

## SYMBOL SPOTTING IN ELECTRONIC IMAGES USING MORPHOLOGICAL FILTERS AND HOUGH TRANSFORM

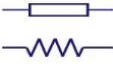

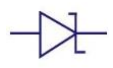
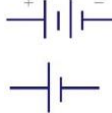

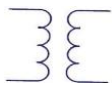
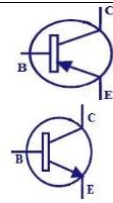

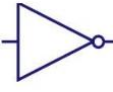
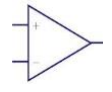
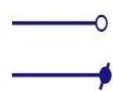
Dilovan S. Ramadhan<sup>a,\*</sup>, Hasan S. M. Al-Khaffaf<sup>b</sup><sup>a</sup> College of Basic Education, University of Duhok, Duhok, Kurdistan Region, Iraq – dilovan.salah@uod.ac<sup>b</sup> Dept. of Computer Sciences, University of Duhok, Duhok, Kurdistan Region, Iraq – hasan.salim@uod.ac**Received:** 29 Dec., 2021 / **Accepted:** 20 Jun., 2022 / **Published:** 24 Aug., 2022 <https://doi.org/10.25271/sjuoz.2022.10.3.874>**ABSTRACT:**

In this paper, two algorithms (a preliminary and enhanced algorithm) to detect electronic symbols in document images are proposed. Morphological operations coupled with Hough transform are used in the proposed methodology. The objective of the proposed algorithms is to detect electronic symbols of open and closed shapes. The methods can successfully spot many complex types of electronic symbols such as Fixed Resistor; Zener Diode; Pn Junction Diode; NPN transistor; Not Gate; Input and Output Terminals; Ground; Single Cell Battery; Transformer and LED symbols. The experimental results on the Systems Evaluation Synthetic Documents (SESYD) dataset show that the proposed preliminary method detects 86.2%, (12151 symbols out of 14100 symbols), precision of 0.94, recall of 0.91, and F-measure of 0.92. An enhanced algorithm that used line Hough transform is also demonstrated with accuracy of 91.2% (12864 symbols out of 14100 symbols), precision of 0.97, recall of 0.93, and F-measure of 0.95.

**KEYWORDS:** Electronic symbols, Hough Transform, Symbol spotting, Morphological filters.**1. INTRODUCTION**

Symbol spotting is a field of document image analysis and it is the process of locating symbols in document images. One application of symbol spotting is document retrieval in large databases of documents. With symbol spotting, automatic search can reduce search time and effort because the recognition stage needs to process only areas-of-interest of the original image. The idea behind this process is to find the exact position of the symbol in a document and it is the backbone for the recognition and analysis process. In the past years, researchers worked on several algorithms to spot symbols in the document image. While computerized recognition of printed and handwritten text has been one of the most researched pattern classification problems (Moetesum, 2018). The symbol spotting problem is very challenging when it comes to machine recognition of these sketches especially when they are in a context (i.e., not isolated). In general, symbol spotting methods consist of many phases: description, matching/locating, and performance evaluation phases (Rezvanifar, 2019). The description phase might be pixel or vector-based where the pixel-based works directly on the pixel data of the image while the vector approach needs to convert the image to vectorial formats such as lines and curves. Each of these approaches has its advantages and disadvantages. In this work, we opted for the pixel-based approach due to its simplicity and ability to achieve a good spotting ratio with the chosen dataset. Additionally, this work will focus on electronic symbols. There are many electronic symbols with different morphological complexity (Table 1). Some symbols like Ground and Cell Battery are simple in shape and consist of only straight lines while other shapes like Led, Zener diode, and Transformer are formed with lines and curves and create closed shapes. The difference in shapes requires careful consideration for the spotting algorithm to avoid the removal of symbols. The challenge is to propose an algorithm that spots all of these symbols.

Table 1: Electronic symbols detected by proposed algorithms of this study.

Symbol	Symbol Name	Symbol	Symbol Name
	Fixed Resistor		Ground
	Zener Diode		Single Cell Battery
	Pn Junction Diode		Transformer
	NPN transistor		LED
	Not Gate		Operational Amplifier
	Input and Output Terminals		

Symbol spotting of electronic symbols was studied by many researchers. Moetesum (2018) used morphological operations followed by thinning to spot symbols. Their method achieved

\* Corresponding author

This is an open access under a CC BY-NC-SA 4.0 license (<https://creativecommons.org/licenses/by-nc-sa/4.0/>)

87.7% segmentation accuracy. Edwards *et al.* (Edwards & Chandran, 2000) also used morphological operations to spot electronic symbols. They achieved an average of 89% recognition accuracy on the dataset.

