MACHINE VISION INSPECTION OF COMPONENT FOOTPRINTS IN PCB ARTWORK

RADHA KRISHNAN* AND H. J. SOMMER III†

Abstract. A new printed circuit board (PCB) inspection scheme designed to perform automated inspection of component sites is presented. A digitized image of the artwork for a board is stored after digitization via a scanner (300 dots per inch) in a TIFF (Tag Image File Format) file. At inspection time, after reading in the compressed format TIFF file and decoding the compressed image, the PCB is graphically displayed on a PC screen together with a command menu. The user is then taken through a series of menu-driven commands to ultimately check the dimensions of center-to-center distance between holes and pads for electronic components on the PCB. The significance of this research is in the adaptation of the UNION-FIND procedure to develop a robust algorithm to segment an image into component objects and background to facilitate dimensional analysis and inspection of the artwork.

1. Introduction. Traditionally printed circuit boards have been inspected manually for dimensional accuracy. With the advent of machine vision inspection schemes, it has become expedient to automate such tedious tasks. Printed circuit boards are manufactured by first producing an artwork, which will subsequently be used as a mask for the board fabrication. It is thus important to inspect the artwork for dimensional fidelity as well as electrical continuity prior to the masking operation. A complete package was developed to read a TIFF file containing the image of such artwork, zoom into small target areas of this image and segment the image portion into features (holes or pads) whose center-to-center distance can be measured. In addition a visual ruler facility is provided to manually measure distance between arbitrary locations.

The inspection scheme proceeds as follows. Artwork of a printed circuit board is digitized using commercially available scanners with a reasonably good resolution such as 300 dots per inch. The digitized images are stored in a compressed TIFF format based on the Huffman encoding scheme. At the time of inspection, an image is brought up on an IBM PC screen as a graphical object. Small regions of the image are examined one at a time by zooming on specific areas. First, manual and automatic software tools are provided to remove the effects of noise due to digitization. Next the traces (connections between holes) in the zoomed area are thinned and removed using morphological erosion. This leaves an image consisting of only the holes and pads on the artwork. Next, a segmentation operation using the UNION-FIND algorithm is performed to classify these holes/pads as individual objects. A moment-based method is used to compute the centers of these objects to enable distance measurements between objects. Finally, upon identification of the part by

* Mechanical Engineering, University of Maryland Baltimore County, Catonsville, MD 21228.
† Mechanical Engineering, Pennsylvania State University, University Park, PA 16802.

0-8194-0873-5/92/$4.00

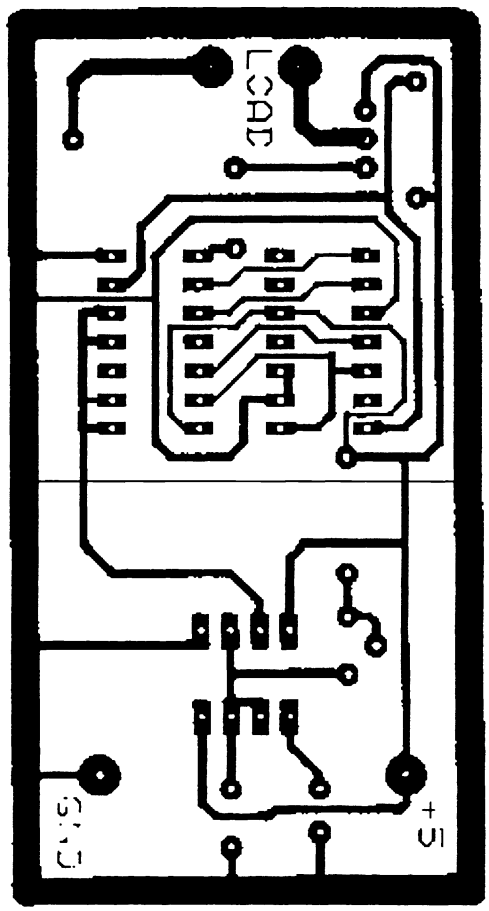
the user, a master database is consulted for the correct dimension and checked with the measured value. An accept/no-accept result is written into the board database against the part under consideration.

