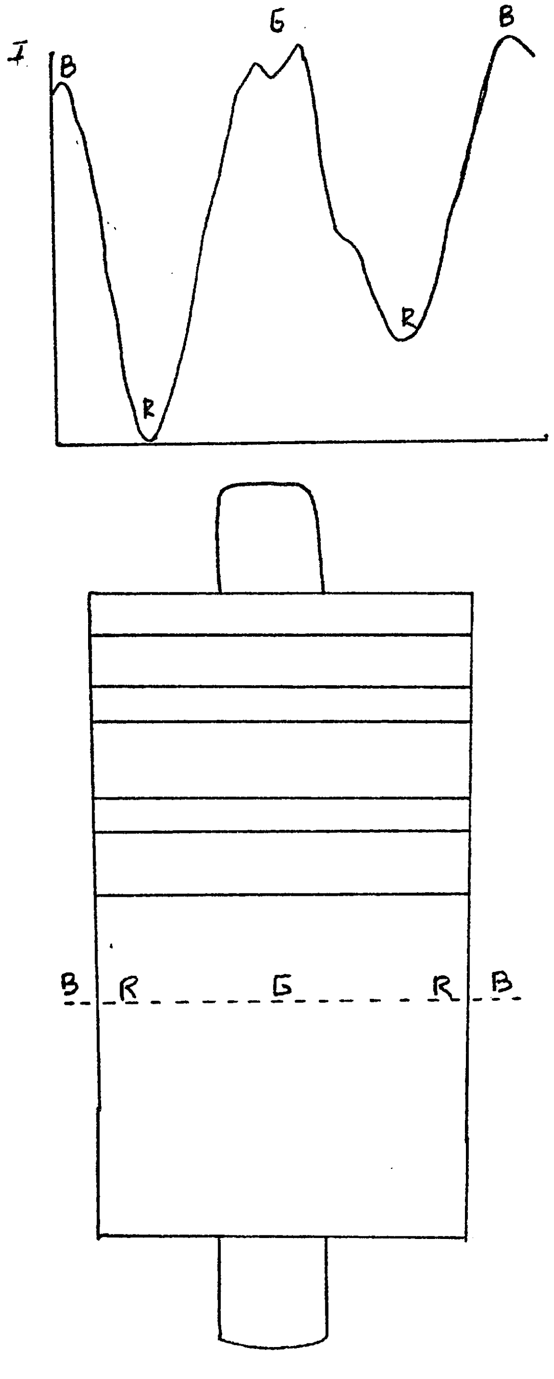

FIGURE 2.2

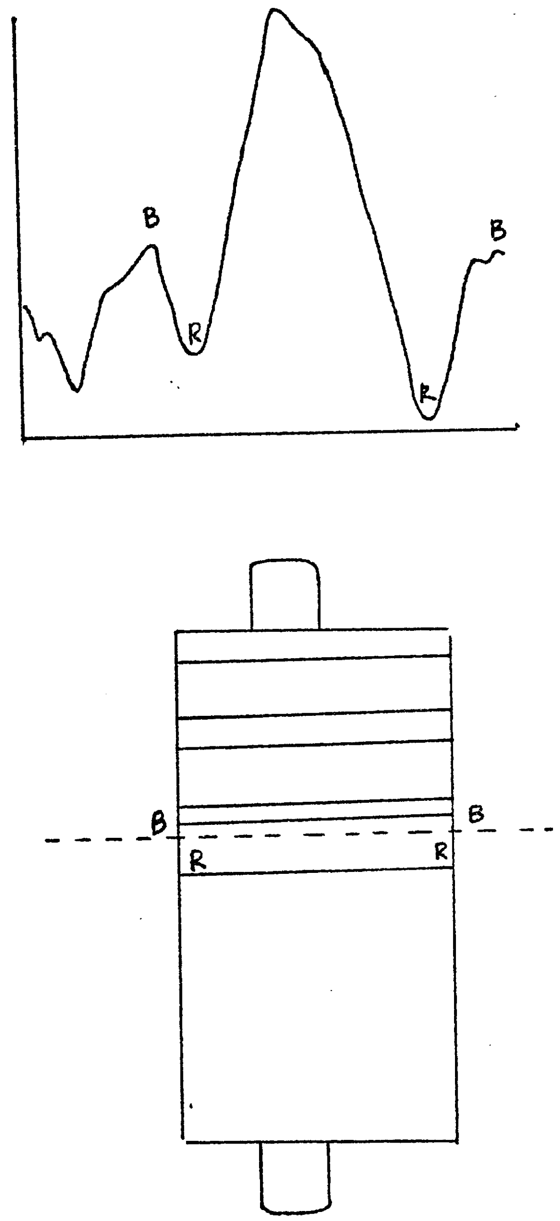

FIGURE 23

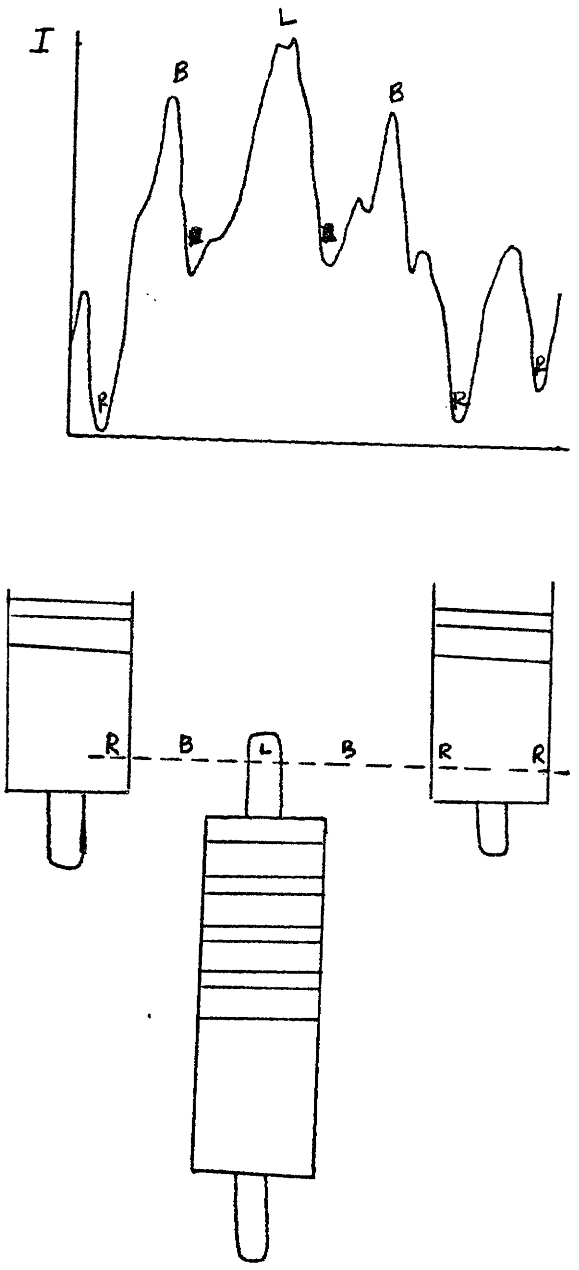

FIGURE 24

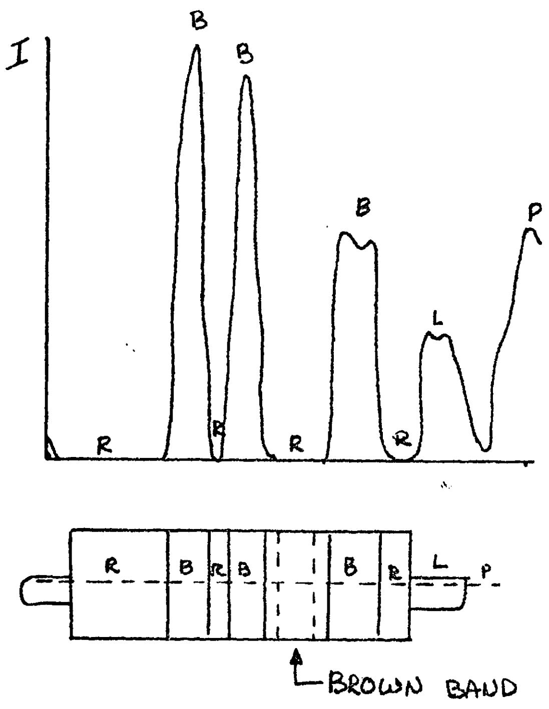

FIGURE 2.5

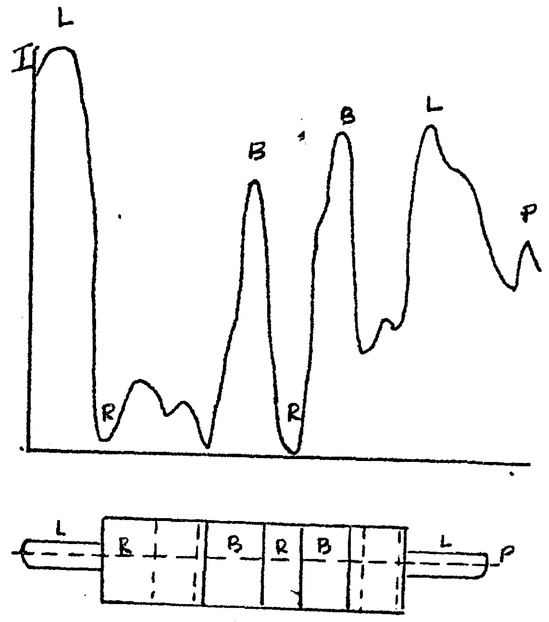

FIGURE 2.6

length of an glossless resistor. B's indicate bands, R's indicate dark resistor material, L indicates the lead and P's indicate board intensity. Figure 2.6 shows a glossy resistor.

Dark Ceramic Capacitors

A rough drawing of these capacitors is shown in figure 2.7. They appear as an almost rectangular region with a dark outline and a bright highlight along the center. The variation in intensity along this highlight is quite large due to surface irregularities. Figure 2.8 shows a crossection across such a capacitor. B's indicate board material, O's indicate the dark outline and the H indicates the highlight. Figure 2.9 shows a profile along the capacitor, the labellings are the same.

Metal Case Transistors

The key feature of these components is a very bright circular region. Figures 2.10 and 2.11 show orthogonal crossections across the top of a transistor. The gradual slope of the intensity profile in Figure 2.10 is caused by the angle of the reflecting surface with respect to the light source and the vidissector.