In this work, a morphological based algorithm (preliminary algorithm) coupled with circular Hough transform is proposed to find and locate the electronic symbols in document images. An enhanced algorithm that uses linear Hough transform to improve symbol spotting is also proposed. The steps are carefully sequenced to spot symbols of different complexity. The Systems Evaluation Synthetic Documents (SESYD) dataset that is readily available to the researchers will be used in this study. The synthetic dataset has 1000 images of different diagrams. The experimental results show that the proposed work efficiently spots symbols in the test images.

In Section 2, the related works are explained. The methodology of the preliminary algorithm is explained in Section 3. while the methodology of the enhanced algorithm is explained in Section 4. The dataset is explained in Section 5. The experimental setup and the results are explained in Sections 6 and 7. The discussion of the results is shown in Section 8. Finally, section 9 presents the main conclusions arrived at.

## 2. RELATED WORKS

Symbol spotting in document images is studied and many methods have been presented. A summary of some of the papers from the literature is presented in the following paragraphs.

Müller and Rigoll (2000) presented one of the first methods of symbol spotting. It used segmented technical drawing in a small cell. A two-dimensional Hidden Markov Model (HMM) training algorithm is used to find a symbol that matches the model database. In their work, isolated symbols, as well as cluttered symbols, can be spotted. Furthermore, in Tabbone (2004), after extracting the skeleton from a document image the process of segmenting symbols starts by finding the junction points of the skeleton image by the loop extraction process. After this segmentation process, each segment is composed of a set of pixel features. Features are extracted from model symbols. Other techniques, as in Qureshi (2008), used graph-based representation to spot symbols in the document image. The main idea in this process is to use the structural definition of graphical symbols. The method used a single step to locate and recognize graphical symbols by isomorphism techniques. However, this process does not work properly when applied to a big dataset. Another work by Zhang and Liu (2007) focused on symbol signatures which are computed in regions of interest in document images. The main drawback of this process is that it always assumes that the symbols can be located in regions of interest and noise or occlusions can affect symbol signatures. Besides, Zuwala and Tabbone (2006) used a hierarchical definition of the symbols to find symbols in documents. The main idea is to use a dendrogram structure to make a hierarchical tree of symbols. A symbol is divided into small pieces (subparts) split at the junction points. A measure of density is used to merge subparts to build the dendrogram structure. Each subpart is described by a shape descriptor. The dendrogram can be subsequently traversed to retrieve the regions of interest of a line drawing where the queried symbol is likely to appear. Najman (2001) presented a method of symbol spotting in technical documents. In the method, the focus on the text contained in the symbols helps to extract the graphical areas annotated by these text strings. Rezvanifar (2021) presented a hybrid method that combines concepts from both vector-based and

pixel-based symbol spotting techniques. In the description phase, the prominent features of a symbol are extracted by different vectorization techniques, including a voting-based algorithm for finding partial ellipses. This enables the algorithm to better handle local shape irregularities and boundary discontinuities, as well as partial occlusion and overlap. The matching phase relies on the relationship between the primitives through a primitive-aware graph. Fan (2021) introduced a novel method that combines Graph Convolutional Networks (GCNs) with Convolutional Neural Networks (CNNs) and which captures both non-Euclidean and Euclidean features. Rezvanifar (2020) presented a symbol spotting method of real-world digital architectural floor plans with deep learning (DL)-based framework. The method works by adapting an object detection framework based on the You-Only-Look-Once (YOLO) architecture. In KaiBin and Hooi (2021) a machine learning feature is used to retrieve more precise symbols. Rezvanifar (2019) thoroughly reviewed symbol spotting methods focusing on architectural drawings.

## 3. METHODOLOGY (PRELIMINARY ALGORITHM)

Symbol spotting in document images is a sequential process to detect the location of symbols. It goes through different steps that help to achieve the relevant result with minimum steps and with a good accuracy rate. So, an algorithm named preliminary algorithm is designed to get the best results with minimum and simple operations that is shown in the flowchart of Fig. 1.

