VISUAL UNDERSTANDING OF HYBRID CIRCUITS VIA PROCEDURAL MODELS*

R. T. Chien and W. Snyder

Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urban*, Illinois 61801
U.S.A.

Abstract

A system of programs has been constructed to process the television image of a hybrid circuit* This system is capable of "inspecting" the circuit image regarding defects. To achieve this a procedural model of the circuit is developed. The computer "sees" the image and interprets what It sees with the procedures in the model. In this manner it is able to determine certain Important characteristics of the image in its view. It appears that this procedural approach can be useful to many domains of visual image processing and object recognition.

1. Introduction

One stage of the manufacturing process of hybrid circuits which is not automated is a visual inspection of these circuits, tills paper describes the development of a system to demonstrate automation of visual Inspection by using the techniques of computer vision, and in particular, the principle of procedural models.

Hybrid circuits are built on a rigid, insulating substrate, generally some ceramic material. Resistors and conductors are painted on using paints, which, when dry, have different resistivities. Transistors, capacitors, and integrated circuits are then mounted with either solder or glue. Finally, wires are attached which make the connection to some sort of holder-

In hybrid circuits factories, the visual inspection is generally performed by peering through microscopes. One major purpose of the human inspectors is quality control. It Is desirable to determine not only if a circuit Is defective, but why it is defective so that corrective measures may be taken. Another Important use for visual inspection is to detect defects in partially completed products so that the defective part may be rejected early in the assembly process, before It has become part of a much larger and consequently more expensive product.

The Inspectors check for missing, misplaced or misorlented parts, bad solder bonds, etc. This Inspection has not been automated previously because the manufacturing process Introduces numerous irregularities in normal circuits. For Instance, in the soldering process the solder rosin may flow in unpredictable ways, discoloring the substrate and producing extraneous edges.

*Research supported by the Joint Services Electronics Program (U.S. Army, U.S. Navy, U.S. Air Force) under Contract DAAB-07-72-C-0259.

The first objective of the development of the circuit inspection system was to provide a context in which to examine the principle of procedural models. The system provides this context in three different ways:

1) Inspection provides an application of the top-down approach to vision, and of the procedural models which incorporates this approach.

2) The problem of re-training the system for inspection of varied circuits provides an exercise in automatic generation of procedural models.

3) The fact that identification of certain circuit components may be dependent on the identity of neighboring components provides an instance to make use of the power of procedural models whereby one model may call another model for assistance.

Another objective was to develop a system which could run under "reasonable" assembly line conditions. Since hybrid circuit assembly lines have considerable accuracy in positioning, a certain amount of predictability was assumed in location of the circuit. It is also reasonable to assume a degree of control over direct lighting, but control over ambient light could not be assumed

2. Basic Characteristics of the System

The heart of the system is a procedural model. This model consists of the overall circuit model and several component models. The system understands the characteristics of each type of component via the component models; while it understands the topological and structural relationships via the circuit model.

The operator positions a circuit under the lens of the television camera and starts the system which takes five pictures: two grey-scale pictures with the lights turned at different angles, and three with color filters. These five pictures are stored on disk files for convenient later reference The operator then uses graphics to specify the circuit details.

The output of the initialization phase Is a model for the entire circuit. This model Includes a data structure and a control structure in combination which when compiled and loaded become the inspection phase of the system.

In the initialisation of the system another picture, representing the hue, is computed from the three color pictures and stored. Another plcure, the "masked difference" picture used in the wirefinding algorithm Is also computed and stored at this time.

Since wires can obscure other components, they must be Identified prior to any other compo-

nents. Thus, wires are located first and their locations remembered for the other component models to check.

3. Procedures as Models for Visual Objects

This section contains a discussion of some of the problems associated with the creation and use of models for visual objects and presents the concept of "procedural models." When using such models, recognition of an object becomes equivalent to executing the appropriate model. Instead of having a multiplicity of models for objects stored in a data base and some general purpose program for choosing and using these models, the models themselves are programs.

Procedural Models in the Circuit Inspection Problem

In the analysis and inspection of hybrid circuits, procedural models can become involved at three different levels: in component recognition, that is, answering the question "is X a resistor?", in circuit recognition, answering "is this circuit what it should be," and in a combination of these, component/circuit identification with a symbolic model. (In this section, program examples will be presented in a meta-language for ease of reading. In fact they are not implemented in this form.)