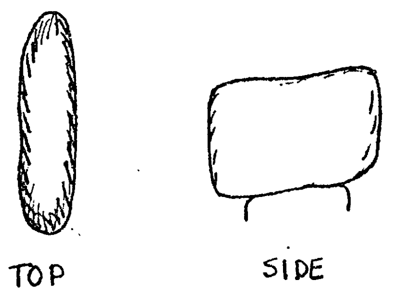

FIGURE 2.7

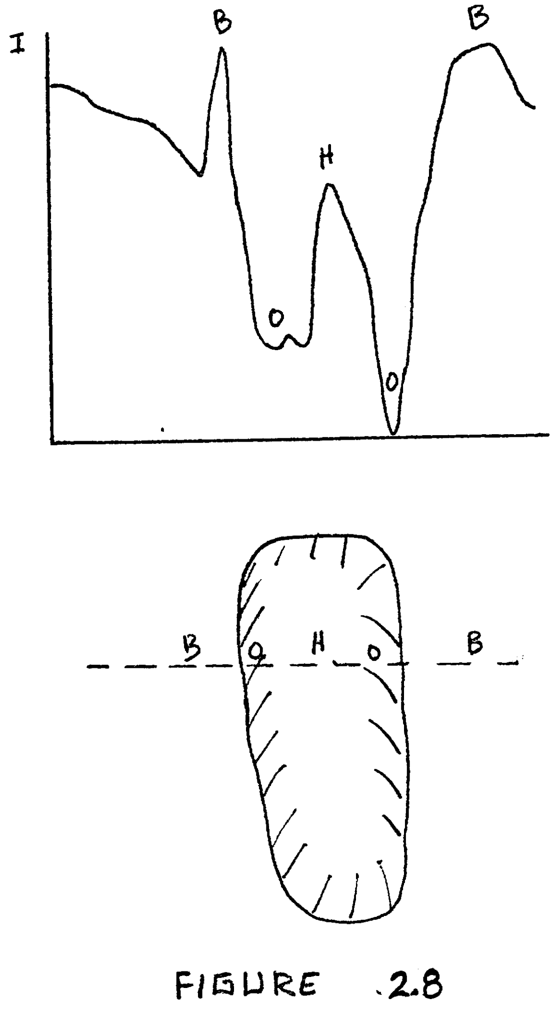

FIGURE 2.8

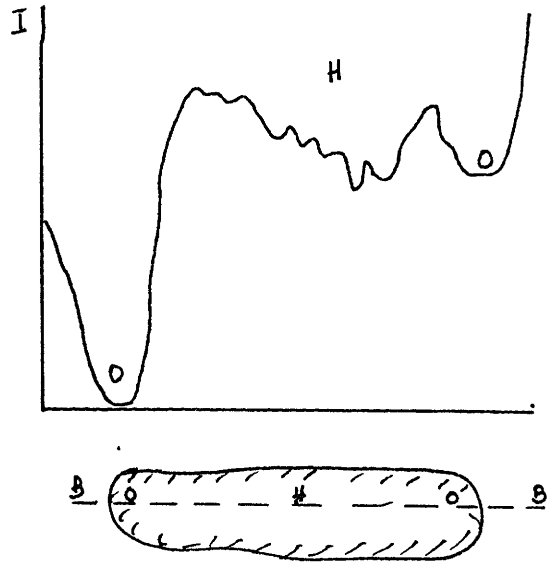

FIGURE 2.9

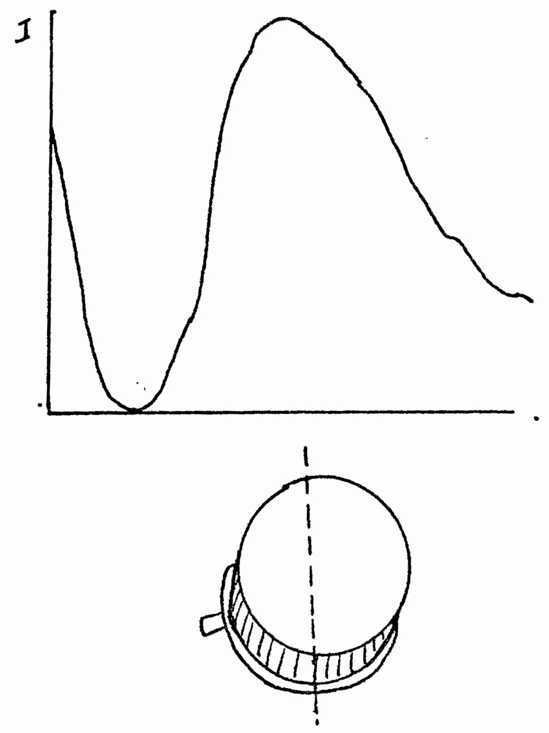

FIGURE 2.10

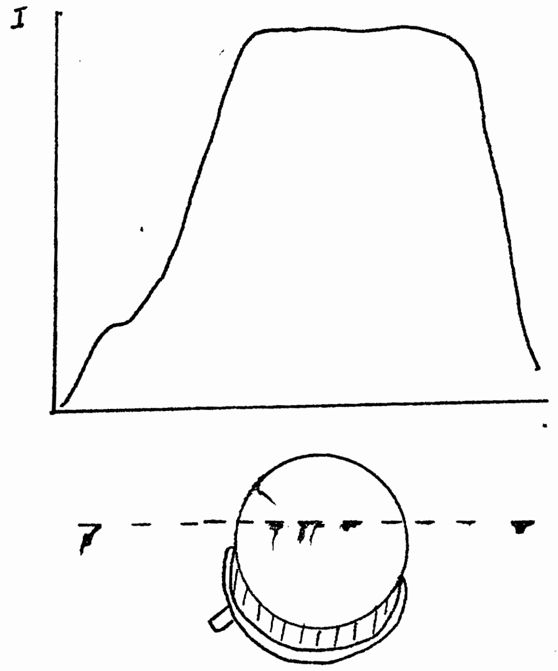

FIGURE 2.11

SECTION 3: Strategies and Approaches

The central problem treated in this paper is that of finding and identifying the components on a circuit board. Our research has been carried out mainly in terms of resistors. However the methods are believed to be applicable to other types of components.

We will consider several alternate approaches to this problem in order to relate the present work with other work in vision and justify the approach taken. These approaches are:

1. line finding
2. depth
3. color
4. texture
5. special markings

1. The identification of components on a circuit board could conceivably be carried out by first obtaining a line drawing of the circuit board and working from there. The drawbacks to this approach are best understood in the context of the development of the approach and its related techniques. So first a little overview.

Line finding and line drawing analysis have been the mainstays of research in computer vision since its inception. Fairly succesful line oriented systems have been developed to deal with the blocks mini-world (5). Until recently most of these systems were limited by what has been called the Hour-Glass problem (6). The problem is that higher level programs never communicated much with the line finders. Thus the whole system depended on the accuracy of the line finders. The advent of the heterarchical approach (7) has motivated at least two fairly succesful attempts (5,8) to guide line finders by some higher level knowledge of the perceived world.

The need for heterarchy points out what is perhaps at the heart of most perceptual problems in general and line finding in particular. This basic problem is the ′global-local paradox′. Global features are composed of local features but the whole, in turn, influences the interpretation of the parts. Heterarchy attempts to approach this problem by encouraging the shift in ′level′ and thus allow experimentation with alternate interpretations of local evidence as the global picture becomes clearer. It also allows guiding the process of obtaining local evidence by knowledge of global constraints. Heterarchically organized line finders use a model of the scene to determine what constitutes enough evidence for a ′semantically relevant′ line.

Scenes composed of light colored blocks against a dark background are ideally suited to line finding, not only because the blocks are fully described in terms of lines but also because there are few, if any, irrelevant lines or regions of unknown potential interest. If a region is above some threshold it is interesting otherwise it's background. On a circuit board this does not hold true. The 'predicate of interest' is more complex. Many more irrelevant lines exist and many areas cannot be ruled out from consideration by a simple thresholding technique. Not only do irrelevant lines abound but the lines that we would like to be relevant are not necessarily the most clearly defined. For example a color code band on a resistor looks more like an irregular ellipsoid than a band.

We should distinguish here between two line-oriented approches to object recognition. The first of these uses "undirected" line finding. By this I mean attempts to derive line drawings from the data without much recourse to knowledge of what is to be found in the scene. Some knowledge is almost always needed for any degree of competence, but in these line finders it usually takes the form of simple unvarying constraints such as demanding that all regions be closed and lines not be too short. The arguments against this approach are many but they have been discussed elsewhere (8, 9).

The heterarchical or "directed" approach attempts to use partial results together with an extensive set of heuristics to guide line finding. This approach is exemplified by Shirai's program to recognize polyhedra (8) and Wizard (10). These have proven quite successful for scenes composed of light colored blocks on a dark background. The use of line directed models is, on the other hand, probably not the right approach on a circuit board. A line model can only suggest where to look for more lines. The abundance of "fuzzy" and irrelevant lines tend to choke the line finder.

A very practical objection to line finding is that it is an inefficient method for hypothesizing many objects (except for blocks). Line finding is a computationally expensive operation and the usual global checks on the relevance of a line is the nature of its relationship to the other lines. This implies that several lines must be found and examined before any one can be dismissed as being irrelevant. On the other hand the dependence on line-shape oriented models means that several 'relevant' lines are needed to hypoyhesize an object. Thus if irrelevant lines abound constructing a hypothesis might prove very expensive. One way to interpret what has been called the 'Waltz effect' (11) is that many weak constraints serve to characterize objects more efficiently than a few strong constraints. My feeling is that lines are too strong (and expensive) a constraint to be an efficient hypothesizer.