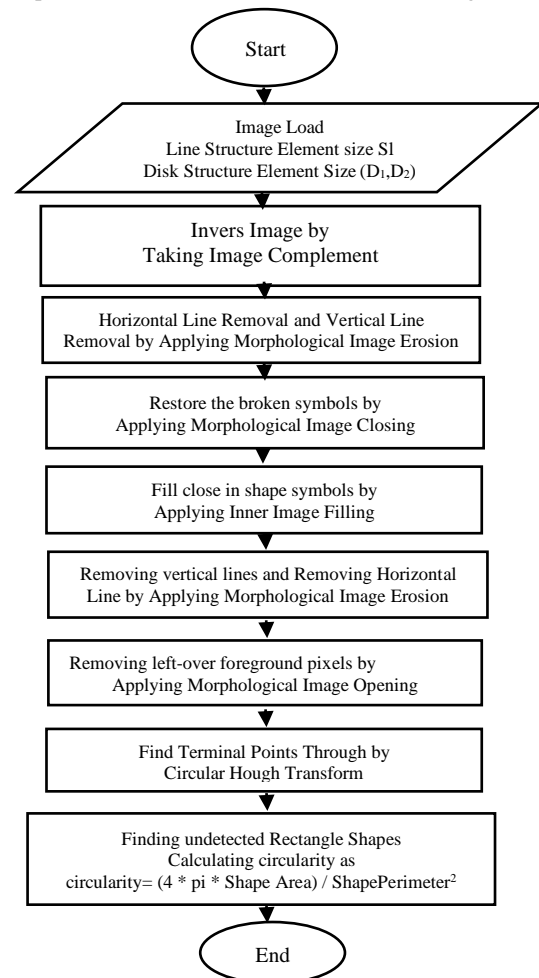


Figure 1: The diagram of the proposed preliminary symbol spotting algorithm

### 3.1 Background

Morphological image processing is a combination of nonlinear processes that deal with the shape or morphology of image features. Only the relative ordering of pixels is used in morphological operations, as a result, they are particularly well adapted to the processing of binary images. Morphological approaches use a small form or pattern called a structuring element (SE) to explore an image. The SE is placed in all conceivable positions in the image and compared to the pixels in the surrounding neighbourhoods. Some operations determine if an element "fits" into the neighbourhoods, while others determine whether it "hits" or intersects it (Moetesum, 2018).

### 3.2 Image Pre-processing

Image is loaded and the polarity is flipped (if necessary) by taking image complement so that the foreground pixels are set to 1 while the background pixels are set to 0. Figure 2 shows an image as an example.

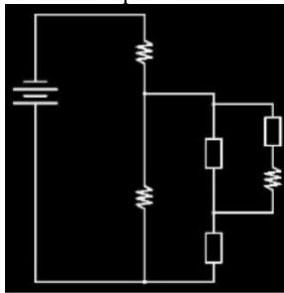


Figure 2: Sample image of an electrical circuit.

### 3.3 Horizontal and Vertical Line Removal

In this step, the horizontal and vertical line layers are removed from the document image. This process is accomplished by using erosion morphological image processing defined below:

Let  $E$  be an integer grid or a Euclidean space

$$A \ominus B = \{z \in E | B_z \subseteq A\} \tag{1}$$

where

- $A$ : binary image in  $E$
- $B$ : is a structural element with a 'line' parameter
- $B_z$ : is the translation of  $B$  by the vector  $z$

An SE is a pattern used to probe or interact with a given image. The objective is to withdraw inferences about how this form fits or misses the shapes in the image. Dilation, erosion, opening, and closing are some of the morphological procedures.

The SE has a specific shape. A "ball" or a line; a convex or a ring, for example, can be used as an SE (Fig. 3). By selecting a SE, one establishes a method for distinguishing some items (or sections of objects) from others based on their shape or spatial orientation. The SE has a specific size, for instance, a 3\*3 square or a 7\*7 square. Establishing the size of the SE is comparable to setting the observation scale and criterion for separating picture objects or features based on their size.

We experimented with different sizes for  $S_i$  (Fig. 1) from 10 to 100 and the size 50 was the most appropriate size to detect horizontal and vertical lines in the electronic document diagram in the SESYD dataset. For removing unwanted shapes, a disk SE of sizes  $D_2= 15$  and  $D_1= 30$  is used to restore the broken symbols. These two sizes were the best value parameters to restore and remove symbols using image closing and morphological erosion operations.

Disk shape	Square	Vertical rectangular and Horizontal rectangular
Horizontal line	Irregular	Vertical line

Figure 3: Samples of different SEs.

