MACHINE RECOGNITION OF HAND-DRAWN CIRCUIT DIAGRAMS

B. Edwards and V. Chandran

Research Concentration in Speech, Audio and Video Technology,

School of Electrical and Electronic Systems Engineering

Queensland University of Technology, Brisbane, Australia

mailto:v.chandran@qut.edu.au

ABSTRACT

An application of image processing techniques to recognition of hand-drawn circuit diagrams is presented. The scanned image of a diagram is pre-processed to remove noise and converted to bi-level. Morphological operations are applied to obtain a clean, connected representation using thinned lines. The diagram comprises of nodes, connections and components. Nodes and components are segmented using appropriate thresholds on a spatially varying object pixel density. Connection paths are traced using a pixel-stack. Nodes are classified using syntactic analysis. Components are classified using a combination of invariant moments, scalar pixel-distribution features, and vector relationships between straight lines in polygonal representations. Node recognition accuracy of 92% and component recognition accuracy of 86% was achieved on a database comprising 107 nodes and 449 components. This recogniser can be used for layout "beautification" or to generate input code for circuit analysis and simulation packages.

1. INTRODUCTION

Document image analysis [1] is an active and challenging area of research in computer vision. Documents comprise of text and graphics. Graphics recognition has thus far generated less interest than text recognition. Machine recognition of hand-drawn graphical entities such as circuit diagrams, flow charts, tables, etc. will add another dimension to human computer interaction. It can be used to "beautify" diagrams to professional standards of layout and drafting, or to produce input code for circuit analysis programs.

2. SYSTEM STRUCTURE

Circuit diagrams are described using the following terminology for parts:

Component: Graphic symbol for a circuit element
Port: Point for input or output
Contact: Point on a component for connecting
Connection: Single line representing a conducting path that joins components, ports, contacts or nodes.
Node: Point where two or more connections meet or a set of such points

Nodes can be of different types, depending on the number of connections, and are classified in this work using a tree grammar. Components are also of many different types. For the purpose of implementing a quick prototype solution, a limited set of 9 components was used in the work described here. It is general enough to include all BJT circuits. Labels and component values are ignored in the work described here.

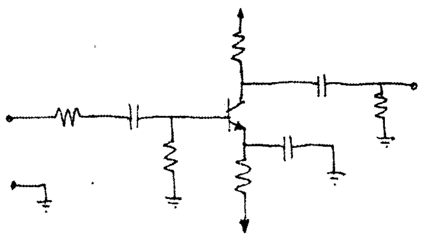

Figure 1. A hand-drawn circuit diagram

Images of hand-drawn circuit diagrams are acquired with a scanner. They are passed through the following stages - (a) pre-processing, (b) segmentation, (c) path tracing, (d) node recognition, and (e) component recognition.

3. PREPROCESSING

It is assumed that the paper is clean with no ruled lines or other patterns. It can be plain or any colour provided the contrast with the pen is good enough. The system is designed to be tolerant of common errors such as (a) small gaps where lines intersect, (b) small spurs anywhere on lines, and (c) curves in otherwise straight lines. To provide a good balance between processing speed and accuracy in representing image detail, the resolution of the scanner was determined to be between 150 and 250 dots per inch. 8-bit quantised colour images were input to the pre-processor. The pre-processor performed four operations - (a) monochrome conversion, (b) noise reduction, (c) closing and (d) thinning.

The colour image is first convert to graylevel using only luminance from the Red, Green and Blue values. Figure 1 shows the graylevel image of a hand-drawn circuit. The image is binarised using a moment-preserving threshold proposed by Tsai [2]. O'Gorman and Kasturi [1] have found this technique to be "more effective in binarising document images containing text." It sets a threshold that best preserves moment statistics in the binary image compared to the original image.

If the paper is not clean or the quality of the scanner is poor, the binary image can contain salt and pepper noise. It is then removed using the k-Fill technique [1]. It is effective in removing small spots but is an iterative and slow technique.

Morphological closing (dilation followed by erosion) is used to close small gaps between lines and small circles drawn to represent ports.