This is getting at what I believe to be one of the basic issues in a heterarchical system — that of when to commit resources to verifying a hypothesis as opposed to obtaining more general information to make a better decision. It should be fairly clear that this depends on the particular mini-world we deal with. In the blocks world it did not make sense to quickly hypothesize object partition until a line drawing was constructed. This is true because partitioning depends on subtle interactions between lines. Witness the many ambiguous block scenes that have become part of A. I. folklore. The key issue is how quickly one can reduce the size of the search space. The proper time to hypothesize is when there are few enough choices left so that the process of confirming one hypothesis will give us most of the information needed to make the right choice. If you are 'close' enough the process of confirmation will either succeed or be able to point you in the right direction. Even a failed hypothesis is useful because it implicitely selects a set of features to 'notice'. One cannot afford to apply all known predicates everywhere. A hypothesis suggests which might be useful.

In a small world such as circuit boards where components are very different from each other we want to be able to pay attention to those simple features that cut down the search space quickly and then make a hypothesis early in the game. Lines are obviously the wrong choice for this role of pruning the search space because they are expensive and carry very little global information. This might be a lesson in how to deal with the 'real world'. Homogenecus models such as line drawings are bound to be less adequate for some domains than for others. One should be ready to use many types of information and knowledge at different points in the search process depending on the domain and the size of the space.

2. The availability of depth information might allow us to identify components fairly accurately from knowledge of their characteristic three dimensional shapes. This information could be obtained via direct range finding (lasers, etc.), stereo ranging or monocular depth computations. Although none of these techniques have, at present, been developed to the point where they could be reliably used in a practical application, it is hoped that they will soon be available.

Given that one could obtain a depth map of a circuit board, recognition is by no means assured. In a general scene a depth map presents most of the same problems as an intensity map such as occlussion and the body partitioning problem. In the circuit board domain, the planar arrangement of the components on the board would simplify this process. One basic problem still remains, the shapes of many components are similar (resistors, diodes , electrolytic capacitors, etc.). This necessitates the use of additional information to distinguish between them. The approach is a promissing one but the lack of appropriate hardware and techniques prevented us from considering it for immediate use. As more work is done in perfecting this approach, more attention should be given to it.

3. Color is a powerful cue to region aggregation, as such it could serve as a basis for an object recognition algorithm.

But since electronic components are not fully characterized by their color patterns recognition reqires other cues. Furthermore the lack of a sound theoretical framework and many hardware problems have hindered the development of a good color labelling program. Further use of color must await this developmemt.

4. One of the most pervasive differences between objects is texture. Conceivably texture differences or characteristic textures could be used as a means of component identification. Exactly the same problems exist here as in depth and color, technical difficulties in obtaining textural information and problems of grouping once it had been obtained. Very few theories of 'real' texture perception have been advanced and these are not useful for practical implementations.

5. All sorts of special marking techniques are always proposed in connection with industrial applications of computer vision. The answer to these schemes is usually the same: markings simplify at the expense of flexibility. If one were to require resistors marked with fluorescent paint then few applications could afford them. What if one wanted to shift attention from finding resistors to finding apples? Luminescent apples?

One general theme should have become apparent in the discussion of possible methods. All the methods can provide useful information but what remains the key problem is developing a framework where this information can interact to the best advantage. As was stated earlier several methods should be used in providing information. Methods with the highest yield of global information in the least computation should be used to construct hypotheses. The other methods should be used where possible to confirm and verify these hypotheses. One of the key issues raised in this paper is when and how to construct hypotheses. Technical constraints at the present limit us to the information available through a standard imaging device such as a vidissector or T.V. camera. In principle the use of any set of cues is desirable.

The approach chosen to carry out the identification is a simple one. It consists in using the typical intensity profiles of components as a hypothesis which is then confirmed by using boundary information such as an outline. The next section discusses the approach in more detail.

3.1 The Approach

Insofar as an object can be characterized by the general form of the intensity profile along orthogonal crossections this profile can be used to identify the object. In practice only a few features of the intensity profile can be said to be characteristic or invariant. For this reason the straightforward template match idea, as usual, fails to be a good approach.

Usually the features found in the profile are not enough to recognize an object with any degree of accuracy but serve well to formulate an initial hypothesis. The outline characteristics of the hypothesized object are usually sufficient as a means of verifying the hypothesis.

On a circuit board where components have simple shapes and litle variation in appearance within a class, the crossectional characterization of components is a fairly useful tool. Chapter 2 used these profiles to describe the visual characteristics of some of the most common electronic components. Very little confirmation, in the form of appropriate boundary line detection, is ususally needed for accurate identification.

On the other hand, it is fairly clear that intensity profiles are hardly adequate for describing blocks. The key issue here seems to be the importance of shape in describing the objects of interest. Electrical components have many characteristic visual properties other than outline (such as color bands). Blocks have almost none. This lack of features is precisely what makes blocks so amenable to line finding approaches. Objects with rapid variations in the intensity profile but simple shapes are ideal for profile characterization. Electronic components have both these properties.

The use of typical intensity profiles is powerful when it provides enough global evidence to facilitate the acquisition of supporting local evidence. A case in point is that of obtaining supporting shape evidence once a profile is found. A crossection only samples the outline at two points. Thus it provides absolutely no shape information but serves as a guide in deciding where to look for the outline. In this sense the "above threshold —> it's a block" predicate of line finders is a case of using a block's typical profile to suggest their presence. This, of course, is the trivial case.

The use of profiles has several interesting side effects:

1. Because of the additional evidence available through the profile, line dependent information, such as outlines, can be obtained via line verifiers, as opposed to the proposer-verifier pair. This allows more flexibility in considering evidence and has drastic effects on execution time. The most expensive phase of the line finding process is that of suggesting lines. This is due to the fact that a full blown line proposer must examine most significant discontinuities in the intensity array for possible line candidates, thus wasting quite a bit of effort. The constraints imposed by the profile and other higher level features severely limit the places where line candidates should be sought. This limitation of the search space is one of the key advantages of heterarchy.

2. Intensity profiles also provide cues of how to set parameters in the line verification phase. The additional

evidence thus serves to trade between sensitivity and noise immunity more intelligently.

3. The predicative, as opposed to constructive, nature of the approach allows the programs to ignore portions of the scene which are not 'understood'. The typical line drawing approach is to draw the complete scene and then examine the drawing. This method wastes much effort in finding drawings of regions which might easily be found to be irrelevant via their profile. Even a 'low level filter' approach such as searching for a particular pattern of lines, gets bogged down because of the many 'false' lines present. It is interesting to note that this last point is only a "feature" (as opposed to a "bug") in this particular application environment. In a general vision system one would want the ability to apply some general knowledge to all regions of possible interest. In fact the system to be presented here tries to get away with using almost no 'general knowledge'. One can always add a 'general interest predicate' that claims regions that no special purpose predicate claims and supplies some more useful description of the features present. This seems to be the wrong way to go about it in general, but in such a constrained mini-world it might prove adequate (if it proves needed).