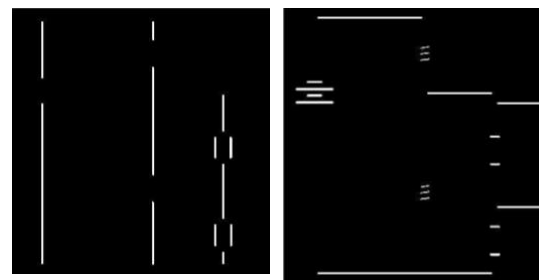


Figure 4: Horizontal and Vertical layer line extraction. (a) Horizontal layer. (b) Vertical layer.

To detect symbols closed in shape (like resistor), the inner region filling is performed but that cannot be performed immediately because loop lines are connecting symbols. The idea is about extracting Horizontal Layer (HL) and Vertical Layer (VL) as shown in Fig. 4. Next, one layer (firstly HL) is eliminated from the image by subtracting it from the original image and continues the process to break the line loop connecting symbols in the document. Finally, the VL is subtracted from the image and the whole process is repeated with the HL layer (Fig. 5).

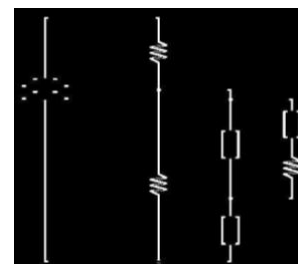


Figure 5: Removing horizontal layer from the original image of Fig. 2.

### 3.4 Restore Broken Symbols

The process of image erosion may break some symbols or damage their shapes. Image closing operation is performed to restore the broken symbols (if any) as shown in Fig. 6. The process needs two images (A and B). Image A is the original image of Fig. 1 minus HL while image B is the disk SE of size 30.

$$A \cdot B = (A \oplus B) \ominus B \quad (2)$$

where

- A: binary image in E
- B: structure element
- $\circlearrowleft$ : closing operator
- $\ominus$ : erosion operator
- $\oplus$ : dilation operator

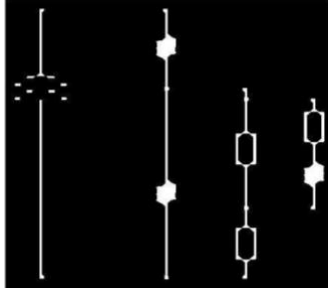


Figure 6: Restoring broken symbols through closing.

### 3.5 Inner Region Filling

After performing image closing in the previous section, the image contains, almost, closed shapes symbols. Inner region filling can be performed next to fill the entire symbol. A hole is defined as an area of dark pixels surrounded by lighter pixels.

Assume that A is a binary image and that the marker image,  $f(x,y)$ , is set to 0 everywhere except on the image border, where it is set to  $1-A$ .

$$f(x,y) = \begin{cases} 1 - A(x,y) & \text{if } f(x,y) \text{ lies on the border of } A \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

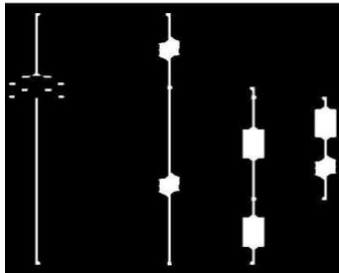
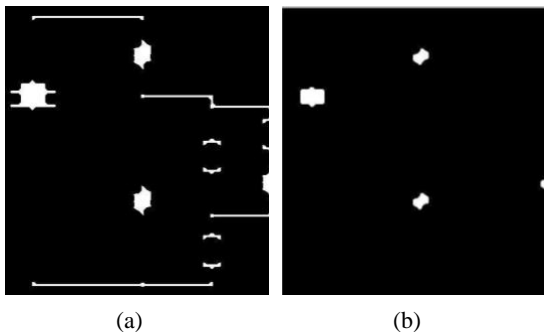


Figure 7: Image of Fig. 6 after inner region filling.

### 3.6 Removing Vertical Lines



The next step is to perform image erosion to remove vertical lines and to get only symbols locations (Fig. 8).

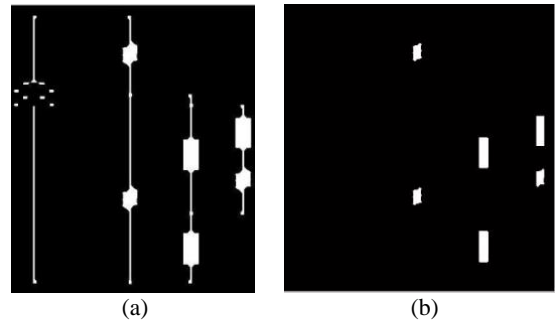


Figure 8:(a) Extracting symbols by removing HL (b) Detecting symbol location through erosion.