Component Recognition

In a resistor model data Is passed in the form of pointers to picture points and what the intensities should be at those points Resistors are the simplest of this type of model. Other more complex mode]s make use of backup, color information, region growing, etc.

Another more difficult problem arises when the appearance and/or location of the component cannot be known in advance. In that case, the model must first locate the desired object and then inspect it, or, in failing to successfully locate it, report it missing. This problem arises in the hybrid circuit inspection system in the case of wires.

CircuitVerification

The second type of model is similar to the first, in that the circuit Is described in terms of a coordinant-based model. In this case, the circuit model might look like this:

```
PROCESSCIRCUIT(X)«BEGIN
     %x is assumed to point to a list structure
describing where components should be located.
The process FINDCOMP analyzes this structure to
select a descriptor and returns it %
     DECLARE OWN Y;
     Z<--
     RTEST:BEGIN
          DECLARE BACKUP TRUE:
          Y<---FINDCOMP(RESISTOR,X);
          IF Y EQL FALSE THEN LEAVE RTEST WITH TRUE;
          IF NOT RESISTOR(Y,IMAGE) THEN FAIL;END;
     IF Z EQL FALSE THEN BEGIN
          TYPE('DEFECTIVE RESISTOR AT');
          TYPE(GETPROP(Y.XCOORD),',',GETPROP
          (Y,YCOORD));END;
END
```

This simple process uses backup in a limited form to verify all the resistors on a circuit board. The list containing the desired locations of the resistors is passed as an argument, X. FINDCOMP finds an entry on X which has been marked as describing the location of a resistor. FINDCOMP returns FALSE when ft cannot locate any more resistors. If RESISTOR locates a defective resistor, it marks the property list of its argument, Y, with the coordinants of the defective resistor and returns FALSE. If RESISTOR returns TRUE, then the conditional IF NOT RESISTOR fails, and control backs up to the FIND to locate another component totry.

Symbolic Representation of Circuits

The final form of procedural model in the case of circuit analysis makes use of a symbolic or schematic representation for the circuit and performs the analysis in terms of that representation. In this case, the problem is: given a schematic representation of the circuit, one in which the components are simply described as "Rl" for some resistor, etc., and given an image of a real circuit which implements the schematic representation, perform the correlation between the components of the image and the symbols in the schematic representation.

Let us consider such a representation of the circuit shown below.

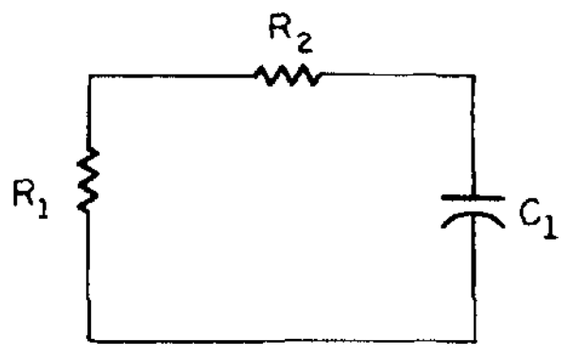

Figure 1. A Simple Circuit

```
PROCESS SYMCIRCIT+BEGIN
     DELCARE BACKUP TRUE;
     R1«-FINDREGI0N(DARK,0) , '.find any old dark
     region
     FAILTEST(Rl);          Ican't find one, quit
     IF RESISTOR(RI) THEN Y1«-FINDREGI0N(ADJACENT,
     Rl);
     IF CONDUCTOR(Yl) THEN R2<---FINDREGION (ADJACENT,
     Y1)
     IF RESIST0R(R2) THEN Y3<---FINDREGI0N(ADJACENT,
     R2);
     IF CONDUCTOR (Y3) THEN C1<---FINDREGION(ADJACENT,
     Y3);
     IF CAPACITOR(Cl) THEN Y5<---F INDREGION(ADJACENT,
     CI);
     IF C0NDUCT0R(Y5) AND ADJACENT(Y5,R1)THEN
     SUCCEED;END
```

This process embeds the structure of the circuit Into the process without resorting to data structures. First, the assumption Is made that the dark region found is Rl, and search is made for regions bordering Rl which might be the conductor leading to R2. In case of failures, the process backs up, changes its hypothesis, and continues. Some of the intricacies which could easily be added

include a more sophisticated search, remembering partial results, rather than this essentially depth-first search.