This paper is organized as follows. The first section describes the preprocessing steps consisting of zooming, erosion, filling small holes to remove spurious/noise objects and so on. The second section describes the segmentation algorithm. The third section outlines the computation of the centers and measurement of distances using rulers. The fourth section briefly indicates the processes involved in interfacing with the master and part-specific databases. Finally, concluding remarks, future directions and scope of the research work are presented.

2. Preprocessing. TIFF image files of artwork inherently contain binary images and hence a question of thresholding does not arise. The image consists of the entire PCB as shown in Fig. 1, and therefore, in order to examine specific components, one needs to zoom on specific areas of interest by interactively positioning the area-of-interest (AOI) rectangle on-screen. With the AOI rectangle positioned as shown in Fig. 1, the result of a zoom operation is shown in Fig. 2. In addition, the size of the area of interest can also be manipulated to enable zooming into variable sized objects in the image. The zoomed image portion can then be enhanced prior to segmentation. This is necessary due to digitization noise present in the image [1]. This noise may introduce small spurious objects or produce extensions to real objects. To handle the former, an area threshold function is provided, which performs an image-segmentation operation and removes objects which have areas less than a pre-defined size. To remove other noise effects such as extensions to the objects, a manual eraser is provided. This is a simple graphical interface which removes the area under the cursor.

PCB's and their artwork, consist of traces running between pads, which will subsequently form the electrical connections between the components that will be located at these pads. Since center to center distance estimates between connected pads will be skewed if the traces are not removed, the second preprocessing stage consists of eroding away the traces to leave behind an image consisting exclusively of features to be inspected. The erosion operation [3, 7] is a simple morphological process which reduces the object by one pixel thickness over its entire perimeter. This operation is performed repeatedly until all traces are completely eroded. In order that the erosion operation does not exploit holes present in the footprints, a fill operation is performed prior to the erosion. Fig. 3 shows the effect of a filling operation performed inside the AOI rectangle. The result of performing the erosion operation on the artwork is shown in Fig. 1 is shown in Fig. 4. Finally, spurious noise-induced projections are cleaned out in a manual cut operation. Effect of such an operation on the image in Fig. 1 is shown in Fig. 5.

3. Image Segmentation. Next, to delineate features from background, a segmentation operation [12] has to performed. Various heuristics and algorithms have been proposed for image segmentation [8, 10, 11, 9]. In this application, the UNION-FIND algorithm [4] was chosen because of its elegance and ease of implementation. This algorithm works as follows. Initially every vertex (pixel) is considered to be in a



Row
10
Column
3

Use arrow keys to
position AOI box.
Then choose :