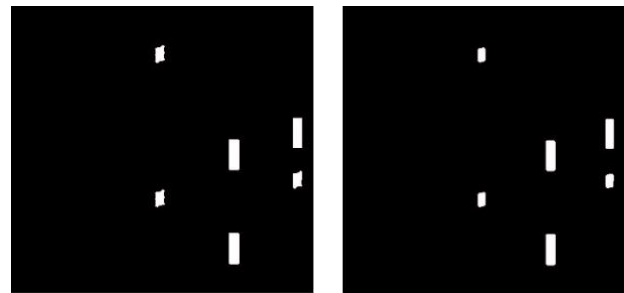
### 3.7 Removing Left-over Foreground Pixels

In the previous step, only the symbols' locations were detected. Some images might have graphical objects that are not electronic symbols. In this work, if the shape is smaller than a certain size, that shape must be eliminated from the image and this is performed by using image opening.

$$A \circ B = (A \ominus B) \oplus B \quad (4)$$

where

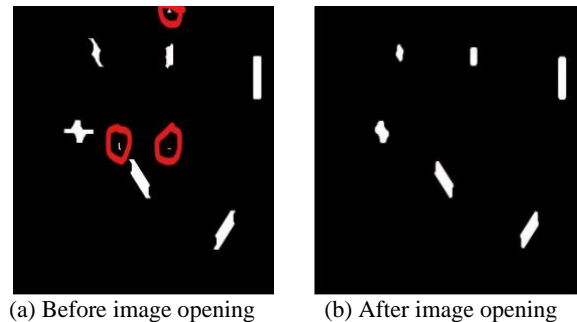
- A: binary image in E
- B: structure element
- $\circ$ : opening operator
- $\ominus$ : erosion operator
- $\oplus$ : dilation operator



(a) Before image opening (b) After image opening

Figure 9: Removing left-over foreground pixels. All elements are large enough to avoid removal (No elements were removed).

In Fig. 9 above, the left-over foreground pixels are not found while they were found and removed in Fig. 10.



(a) Before image opening (b) After image opening

Figure 10: Removing left-over foreground pixels. Elements in (a) to be removed are circled in red.

After performing the previous process on HL, the same processes from sections (3.2-3.6) are repeated but this time relying on VL, not HL (Fig. 11). The reason behind this is to detect the symbols that are missed previously.

Figure 11: (a) Extracting symbols by removing VL (b) Detecting symbol location

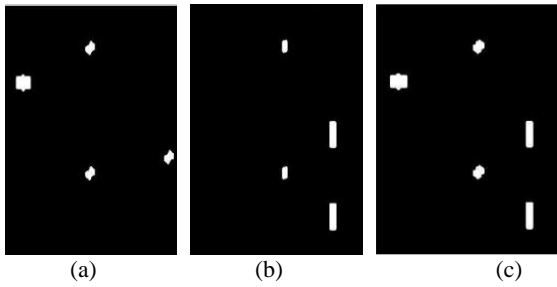


Figure 12: (a) Symbols location based on VL (b) Symbols location based on HL (c) Merging of two images (a) and (b).

The next stage is to locate the shape's placement in a black and white image by measuring a range of image characteristics and attributes. The 8-connected components (CC) from a black and white image are used to locate symbols. For each CC, the width and height are found. When drawing the red bounding box, the box is enlarged by a constant value so that the box surrounds the symbol.

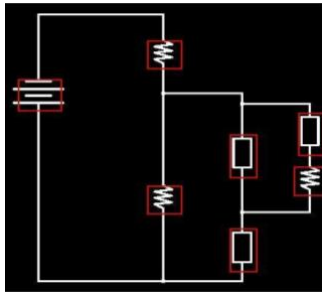


Figure 13: Detected symbols surrounded by a red box.

### 3.8 Detecting Terminal Lines

The previous algorithm steps did not detect the input and output terminals. This weakness can be solved using Circle Hough transform (CHT).

The CHT is a basic feature extraction technique used in digital image processing for detecting circles in imperfect images. The circle candidates are produced by "voting" in the Hough parameter space and then selecting local maxima in an accumulator matrix. Three pieces of information: the radius, the centre (xc, yc), and the circumference describe a circle. A full circle of radius r is created when  $\theta$  ranges from 0 to 360 degrees.