A reasonable approach to the problem of generating procedural models is to allow the system to learn new models by examples. This Is the approach taken in the system implemented and described in the next section. The model which the system generates Is a process description for identifying the various components and checking that those components are properly located within reasonable assembly line tolerances. The model is them compiled, and can be used to check for defects on other, similar circuits.

4. Component Models

Resistors

Resistors are typically so dark as to have no significant color but can be distinguished easily by the fact that they are so very dark. They usually (but not always) have borders which are straight lines. They border either conductors or the substrate, both of which are bright regions and thus (except when obscured by wires or pads) resistors have easily distinguished borders.

The setup phase, when told where a resistor is located checks the intensity of the region and locates the borders. It then sets up the model of the circuit it is building to look in the appropriate places when it is inspecting circuits.

The type of capacitor for which the system was developed has two metal plates at either end of a region of homogenous intensity.

First the central region of the capacitor is located and its color determined. Then, an outward scan is made using the Hueckel [6] operator in an effort to locate the end plates. These metal end plates suffer from the same problems as wires, reflections. End plates, however, are parallel to each other, thus when the edge detection operator locates an edge, it verifies that it is parallel to a corresponding edge on the other side of the capacitor. In this way, improperly mounted capacitors can be detected, as well as those that are missing or dirty.

Wires

Because wires are flexible and can be bent in a variety of different shapes while maintaining good contact with both lugs, their location Is unpredictable. In addition, they are three-dimensional in that they may pass over other components, thus obscuring those components from view.

Wires are "shiny." (They are essentially convex mirrors.) The shiny property of wires can be described as follows:

1) Wires have no characteristic color; their color is totally dependent on incident light and angles of reflection.

2) Depending upon the angle of the wire, light source, and camera, a wire may possess highlights (direct reflection of light source into the camera) and nulls (reflection of a very dark region Into the camera). A wire thus typically appears as a set of long, thin bright areas running in a more or less linear fashion from one lug to another. A highlight on a wire may pass over a portion of the substrate which is white, and effectively merge into a region of uniform (high) intensity.

An interesting characteristic of highlights is that they move when the viewing angle la changed, or the light source is moved. Thus one way to identify a highlight is to note its motion when the light source is changed.

Since all the components are generally irregular and noisy, identification of an object whose sole characteristic is that It reflects these irregularities and noise isn't easy. However, since we are given the starting and ending points of the wire with reasonable precision, it is possible to detect it. The visual properties described suggest the following algorithm: first, a "masked difference" picture is generated from two pictures taken with different lighting angles. A new picture is created where each point in the new picture is equal to the absolute value of the difference of corresponding points in the two pictures. In this difference picture, the regions which are most significant are those in which the illumination has undergone a large change. This effect is maximum in the case of nulls, highlights, and shadows. The difference picture is then used as a mask to accentuate the highlights.

Three routines, WIREFI, WIRE, and MAKEWIRE, call each other in the wire-finding process. The flow of control is shown in the figure below.