ESC
Zoom

FIG. 1. Raw Image of a PCB Artwork

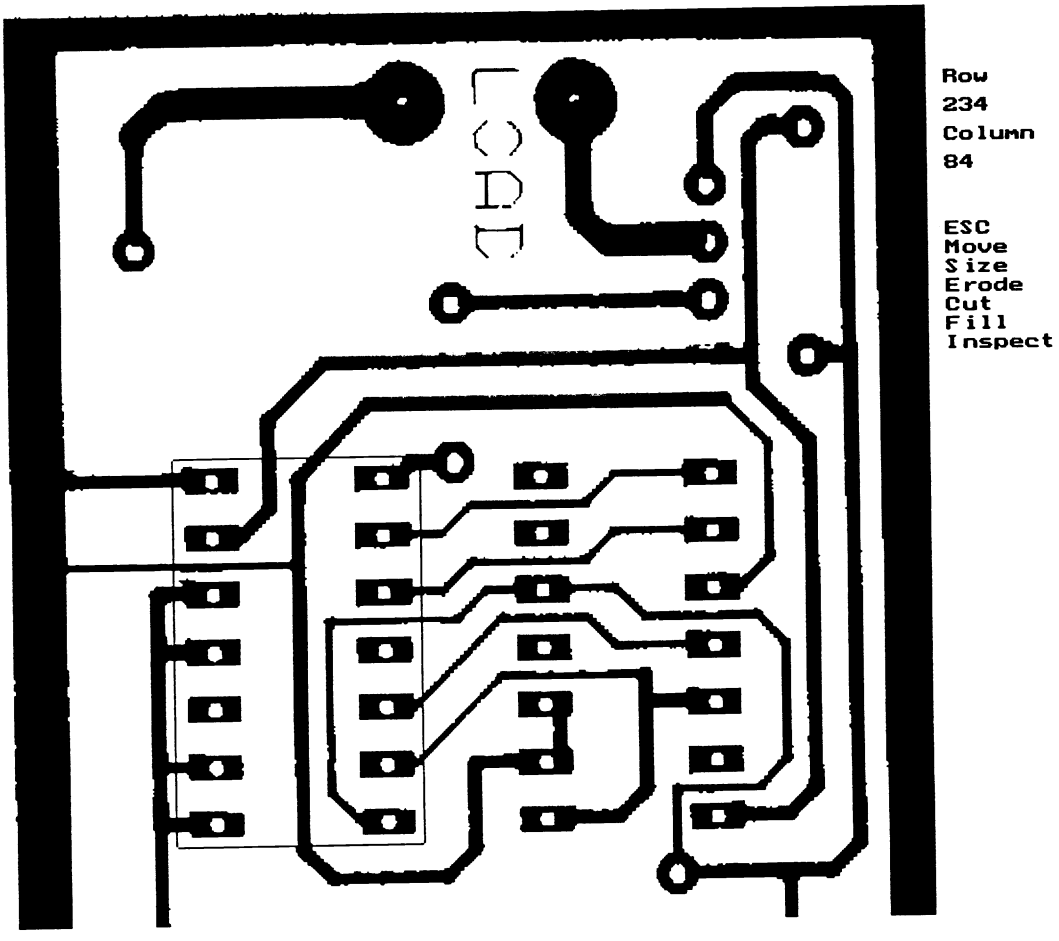


FIG. 2. Result of zooming into AOI

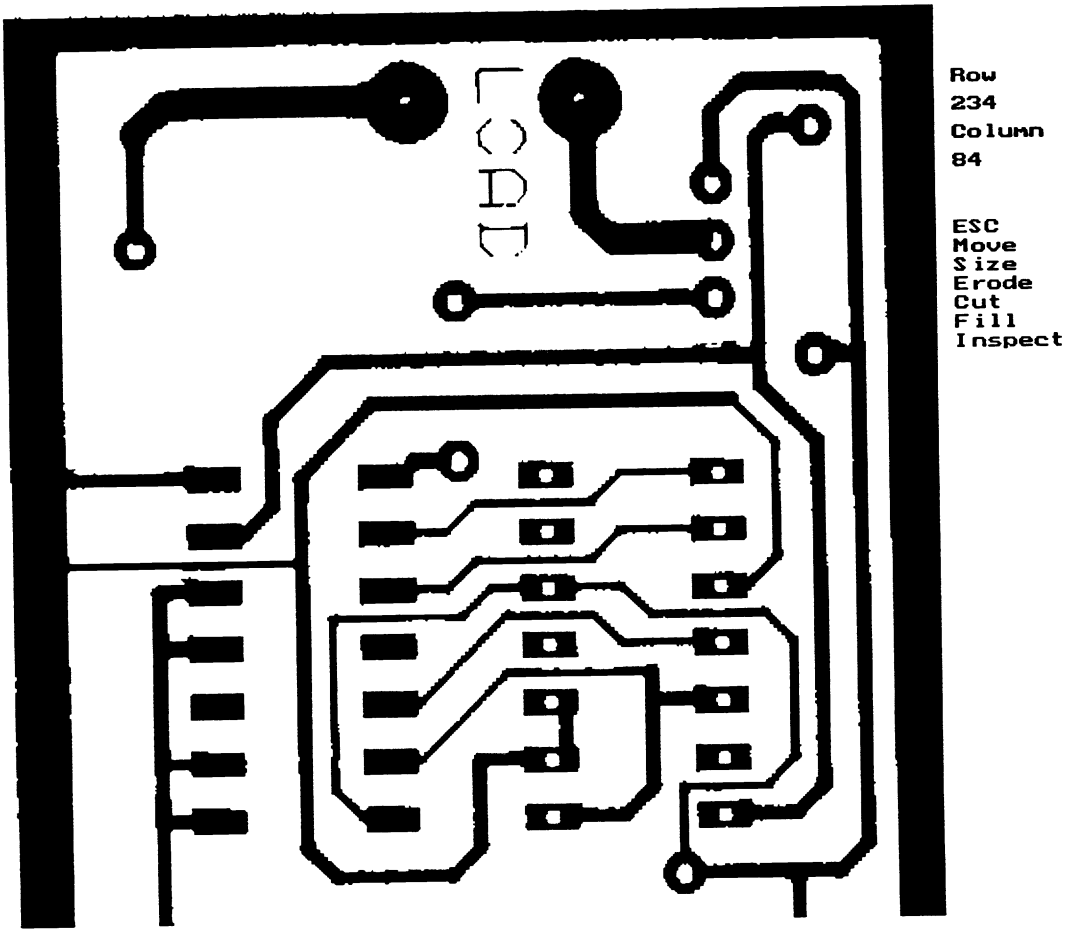


FIG. 3. AOI after fill operation

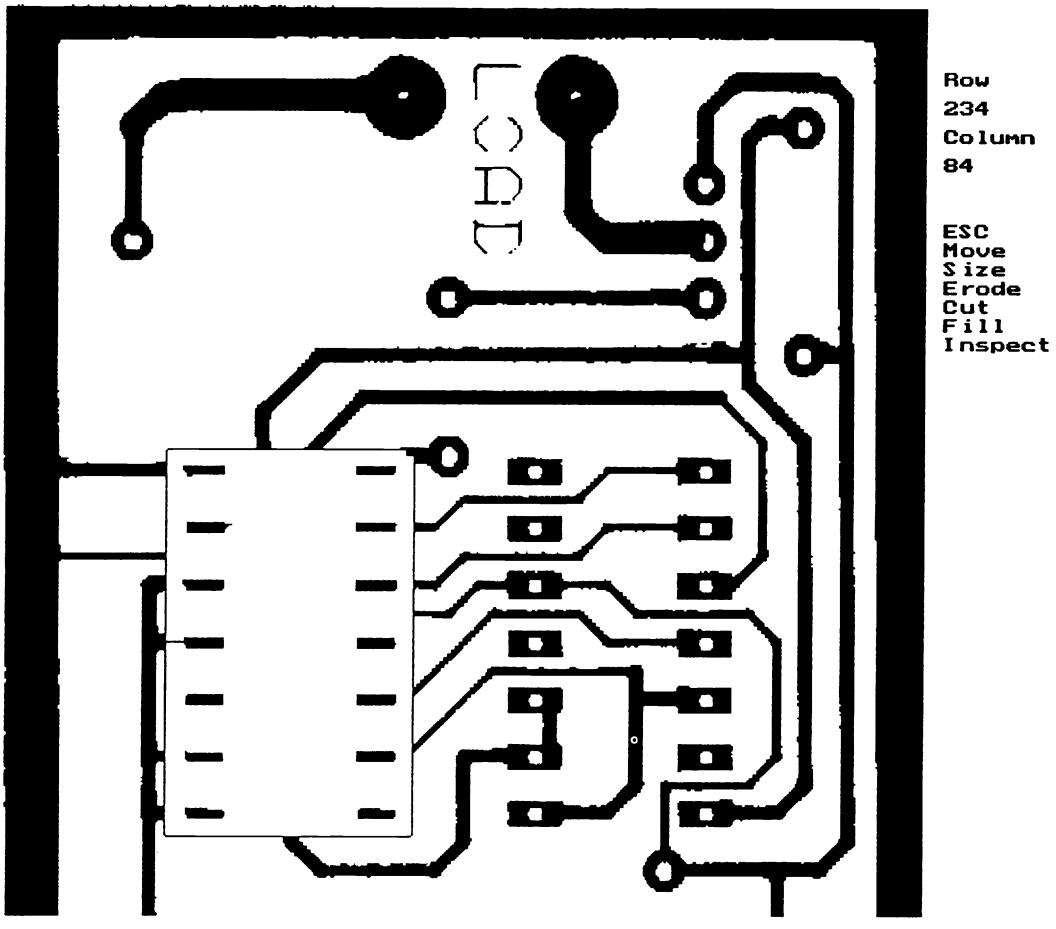


FIG. 4. Result of erosion inside AOI

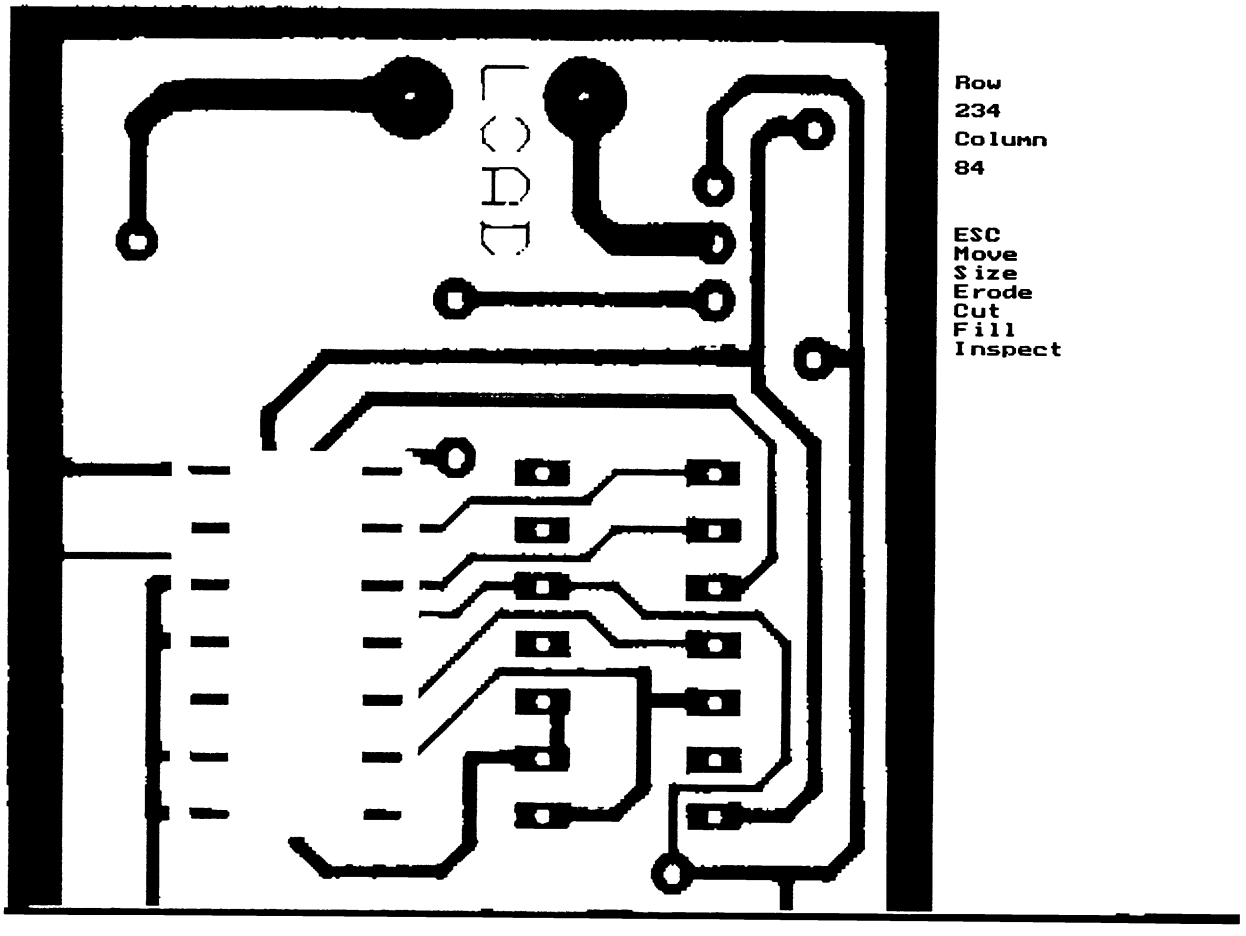


FIG. 5. Result of manual cut inside AOI

set (object) by itself. Two sets are merged if and only if any element of one set is a neighbor of an element of the other set. A neighbor is defined as any one of the pixels from the eight cells surrounding the current pixel. As the segmentation proceeds, if two pixels belonging to two different objects are found to be neighbors then the two sets are merged. Starting with this elementary concept dependent upon connectivity, the algorithm can be optimized to perform the segmentation operation in less time. The optimization process requires the use of appropriate data-structures in addition to a more complex implementation process itself. These procedures are described below.