With the CHT, it is anticipated to identify triplets of (x, y, r) in the picture that almost certainly defines a circle. To put it another way, we are looking for the three parameters (x, y, r). We settled on  $r=30$  for this dataset after testing with multiple values for r (ranging from 10 to 50 and incrementing by 5). At the beginning, the assumption is to seek for circles with a specific radius, r, and that r is known. The circle equation is as follows:

$$x = xc + r * \cos \theta \tag{5}$$

$$y = yc + r * \sin \theta \tag{6}$$

As a result, any point in (x,y) space will be identical to a circle in (xc, yc) space (r isn't a parameter because it is already known). This is because when the equations are rearranged, the equations shown as below.

$$xc = x1 - r * \cos \theta \tag{7}$$

$$yc = y1 - r * \sin \theta \tag{8}$$

For a particular point (x1, y1) and  $\theta$  sweeps from 0 to 360 degrees.

The steps of the Hough transform are as follows:

1. Load the image
2. Perform edge detection

3. Use 'edge' pixel to generate a circle in the xy space
4. For every point on the circle in the xy space, update 'votes' in the accumulator cells
5. The cells with the maximum number of votes are the valid centres

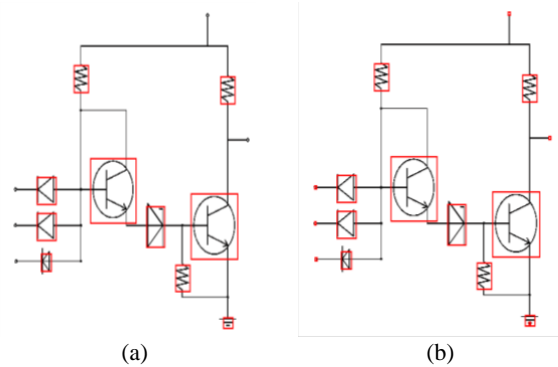


Figure 14: Detecting endpoints (a) Before Circle Hough transform (b) After Circle Hough transform.

### 3.9 Finding Undetected Rectangle Shapes

To improve the detection rate, it was found that in some images the previous algorithm steps did not detect the rectangular-shaped symbols. A process that can detect rectangular-shaped symbols is added. Firstly, image interior is filled and secondly, the original image is subtracted from it (Fig. 15). Next, the connected component is applied to the output from the previous one. A group of graphical objects should be isolated from one another in the binary image. Pixels that belong to an object are labelled with 1 or true, whereas pixels in the background are labelled with 0 or false.

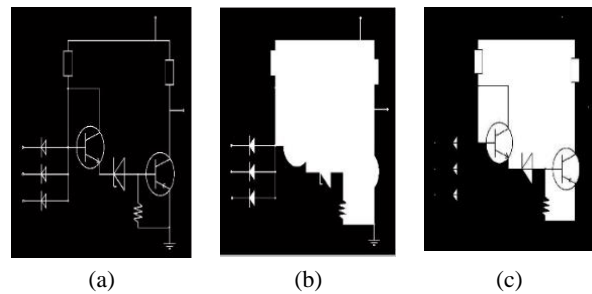


Figure 15: (a) Original image. (b) Inner-filled image. (c) Subtracting original image (a) from inner filling image (b).

The area of a circle is  $\pi * r^2$  and the perimeter is  $2 * \pi * r$ , where r is the radius. The value of a suitable measure of circularity compares these two in such a manner that it is independent of the size of the shape or the units used to measure it. It will also be clearer that if an object's circularity value is equal to 1, then the shape is a circle. If the circularity is less than 1, then it is other shapes (i.e., less circular or compact). The measure of circularity is given by

$$C = \frac{4 * \pi * A}{P^2} \tag{9}$$

$$\approx 12.57 * \frac{A}{P^2}$$

where

- C is the circularity
- A is the area
- P is the perimeter

For the SESYD dataset, it was experimentally found that for rectangles the value of the circularity lies in the range of [1.3, 1.35].

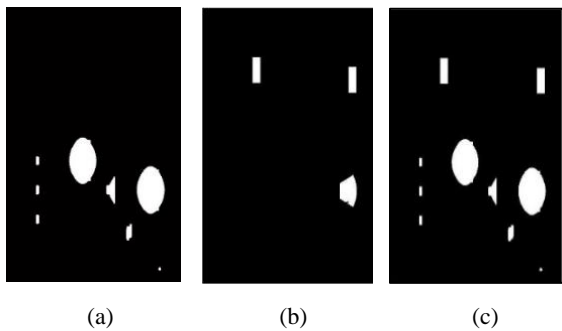


Figure 16: (a) Symbols detected. (b) Detecting rectangular symbols. (c) Merging of (a) and (b).