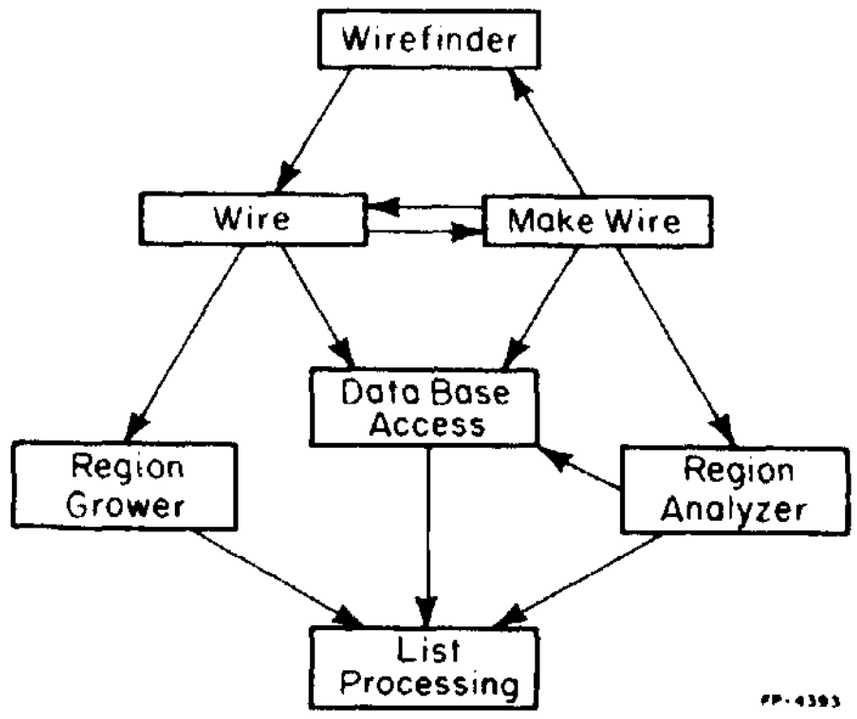

Figure 2. Flow of Control During Wire-Finding

WIREFI sets up the desired end points between which there should be a wire.

WIRE searches between those end points and off at different angles, using either no a-priori information or possibly using "hints" passed from MAKEWIRE to direct its search. It calls a region grower to locate regions and makes tentative decisions about whether or not a region might be part of a wire. When its search is completed, WIRE calls MAKEWIRE.

MAKEWIRE uses higher level knowledge to operate on the list of regions passed Iroin WIRE to decide whether or not the information given is sufficient to identify a wire. If there are large empty spaces between regions, it can call W1REF1 to examine these spaces in more detail.

One interesting problem which arises in the region analysis is that of finding the extrema of arbitrary regions. It is computationally unreasonable to compare all points on the boundary of a region, two at a time. A search algorithm was developed which usually converges in three iterations and returns those two points on the boundary which are a maximum distance apart.

Performance of the System

Though the system was written in BLISS, many LISP features were carried over. For example, the functions CAR, CDR, etc., and garbage collection were implemented in BLISS for the sake of convenience. The use of lists allows for easy and rapid implementation of fairly complex data structures. In exchange for this ease of developmental programming, one pays a high price in efficiency.

Because the time required to perform the verification depends so strongly on the complexity of the particular circuit being analyzed, it is difficult to estimate. The most time consuming part, the wire finding, with Its accompanying region growing, takes just under a second for an average wire, longer for longer wires.

The accuracy of the system in detecting defects is also a function of the training procedure. Since the system only looks where it is told to look, it is always possible to Invent a type of defect which any particular model could not detect. In general, a large class of defects could be detected by checking the wires, the resistors, and the capacitor.

Bibliography

[1] Brice, C. and Fennama, C., "Scene Analysis Using Regions," Stanford Research Institute, Menlo Park, California; Artificial Intelligence Group Technical note 17, (1970).

[2] Ejiri, M., Uno, T., Mese, M., and Ikeda, S., "A Process for Detecting Defects in Complicated Patterns," Central Research Laboratory, HITACHI LTD., Kokubunji, Tokyo, Japan.

[3] Finin, T., "Tracking Wires on Printed Circuit Boards," Working paper 52, MIT Artificial Intelligence Laboratory, October, 1973.

[4] Harlow, C. A., Henderson, S. E., Rayfield, D. A., Dwyer, S. J., "Automated Inspection of Electronic Assemblies," unpublished manuscript, Image Analysis Laboratory, Department of Electrical Engineering, University of Missouri, Columbia.

[5] Hewett, C, "Description and Theoretical Analysis (Using Schemata) of PLANNER: A Language for Proving Theorems and Manipulating Models in a Robot," AI Memo No. 251, MIT Project MAC, April 1972.

[6] Hueckel, M., "An Operator Which Locates Edges in Digitized Pictures," Stanford University, Artificial Intelligence Memo 105, Stanford, Ca. For a more recent and improved operator, see JOURNAL OF THE ACM, October (1973).

[7] Krakauer, L., "Computer Analysis of Visual Properties of Curved Objects," MIT, MAC TR-82, Cambridge, Mass. (1971).

[8] Lozano-Perez, T., "Finding Components on a Circuit Board," Vision Flash 51, MIT Artificial Intelligence Laboratory, September 1973.

[9] Simon, The Sciences of the Artificial.