Formally the UNION-FIND problem is defined as follows. Initially there are n elements $x_1, x_2, ..., x_n$, each one belonging to its own individual set. The problem is to execute an arbitrary sequence of the two operations:

- **find(i)**: returns the name of the set to which element x_i belongs.
- **union**(A, B): performs $A \bigcup B$ and returns the name of the union.

The simplest and most straight forward data structure to use would be an array representation, where $A[i]$ would store the name of the set to which element x_i belongs. In such a case, the find operation requires only constant time since it is just an array indexing operation. But the union operation requires $O(n)$ time for every operation. This is clearly not efficient.

In order to make the algorithm more efficient we need to improve the data structure. Instead of the simplistic array representation chosen above which just stores the set name, we can choose the array location to be a record [2] which holds the identity of the set as well as a pointer to the last set with which this set was unioned. That is, initially all pointers in the second field of the record are nil (no union operation has yet been performed). When two sets are unioned, the pointer field of one of the sets is made to point to the pointer field of the other. An extra field is introduced in the record to decide which set becomes the parent after the union. Each vertex stores the identity, a pointer to its parent *and* the number of children it has. In order to retain some amount of balancing, the vertex with the smaller number of children of the two candidates for union is made a child of the other. This ensures that the height of all trees is always less than $O(\log_2 n)$ [5]. Since the union operation now takes only constant time per operation, and the find operation takes $O(\log_2 n)$ time, any arbitrary sequence of m operations takes $O(m \log_2 n)$ time.

The above time can be reduced further by utilizing a technique known as path compression which reduces the height of a tree further. Consider a find operation. This is just an array look-up followed by a series of tree traversals all the way to the root of the tree. The idea of path compression is to re-trace this path, changing all the pointers along the way to point directly to the root. This produces a total time complexity of $O(m \log^* n)$ which was first obtained by Hopcroft and Ullman [4], and later improved by Tarjan [6] to $O(m\alpha(n))$, where $\alpha(n)$ is the inverse Ackerman's function which grows slower than even $\log^* n$.

The algorithm was implemented on a PC AT-386 in C-language. Various board images were experimented with and the result of application on the current image AOI is shown if Fig. 6.