#### 4. METHODOLOGY (ENHANCED ALGORITHM)

The preliminary algorithm detects most of the symbols during the experiment (see Section 7). However, it cannot detect some symbols that consist of horizontal and vertical lines like battery and capacitors. Hence, the preliminary algorithm is further improved by replacing erosion step with linear Hough Transform to detect such symbols (Fig. 17). Additionally, the enhanced algorithm incorporates two iterations of symbol spotting. In the first iteration, closed symbols are detected and removed from the image. In the second iteration, the rest of the symbols were spotted. The enhanced algorithm has some differences to the preliminary algorithm. The newly added steps are explained in the next section

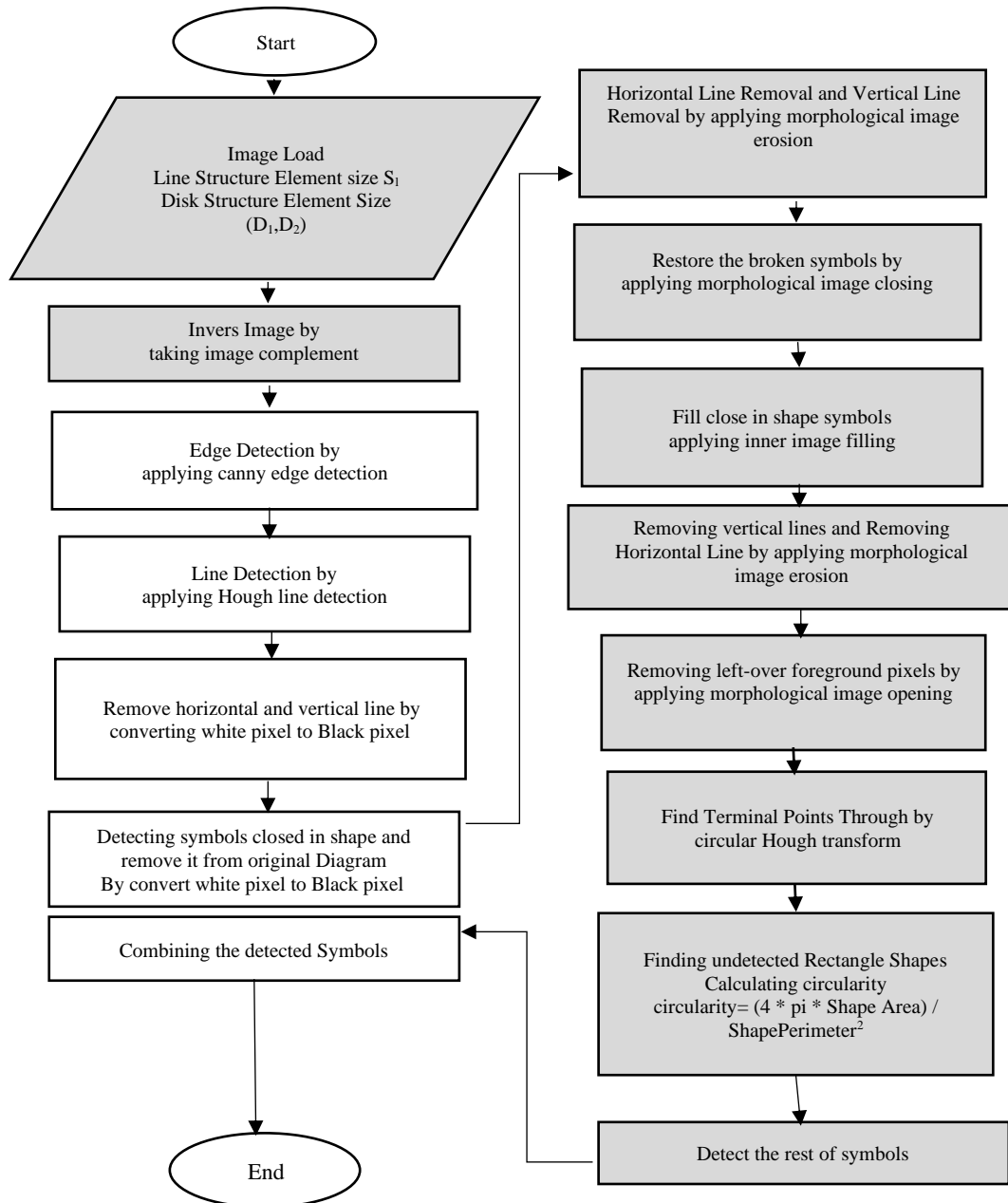
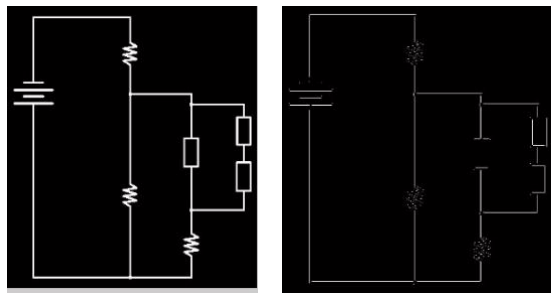