The use of typical intensity profiles as a hypothesis to guide line verification has some advantages in the circuit board domain. It also has some glaring theoretical drawbacks. The major assumption, and consequent weakness, of the approach is that constant viewpoint and lighting conditions are maintained. Intensity profile characteristics change drastically, for most objects, with changes of viewpoint. Under typical assembly line conditions both viewpoint and lighting are likely to remain fixed. It seems neverthless desirable to be able to predict these changes and thus allow for their presence. This is a fruitful area for further work.

3.2 Sketch of a System

A vision system for the electronics application could be designed around the approach introduced in the last section. This section is merely meant to suggest the style of such a system.

The system is based on a set of predicates for the recognition of electronic components. The predicates are a set of programs that can, hopefully, find all of the components of a specific type in a specified area of a circuit board. Associated with each predicate is a hypothesizer for suggesting the presence of the component. The hypothesizer is called either when a particular set of features is discovered or when another hypothesizer or predicate has failed in a particular fashion. This is the only part of the system that depends on what other knowledge the system contains.

Evaluation of the goal to find all the resistors and capacitors on the present board would initiate the following process:

1. A region of possible interest is found. (This region is determined in terms of the profiles of the components of interest. In this case it would be a dark glossy region which could be part of either a resistor or a ceramic capacitor.) The amount of evidence needed to conclude that a region is interesting varies considerably with the size of the universe and with the availability of computing facilities. The larger the universe the more features which should be examined before commiting the system to a verification process. In the "real world" quite a bit of general knowledge about shape, size, color, etc. should be used before a 'high level' hypothesis is formulated. Having only 5 or 10 objects to choose from, it is probably better to hypothesize early and use that to guide the search for confirming features. If, on the other hand, one had a committed processor that might be idle otherwise, (such as the mini-computer in the mini-robot) one might as well exploit it by obtaining more features.

2. Once a region has been found, one (any one) hypothesizer is applied to it. It checks for the occurrence of certain features, profiles, lines or whatever. Success is translated into a call to the associated predicate. Failure produces a report of the reason for failure. This report is then used to choose the next hypothesizer to be tried. Hypothesizers are assummed to be fairly simple so that this initial phase should not be too time

consuming.

3. The predicate then attempts to verify the existence of its associated component, and to obtain details which might prove useful elsewhere (such as to a manipulation or debugging program). Failure of a predicate has the same effect as failure of a hypothesizer except that the data which has been found should not be wasted. Thus the new choice of a hypothesizer must depend on what was found and an attempt must be carried out to salvage this data. Some data, such as outline, should be fairly easy to convert.

In actuality the simplicity of electronic components would allow the use of a simpler method. Each of the available predicates could be applied in sequence to the area of interest, thus obtaining the desired information. The method is inefficient because it leads to multiple examinations of the data. If more complex objects were being recognized the waste associated with erroneous hypothesis could not be tolerated.

SECTION 4 : Finding Resistors

This section presents a set of programs, currently implemented in LISP, that follow the approach described in the last section in recognizing resistors. The programs are given the coordinates of a section of a circuit board and they produce a data base containing a description of any resistors found there. A first version of a ceramic capacitor expert is currently being debugged. These programs attempt to address several issues:

1. The effectiveness of the use of typical intensity profiles as hypotheses,

2. The interaction and relative effectiveness of these profiles vs. line oriented data.

3. Organizational issues, such as how to construct and how to change hypotheses.

The overall structure of the programs is diagrammed in figure 4.1. Shown are the resistor hypothesizer, predicate and their supporting programs, including a proposer and a verifier for bands, ends, leads and lines. I will first present an overview of how the programs interact, and give a more detailed description of the programs in section 4.1.

Upon being called into a region, the hypothesizer scans the region horizontally and vertically in search of a line separating the bright board material from a dark region. If this were the

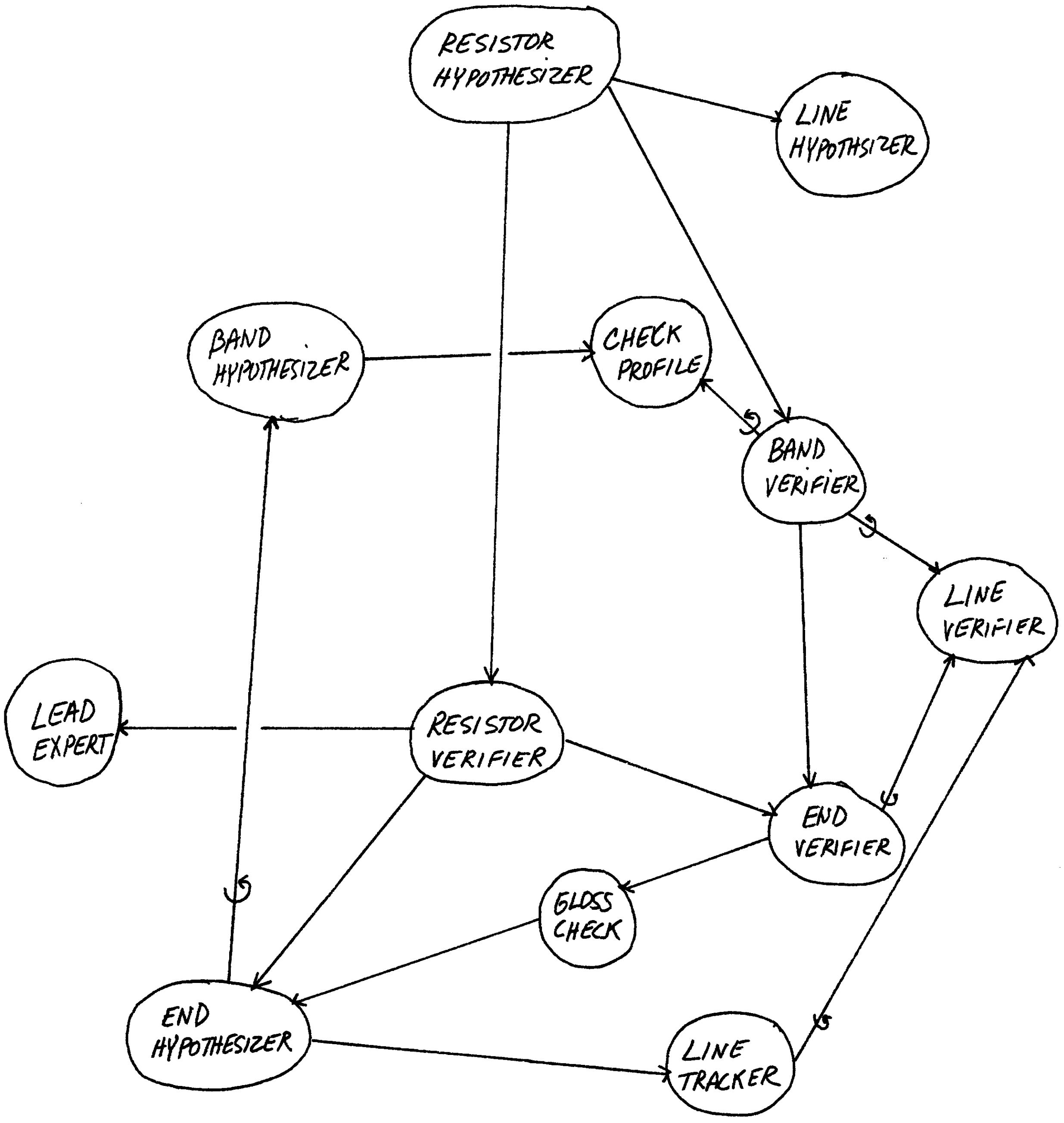
RESISTOR HYPOTHESIZER
LINE HYPOTHSIZER
BAND HYPOTHESIZER
CHECK PROFILE
BAND VERIFIER
LINE VERIFIER
LEAD EXPERT
RESISTOR VERIFIER
END VERIFIER
GLOSS CHECK
END HYPOTHESIZER
LINE TRACKER