4. Measurement. The idea of a geometrically regular shape can be used to provide moment-based measures for the centers of segmented objects in the image. Consider a gray scale image with intensities given by g_{ij} at image point (i, j). The mass **M** of an object and the first moments of this object about the coordinate axes can then be expressed as

$$\mathbf{M} = \sum_i \sum_j g_{ij} \qquad I_x = \sum_i \sum_j (i \cdot g_{ij}) \qquad I_y = \sum_i \sum_j (j \cdot g_{ij})$$

The coordinates of the mass center of the object can then be obtained as

$$x_c = I_x/\mathbf{M} \qquad y_c = I_y/\mathbf{M}$$

Note that for binary images, all g_{ij}'s are equal to one, and the mass of the object is simply equal to the area of the object.

Having obtained the centers of the objects (shown as crosses in Fig. 6), an initial visual ruler is provided by placing a pivot at the center of the object and a tip at the center of the second object. From then on, both the pivot and the center can be relocated anywhere the user wishes and the screen measurement provides both the center to center to distance of the objects and the center to tip measure of the ruler. This second measure is useful in case of both computationally anomalous centers and for measuring arbitrary distances on the objects.

5. Interface to Database. The ultimate objective is to check if all the measurements performed on any component match the specifications for the component. For instance, in testing holes meant for chip insertion locations, the various parameters are the lateral center-to-center distance between holes on opposite edges of the chip, the longitudinal center-to-center distance between holes on the same side, that is the pitch and separation of the holes respectively. This objective is achieved by selecting into the chip region, cleaning the zoomed area and then segmenting the chip artwork into its holes/pads to check the respective center-to-center distances using the ruler. These distances are then compared to specifications for footprint dimensions required for this particular chip. The user supplies the information regarding the name of this component to enable the database look-up. If the differences between the measured and the required values for all the parameters of the chip are within specified tolerances, then the component location on the artwork is accepted and graphically highlighted as such. Otherwise it is rejected and so marked. This information is then stored in a separate database consisting of individual artwork files.

6. Conclusion. A new automatic PCB artwork inspection scheme is presented together with results using real images. The theoretical interest and contribution lies in the adaptation of the UNION-FIND algorithm to segment the image. Practically, a significant contribution has been made towards developing more autonomous