[10] Shirai, Y., "A Heterarchial Program for Recognition of Polyhedra," MIT, Al memo 263, Cambridge, Mass. (1972).

[11] Winograd, T., "Procedures as a Representation for Data in a Computer Program for Understanding Natural Language," A1TR 84, MIT, Cambridge, Mass. (1971).

[12] Winston, P., "Learning Structural Descriptions from Examples," MIT, AITR-231, Cambridge, Mass. (1970).

[13] Yachida, M. and Tsuji, S., "Application of Color Information to Visual Perception," PATTERN RECOGNITION, 3, pp 307-323, (1971)

[14] Yakimovsky, Y. and Feldman, J., "A Semantics-based Decision Theory Region Analyzer," Proceedings of the Third International Joint Conference on Artificial Intelligence, Stanford, Ca. (1973)

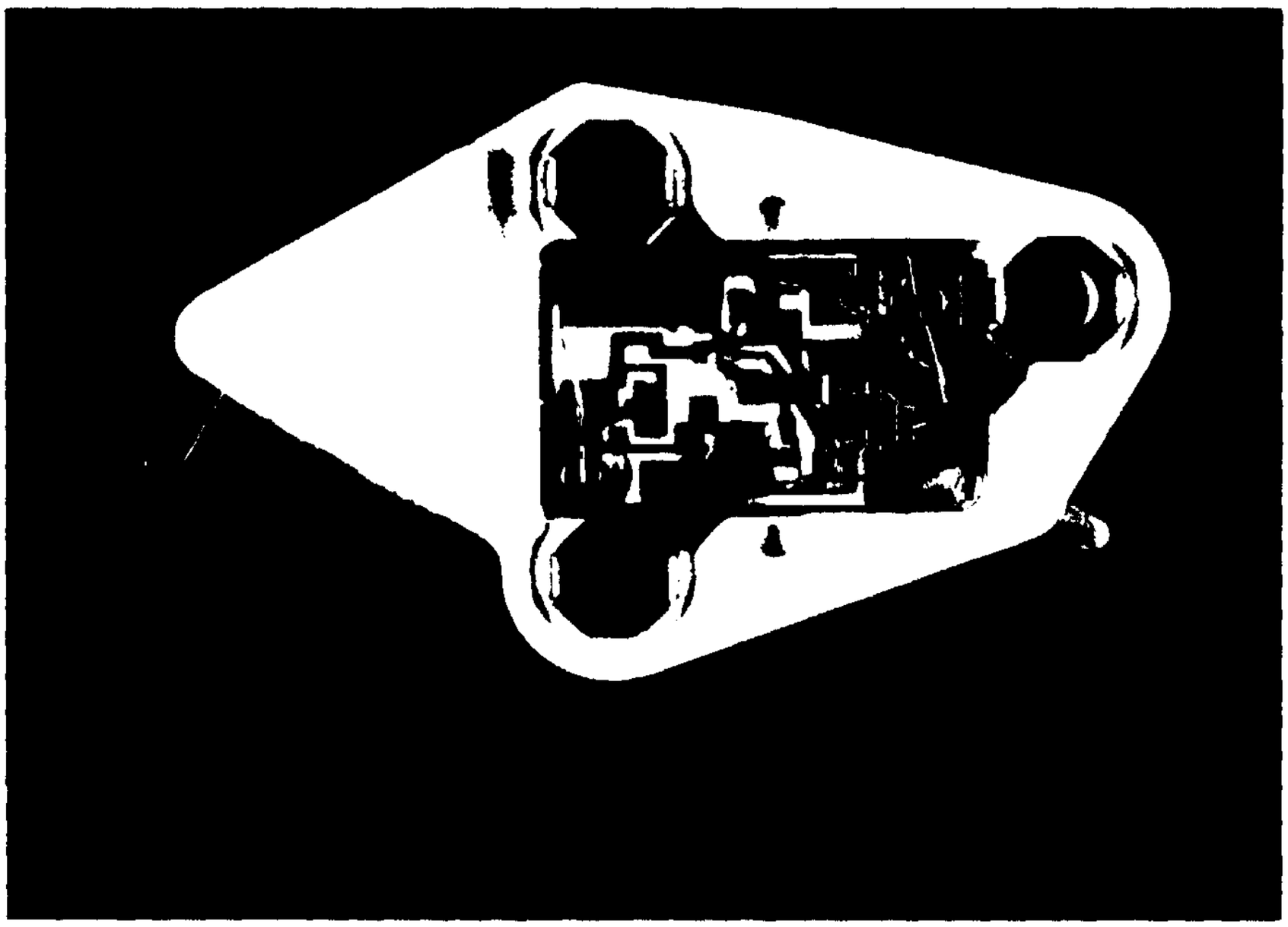

Figure 3. A Hybrid Circuit
(Courtesy of General Motors/Delco)