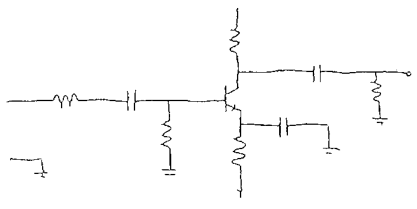

Figure 2. The Pre-processed Image

The Zhang-Suen algorithm [3] was used for thinning because it is fast. Minimal 8-connectivity is achieved by using a staircase removal procedure that removes unnecessary corner pixels. The thinned image is shown in figure 2.

4. SEGMENTATION

Segmentation is the process of breaking up the image into pieces that are small enough to be recognised. For this application, segmentation is used to (a) separate the components and nodes, and (b) separate the connections, from the image. Components, nodes and connections are all made up of lines of different lengths, orientations and curvature. Knowledge of the line structures is useful for segmentation as well as classification. Appropriate thresholds applied to the spatially varying object pixel density were used to separate components and nodes from connections and the rest of the image.

Some components, such as capacitors and grounds are split into two or more disjoint pieces. The segmentation algorithm must detect them as one segment. On the other hand, straight connections should not be merged with the segment of a component to which they connect. In circuit diagrams, components tend to be made up of closely spaced short lines and consequently the pixel-density over components is higher than in other parts. Pixel-density is ideally estimated over a small circular region such that there is no orientation dependent bias. However, for efficient computation a rectangular region may be used. Assuming a circular region of radius R pixels, bounds can be placed on the threshold required for segmentation.

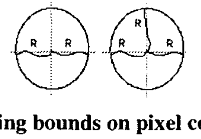

Figure 3. Estimating bounds on pixel count in a region of radius R pixels, from a line and a node.

Since the threshold, T, on the pixel count in the region must be greater than the number of pixels for lines,

$$T > 2R+1 \qquad (1)$$

As it must be less than the pixel count for a T-shaped node,

$$T < 3R+1 \qquad (2)$$

It is assumed in the above formulae that the centre pixel is not counted in the radius. Setting an appropriate threshold that satisfied these bounds would separate components and nodes from the rest of the image, as shown in figure 4.

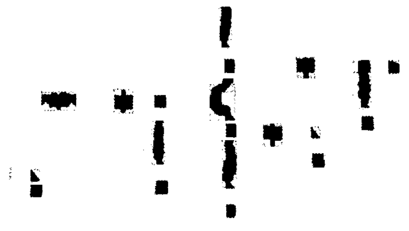

Figure 4. Image showing component and node segments.

The advantage of treating components and nodes alike at this stage is that connections will link no more than two objects. Therefore, connections can be defined with one start point and one end point.

Two errors have occurred in figure 4. The transistor has been split into two segments and the resistor on the right has merged with the node above it. Such errors are fixed in later stages.

5. CONNECTION TRACING

Connections need to be segmented with the following requirements:

- No maximum length assumed
- No minimum length assumed (other than to ports)
- Input and output ports must be recognised
- Parts of components protruding from component boundaries must be ignored, and
- Spurs (short line artefacts) must be eliminated.

Our tracing algorithm uses a stack to store pixels on the current path. The pseudo code for the algorithm is

```
1. Push starting point on to the stack
2. Insert starting point to the visited list
3. For all points that are neighbours of the starting point and
   inside the starting component - insert the point into the
   visited list.
4. While stopping condition is FALSE
Examine neighbours of the stack top pixel
If the neighbour is not in the visited list
        Push the pixel on the top of the stack
        Insert the pixel into the visited list
Else    Pop the top pixel off the stack
If stack is empty
        Set stopping condition to TRUE
        /* found a spur or port */
Else if the stack top pixel is in a new component
        Set stopping condition to TRUE
        /* found a connection */
Else    Set stopping condition to FALSE
        /* continue tracing */
```

When the loop stops with the stack empty, it is either because of a small extruding part of a component or because of an input or output port. To distinguish them the stack size is compared with a predetermined maximum size for spurs. Figure 5 shows the image after connection detection. No connections were missed in this case, and even the longest and shortest nodes were detected.

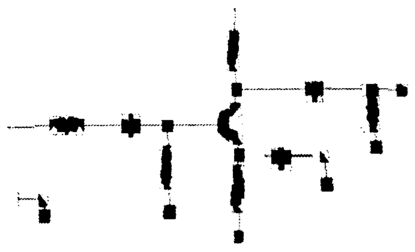

Figure 5. The image after connections are traced.