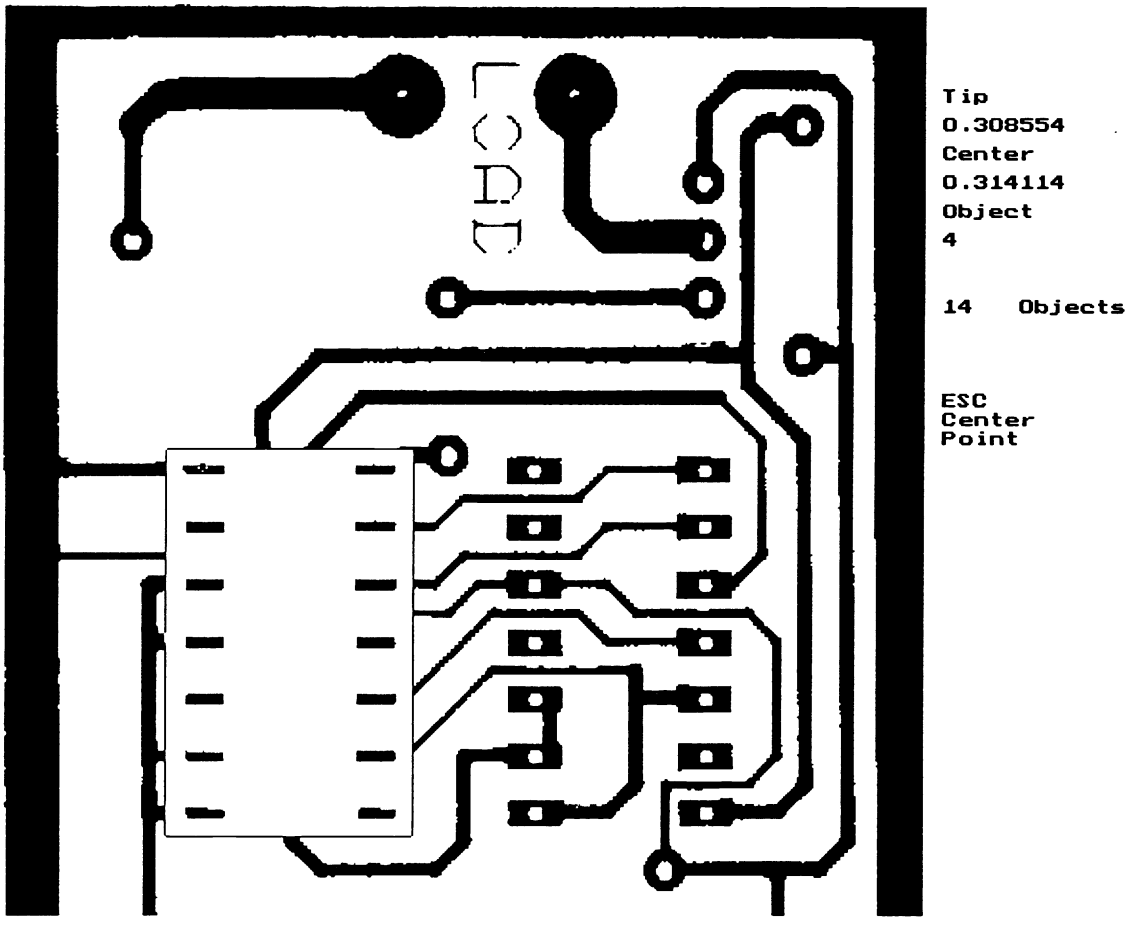


FIG. 6. Segmented AOI with centers superimposed

computer vision schemes to replace tedious manual inspection. Future work should be directed towards eliminating the small amount of human intervention required in terms of area selection and component identification. This can be achieved by segmenting the *whole* image into component objects and identifying the individual parts by their unique footprints or by etching the chip label into the artwork and using optical character recognition on the lettering at inspection time.

REFERENCES

[1] D.H. Ballard and C.M. Brown, *Computer vision*, Prentice-Hall, Englewood Cliffs, N.J., (1982).

[2] B.A. Galler and M.J. Fischer, *An improved equivalence algorithm*, Communications of the ACM, (1964), pp. 301–303.

[3] R.C. Gonzalez and P.A. Wintz, *Digital image processing*, Addison-Wesley, Reading, MA, (1987).

[4] J.E. Hopcroft and D. Ullman, *Set-merging algorithms*, SIAM J. Computing, (1973), pp. 294-303.

[5] U. Manber, *Introduction to Algorithms: A Creative Approach*, Addison-Wesley, New York, (1989).

[6] R.E. Tarjan, *Efficiency of a good but not linear set union algorithm*, J. ACM, (1975), pp. 215–225.

[7] J. Mandeville, *Novel Method for Analysis of Printed Circuit Images*, IBM J. Res. Develop (29), (1985), pp. 73-86.

[8] A. A. Rodriguez and O. R. Mitchell, *Image Segmentation Using Background Estimation*, Intelligent Robots and Computer Vision: Seventh in a Series, Proc. SPIE 1002, D. P. Casasent, Editor, (1989), pp. 170-181.

[9] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Englewood Cliffs, NY, (1988).

[10] A. Rosenfeld and A. C. Sher, *Detection and delineation of compact objects using intensity pyramids*, Pattern Recognition (21), (1988), pp. 147-151.

[11] P. Meer, D. Mintz, A. Montanvert and A. Rosenfeld, *Consensus Vision*, Proc. AAAI-90 Workshop on Qualitative Vision, Boston, MA, July 1990, pp. 111-115.

[12] T. Pavlidis, *Structural Pattern Recognition*, Springer-Verlag, 1977.