FIGURE 4.1

side of a resistor a band should be near on either side of the point of entry. A search for this band is carried out parallel to the line. The hypothesizer calls the band proposer which looks for a profile that might belong to a band. If one is found then the band verifier is called. Provided that the band can be verified the resistor predicate is called, else the hypothesizer continues its search elsewhere for another resistor.

The predicate is thus called with both sides of the proposed resistor and a band known. It then starts off by calling the end proposer, which, in turn, calls the band proposer. Any bands found are stored in the data base marked as likely. When a bright region is found which fails as a band, the band proposer suggests that it might be an end. The end proposer then checks to see if it fulfills the characteristics of an end. If so, the end is proposed and an attempt is made to verify it. If it succeeds the predicate calls the lead proposer and verifier. If an end cannot be verified the predicate goes back to the nearest band which has been proposed, and if it can be verified, attempts to find the end by tracking the side of the resistor. If none of the bands can be verified the resistor hypothesis is flushed. If in fact the ends can be verified the resistor is fairly sure. An attempt is then made to find any bands the band proposer might have missed.

4.1 The Algorithm

1. Given a section of a circuit board the hypothesizer's first task is to isolate a region of possible interest. This is done by sampling the whole region very coarsely and choosing regions that are darker than 80% of the board and larger than some minimum area. This is based on the fact that the resistors and ceramic capacitors are among the darkest region of a circuit board.

2. The hypothesizer scans the region horizontally and vertically for an edge going from the bright background to darkness . The hypothesizer assumes that it is entering the resistor from the side into one of the dark regions of the resistor. <Figure 4.2> This assumption simplifies the programming at the cost of another scan of the region.

3. Once an edge has been found in 2. several scans are made parallel to the original one. If enough edges are found then a line equation is computed by a least squares fit <Figure 4.3>.

4. A positive edge is now sought perpendicular to the line found in 3. This edge is conjectured to be either the other side of the resistor or a gloss highlight. <Figure 4.4>

5. Now the band proposer is called and told to look for a band starting at a point half way between the edges found in 2. and 4. with a slope equal to that of the line found in 3. <Figure 4.5> The band proposer looks along this line for a sharp positive transition followed by a similar but negative