6. NODE RECOGNITION

The segmentation stage produces a set of unknown objects that consist of both components and nodes. Structurally nodes consist of lines meeting at a central point and can be recognised using a syntactic approach. Nodes can be any of the following types as shown in figure 6. In four-way connections the lines may not join at one point; instead, a small line of a few pixels may link two three-way connections. This line must not be confused with spurs.

Before an unknown component can be parsed to determine if it is a node, the bitmap representation of the component is approximated by straight lines. This polygonalisation step greatly reduces the number of primitive features. The Wall and Danielsson [4] algorithm was used. This method starts at a fixed pixel that is translated to the origin. A straight line is then approximated for pixels along the line until the error exceeds a threshold. The error used is the area deviation per unit length of the approximating segment. The end point is then used as the starting point for the next straight segment.

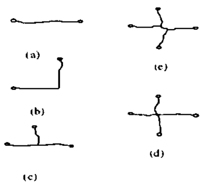

Figure 6. Types of nodes.

The connectivity of a pixel is two for a line, and greater than two for a connection. If the connectivity is greater than two, each branch is checked for spurs, which are assumed to be shorter than the smallest allowable branches to a port. After eliminating the spurs, if the connectivity is still greater than two, the pixel is a connection. The primitive features used for parsing nodes are: (a) contacts, *t*, (b) three-way connections, c_3, (c) four-way connections, c_4, and (d) line, *l*. Parsing starts at one of the terminals. If the unknown object (component or node) fits the structure of a node as defined by the grammar given below, it is recognised as a node. The tree grammar G = {V, R, P, S} where V = $\{NODE,\ T,\ L, C_3, C_3^T, C_4, t, l, c_3, c_4\}$ is the alphabet of non-terminals (in upper-case) and terminals (lower-case), R is the rank of each element in the alphabet, defined as the number of tree branches it may have, P is the set of productions, $T_i \rightarrow T_j$, where T_i, T_j are trees, as given in figure 7. S is the set of starting trees.

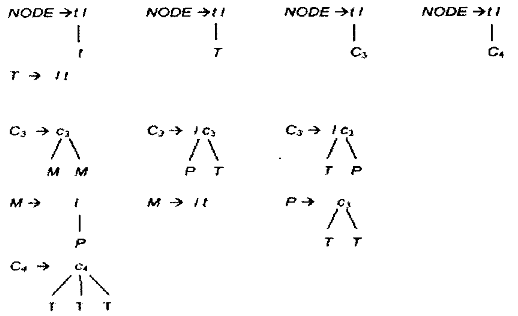

Figure 7. Productions in the node tree grammar

Starting productions have *NODE* on the left side. The rank of each element in the alphabet is evident from the number of branches from each production. This is a regular grammar and, therefore, the parser needs to look ahead by only one feature. The number of productions is rather large because nodes can follow other nodes without components in between.

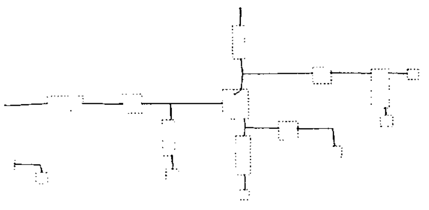

Figure 8. The image after nodes are identified

Figure 8 shows the common emitter amplifier circuit after nodes are identified. All three-way junctions and corners are now correctly identified as nodes. An interesting observation is that the small-unknown object in the corner of the transistor that arose as an artifact of the segmentation process was identified as a corner node, and will not introduce any error in the interpretation of the circuit.

7. COMPONENT RECOGNITION

The component recognition stage uses a statistical classifier [5]. This work was restricted to recognition of the 9 components given in the figure below.

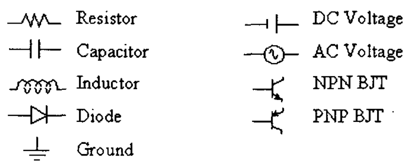

Figure 9. The set of hand-drawn components recognised

A set of 35 features was extracted from thinned images of the unknown components. Six features were Hu moment invariants [6], eighteen were geometric measurements (aspect ratio, rectangularity, and 8 element histogram vectors along the two axes), and twelve features were extracted from polygonal or straight segment approximations [4] as shown in figure 10. Vector features for the components in figure 10 are shown in figure 11.