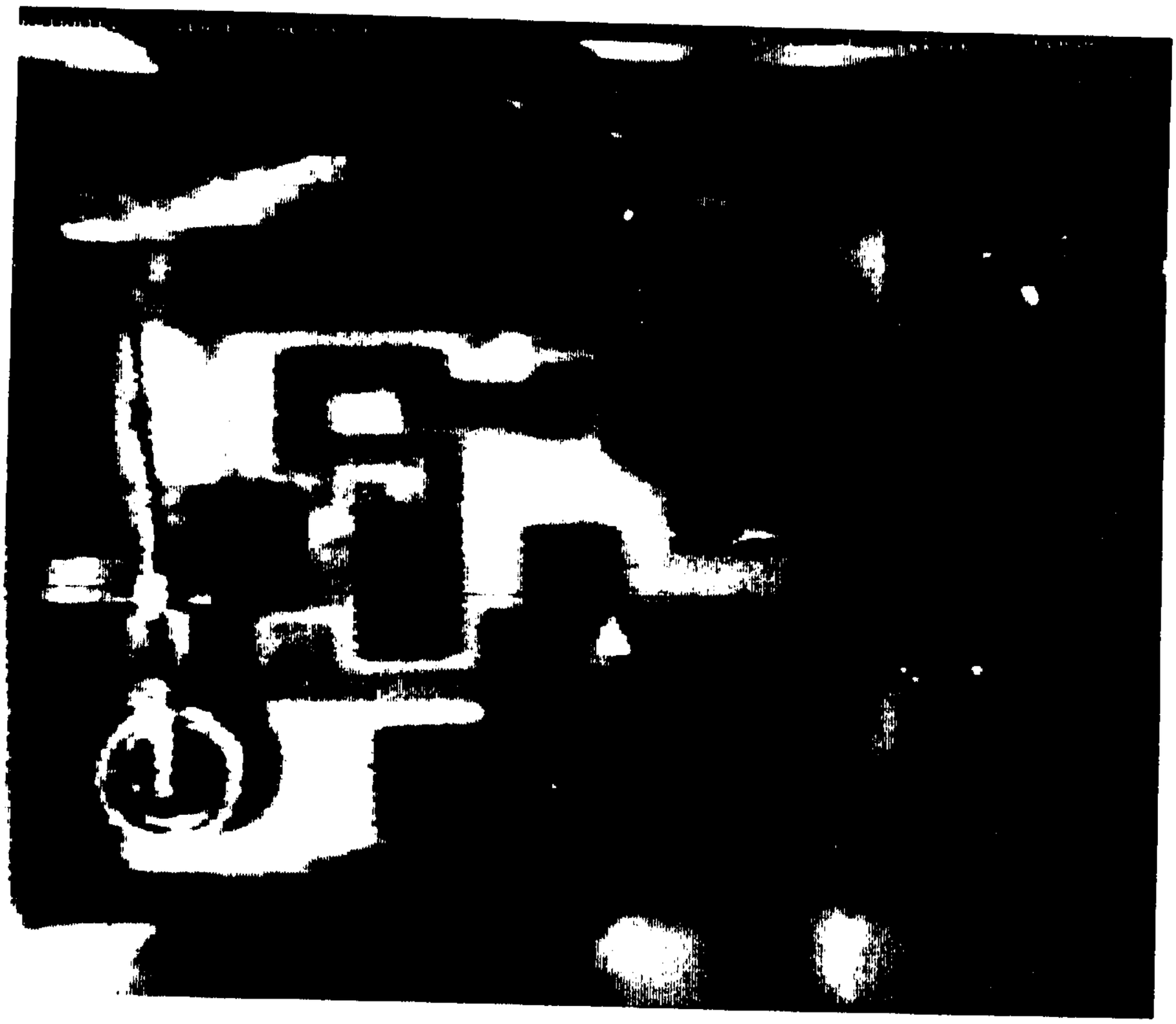

Figure 4. The Same Hybrid Circuit as Viewed by