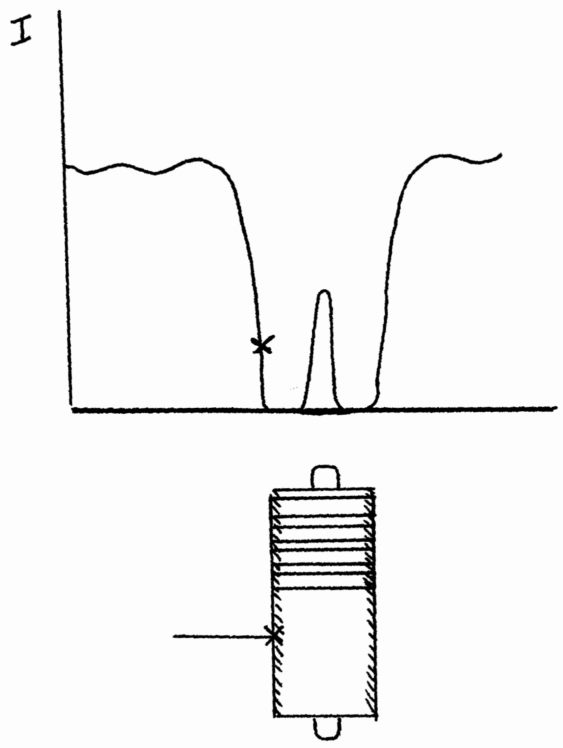

FIGURE 4.2

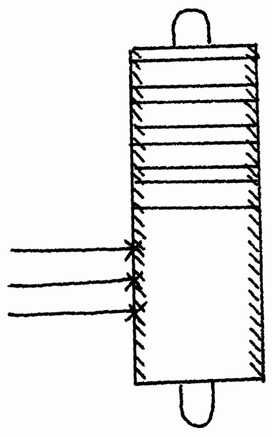

FIGURE 4.3

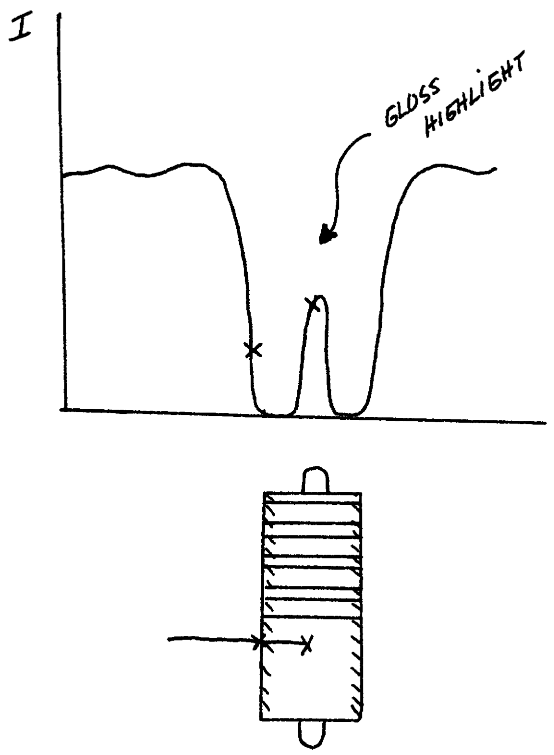

FIGURE 4.4

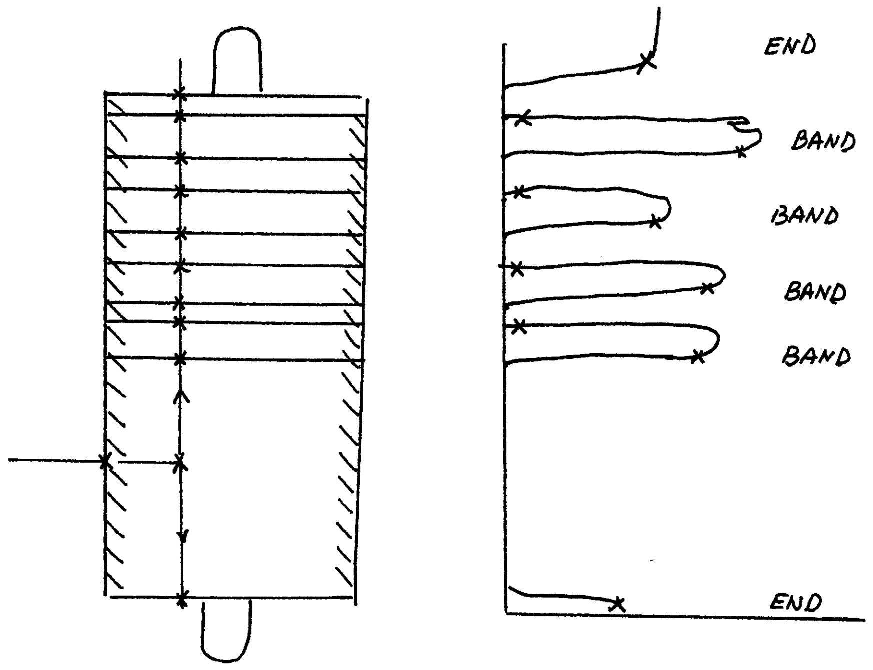

FIGURE 4.5

transition. At the top of the positive transition it looks perpendicularly to its direction for a negative transition. It is thus using the typical profile of a band (as shown in figure 2.3) to hypothesize its presence. The band must fulfill certain conditions in order to qualify. It must become dark enough at the sides of the resistor. This enforces the constraint on the typical profile of a cylinder (4, 12). Its width must also be within a bound set by previously discovered bands and by the known relationship between maximum band size and the width of the resistor.

6. The band verifier is then called. This function checks that the sides of the band coincide with those of the resistor. The verification of the sides of the band is used to update the equations of the lines representing the sides of the resistor. If, as in this case, the equation of one of the sides is not known the new equation is stored as the corresponding side of the resistor. It also checks the profile at several other crossections closer to the edges of the resistor. <Figure 4.6> The assumption is that a real band extends almost to the sides of the resistor because the intensity goes down gradually with curvature while noise peaks and glossy highlights are more restricted to a narrow band along the middle of the resistor. If both the profiles are found to be end profiles then the verifier fails and returns a message suggesting the end. If a mixed veredict (band and end profiles) is received the program lowers

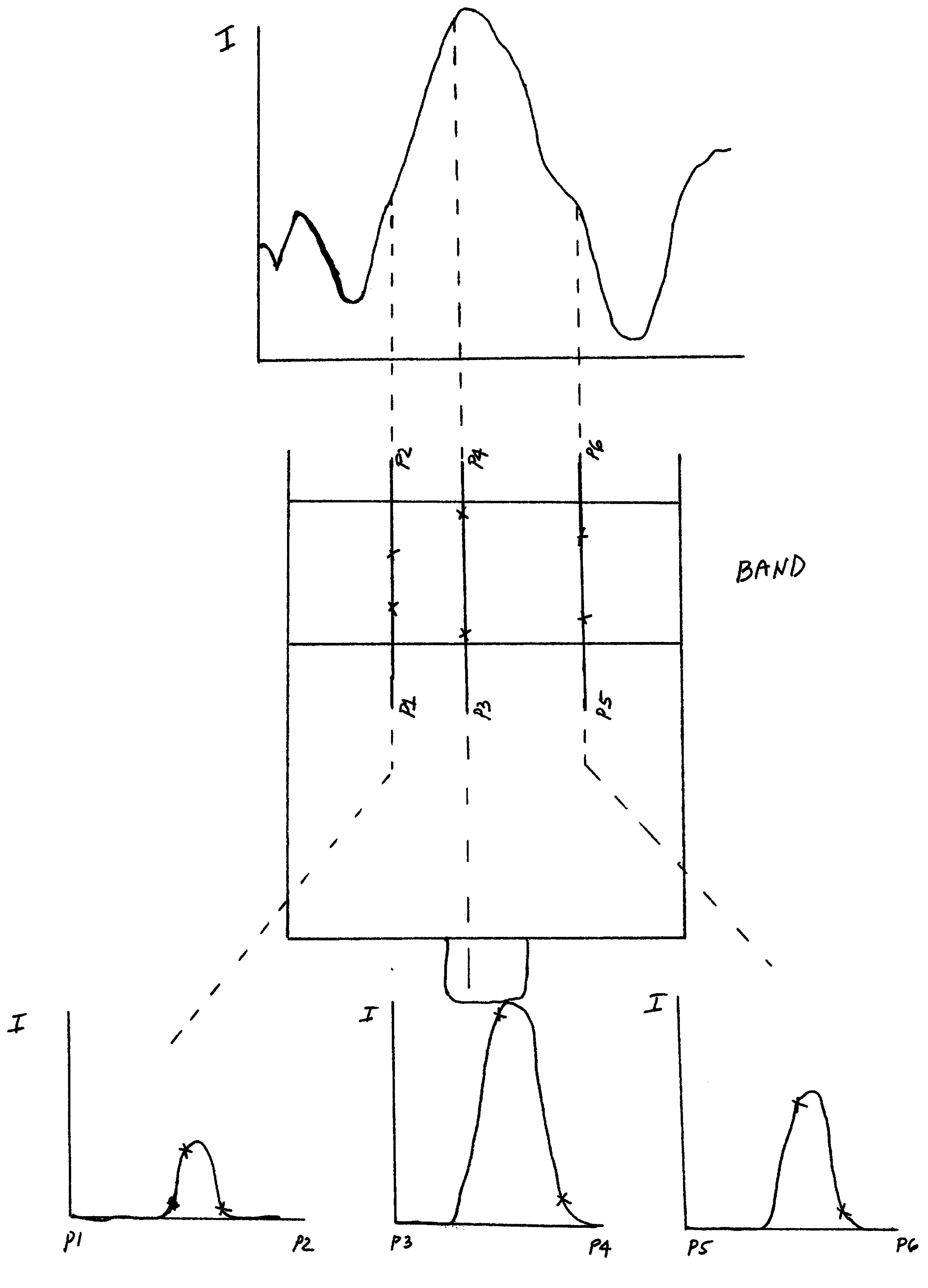

FIGURE 4.6

the threshold of acceptance in the band proposer and calls it again. If the end profile is now found to be a band profile the verifier succeeds. Otherwise it fails with no suggestions.