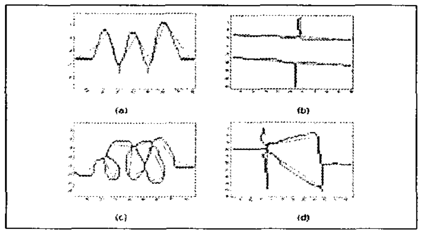

Figure 10. Some polygonal representations

The samples in this work were all taken from one person but they exhibited a range of variations in orientation, scale and shape. Log-Moment and vector features were scaled such that their dynamic ranges were roughly in the range -1 to 1. Geometric features were defined to be between 0 and 1.

8. RECOGNITION ACCURACY

Success rates were determined for both node recognition and component recognition. Out of 107 test images of hand-drawn nodes, 99 were recognised correctly, roughly 92.5% accurate. Breaks in lines that escape the morphological pre-processing are the major source of error in node recognition.

A nearest neighbour classifier [5] was trained on features obtained from 349 of total 449 hand-drawn components. The remaining 100 were used for testing. Five random partitions of the set were used to obtain averaged test results, shown in the table below. Overall 86% of the components were recognised correctly.

The table shows that classification accuracy is very high for resistors and inductors. After segmentation, differences in plate sizes may not be significant resulting in misclassification between capacitors and voltage sources.

Vector Features	R	C	L	D
Horizontal lines	3	4	8	6
Forward diagonals	3	2	7	1
Vertical lines	0	0	2	3
Backward diagonals	3	0	5	1
Acute angles	0	0	1	0
Right angles	4	0	7	1
Obtuse angles	2	0	1	0
Connections	1	2	8	5
Contacts	2	2	2	2
End points	1	4	0	3
Loops	0	0	4	1

Figure 11. Vector features for components in figure 10

		Recognized Class								
		R	C	L	D	Gnd	Vdc	Vac	NPN	PNP
Actual Class	R	100.0%	0	0	0	0	0	0	0	100%
	C	0	53.8%	0	0	0	46.2%	0	0	0
	L	0	0	100.0%	0	0	0	0	0	0
	D	0	10.0%	0	90.0%	0	0	0	0	0
	Gnd	0	0	0	0	100.0%	0	0	0	0
	Vdc	0	22.2%	0	0	0	77.8%	0	0	0
	Vac	0	9.1%	0	9.1%	0	0	81.8%	0	0
	NPN	0	0	0	0	0	0	0	82.3%	17.6%
	PNP	0	0	0	0	0	0	0	0	100%

Table 1. Confusion matrix for component recognition

Diodes and *AC* sources are similarly prone to misclassification because they both contain a single loop. No PNP transistors were misclassified as NPN because the classifier defaults to PNP if an arrow cannot be detected at the emitter.

9. CONCLUSION

A methodology for recognition of hand-drawn circuit diagrams by computer image analysis is presented. With further refinement, it can be used to "beautify" hand-drawn diagrams for documentation, or to generate code for circuit analysis.

10. REFERENCES.

[1] L. O'Gorman & R. Kasturi, *Document Image Analysis*, IEEE Computer Society Press, Los Alamitos, 1995.

[2] Tsai, W-H, "Moment-Preserving Thresholding: A New Approach", Computer Vision, Graphics, and Image Processing, vol.29, pp.377-93, Academic Press, 1985.

[3] T.Y. Zhang & C.Y. Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns," *Comm. ACM*, vol. 27, no. 3, pp. 236-239, 1984.

[4] K. Wall & P.E. Danielsson, "A Fast Sequential Method for Polygonal Approximation of Digitized Curves", *Computer Vision, Graphics and Image Processing*, vol. 28, pp. 220-227, Academic Press, 1984.

[5] R. Schalkoff, Pattern Recognition: Statistical, Structural, and Neural Approaches, Wiley, New York, 1992.

[6] M.K. Hu, "Visual Pattern Recognition by Moment Invariants," IRE Trans. Inform. Theory, pp. 179-187, Feb. 1962.