the Computer

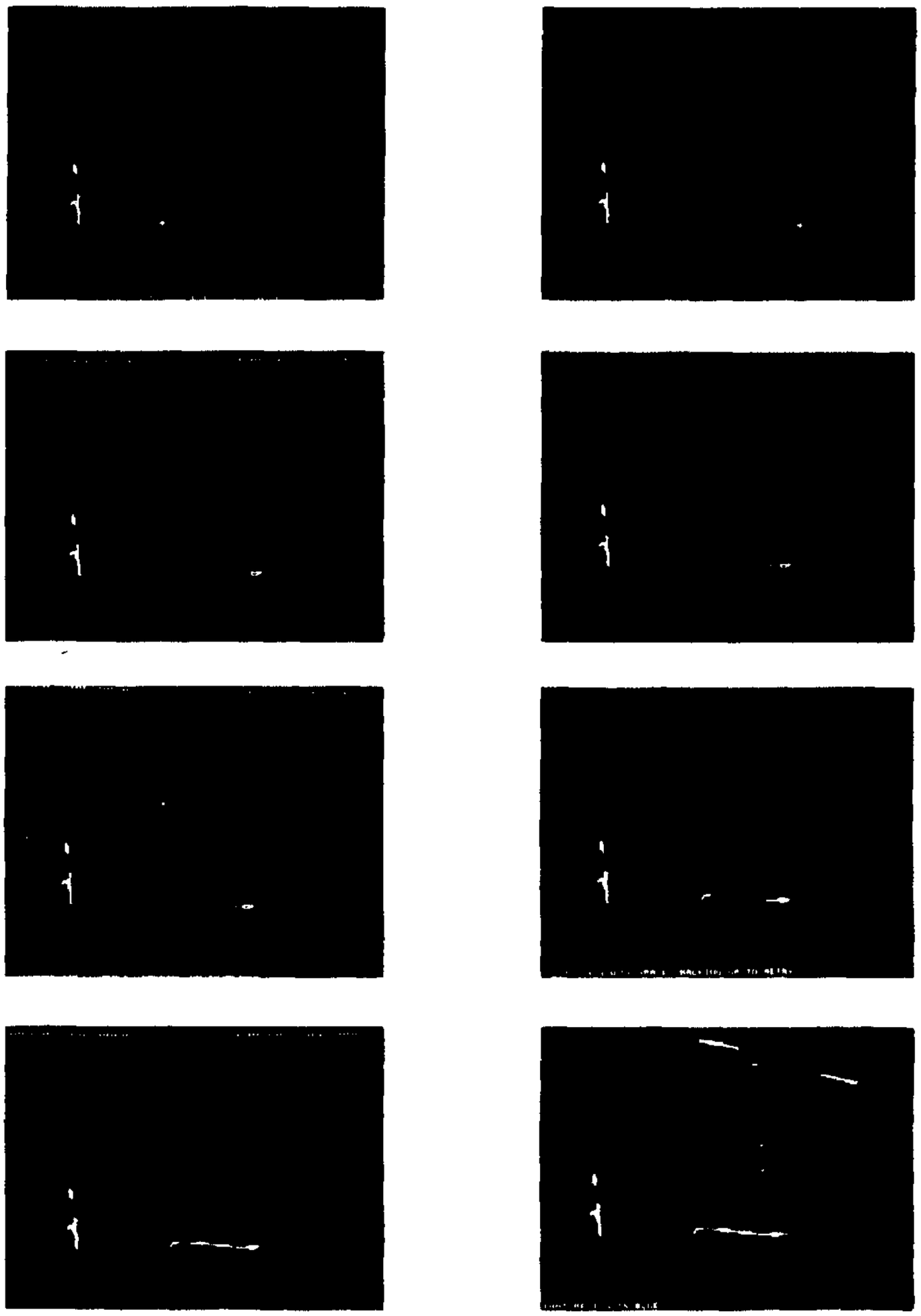

Figure 5. Wire Tracing Sequences