7. If the band verification fails then the same process is carried out going in the opposite direction, along the line. If no bands are proposed then the resistor hypothesizer declares failure. This would result in a call to the capacitor hypothesizer since a dark glossy object without sharp bands might be a ceramic capacitor.

8. If the band is verified then one knows where both sides of the resistor are (from steps 2 and 6) and the coordinates of one band. Now the end proposer is called. This program calls the band proposer. Starting from the known band it searches for a band in the area the band can be expected to be from observed typical interband spacings. If a band is found no attempt is made to verify it. The program then repeats the process starting from this most recent band. If no bright region is found the program sets a "careful" flag and keeps moving down the resistor. If a band is proposed while in 'careful' mode an attempt is made to verify it. Success of the verification procedure causes a "possible missing band" to be signalled between the two bands. This mechanism serves to help overcome the vidissector's inability to detect red, brown and green bands very well. If the band verification fails an end is hypothesized there.

9. On the other hand, if while looking for a band, a

bright region is found but with no corresponding transition to darkness within the maximum allowed width for the band, the band proposer fails. This causes an end to be proposed. The band program also fails when the sides of the proposed band are either not dark enough or else cannot be found at all.

The use of the band proposer to propose ends illustrates a general technique used to change hypotheses. A new hypothesis can be suggested either by the ocurrence of a particular feature or by the failing in a particular fashion (i.e. with a known message) of another hypothesis. Thus it is with ends and capacitors. An end is a failed band and a capacitor is a failed resistor. The process of failing is handled by a system function which keeps a data base of what to suggest upon failure of a particular routine. In some sense the entries in this data base are "similarity pointers" in the sense of Winston's "similarity nets" (13). This process not only allows transfer between different concepts but is an efficient way of storing procedural knowledge without unnecessary duplication.

10. An attempt is then made to verify the proposed end. This is done by verifying that the sides of the resistor continue up to that point but stop there. If the sides are found to continue beyond the expected end of the resistor the program tracks the lines to their end. If in the original end verification step no lines are found near the expected sides of the resistor an attempt is made to track the side of the resistor

from the closest proposed band which can be verified. If none of the bands can be verified the resistor predicate declares failure. Once an end is proposed by the tracking program an attempt is made to confirm it by looking for a sharp discontinuity in the gloss highlight at that point. This is very helpful in pinpointing shadowed ends. If the gloss check fails then the end proposer is called again. The profiles are then used to suggest a new location for the end. Failure of this operation causes failure of the current resistor hypothesis.

11. If the one end succeeds a lead proposer and a verifier are called. These look for a thin bright region midway along the end of the resistor. It expects a dark region where the lead goes into the board.

12. Knowing where one end of the resistor is, the end proposer duplicates steps 8, 9, 10, and 11 but in the opposite direction. It uses its knowledge of the average size of previous resistors to set a limit on its exploration.

13. Once both ends have been found, a check is made to see how many bands have been proposed. If more than four have been found then an attempt is made to verify them. If less than four then the resistor predicate checks to see if a "possible missing band" has been proposed. The band verifier is then called.

14. The predicate then attempts to check these proposed bands by running the band proposer with lowered thresholds. If no success is had in finding them, then the proposal is left in

the data base marked as tentative.

15. The predicate then updates any parameters such as average resistor length etc. by using information about the current resistor.

SECTION 5: Conclusions and Further Work

The programs described here have had a certain measure of success with scenes composed of noncverlapping components. I am currently experimenting with trading off outline vs. profile constraints so that less line verification is done. Line verification is one of the most time consuming operation the system performs.

An interesting result is that even though the edge finder and the line verifier are very sensitive to noise, the global information obtained from the profiles and the other constraints work together to produce fairly accurate results as well as speed.

There are implementation issues which seem to be of interest in understanding heterarchically organized systems. One of the key concerns of this work is the process of generating hypotheses and changing them. This implementation, being in LISP, does not have the explicit pattern directed features of Conniver or Planner. Instead the programs are "driven" by hypotheses. Hypotheses are data structures which "describe" a component or feature. It can be created and has defaults set automatically (Although at present little, if any, use is made of this information I plan to focus on this in later implementations). Most of the programs in the system accept a hypothesis and attempt to expand or verify it. In this way, by having all the information organized centrally it is easy for the program to check the "state" of the process and perform goal directed operations. It allows writing programs as experts and critics which can do things through the data base which the calling programs don't necessarily expect. The explicit use of hypotheses as structures makes for a certain neatness in the code as well as facilitating interaction between programs.

At present we hope to experiment with the interaction of Tim Finin's programs that examine wires on the the back of a circuit board and Mark Lavin's programs which given a path along a resistor determine the colors encountered. The interaction is envisioned in the following fashion:

Finin's programs would examine the back of a board for pads, where leads from the components come through the board. It would generate a list of these points and pass it to the programs discussed here. The hypothesizers would use areas around these points as regions of interest. Once a resistor has been found it is passed along to Lavin's program for color checking. If the colors are found to be appropriate they are stored otherwise corrective action is undertaken.

BIBLIOGRAPHY

(1) Minsky, M., "Mini-Robot Proposal to ARPA". A.I. Memo 251, 1972, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(2) Minsky, M., Papert, S., "Proposal to ARPA for Continuation of Micro-Automation Development". A.I. Memo 274, 1973, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(3) Horn, B.K.P., "The Image Dissector ′eyes′ ". A.I. Memo 178, 1969, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(4) Krakauer, L.J., Computer Analysis of Visual Properties of Curved Objects. A.I.-T.R.-234, 1971, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(5) Winston, P.H., "The MIT Robot". Machine Intelligence 7, 1972, Edinburgh University Press.

(6) Winston, P.H., Lerman, J., "Circular Scan". Vision Flash 23, 1971, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(7) Minsky, M., Papert, S., Progress Report. A.I. Memo 252, 1970, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(8) Shirai, Y., "A Heterarchical Program for Recognition of Polyhedra". A.I. Memo 263, 1972, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(9) Minsky, M. Papert, S., "Proposal to ARPA for Research on A. I. at M.I.T., 1970-1971". A.I. Memo 185, 1970, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(10) Winston, P., "WIZARD". Vision Flash (in development).

(11) Waltz, D., Generating Semantic Descriptions from Drawings of Scenes with Shadows. A.I.-T.R.-271, 1972, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(12) Horn, B.K.P., "Shape from Shading". A.I.-T.R.-79, 1970, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.

(13) Winston, P.H., Learning Structural Descriptions from Examples. A.I.-T.R.-231 , 1970, Artificial Intelligence Laboratory. Cambridge, Mass.:MIT.