Figure 17: Flowchart of the enhanced algorithm which includes old steps of preliminary algorithm shown in grey background.

#### 4.1 Edge Detection

In order to implement the Hough line transform to detect the horizontal and vertical lines (wires), line thickness of graphical elements must be one-pixel for the Hough line to work properly. In this case, Canny edge detection algorithm is used (Fig. 18).



(a) Input diagram (b) Canny edge detection  
Figure 18: Canny edge detection produces one-pixel-wide skeletons.

#### 4.2 Linear Hough Transform

After the application of Canny edge detection to get one-pixel-wide skeleton then the diagram goes through Hough line transform algorithm to detect the horizontal and vertical lines (wires) as showed in Fig. 19.

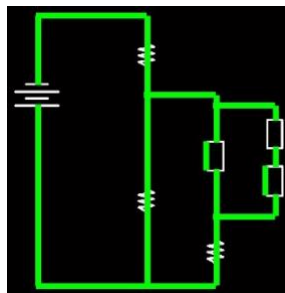
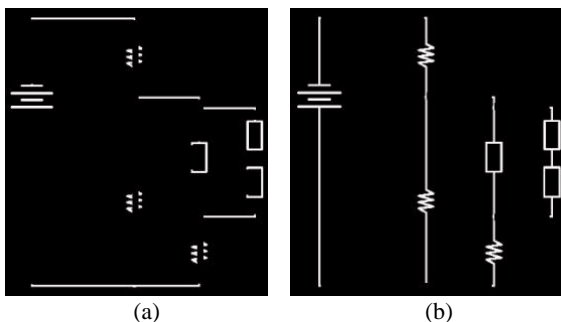


Figure 19: Applying Hough line transform on a sample image.

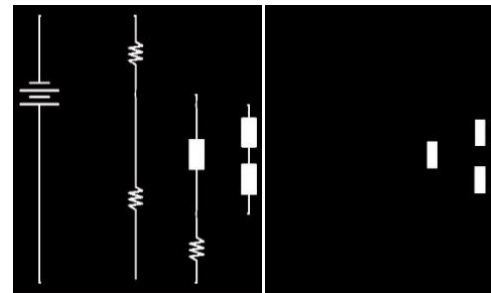
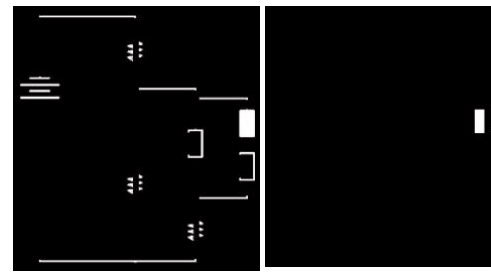
After line detection by Hough line transform, horizontal and vertical lines were removed as a two-layer H-layer and V-layer as in Fig. 20.



(a) Removing vertical line (b) Removing horizontal line  
Figure 20: Horizontal and vertical line removal. (a) Removing vertical line (b) Removing horizontal line

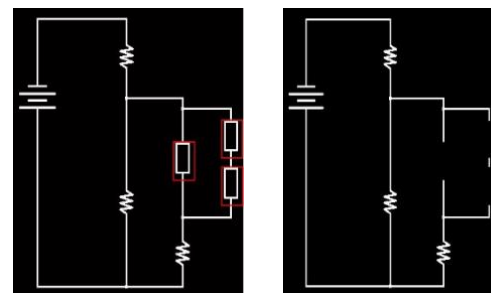
#### 4.3 Adding extra iteration for symbol spotting

In the first iteration, symbols that form closed shapes were detected by using inner fill algorithm. Then the original image is removed from it as in Fig. 21.



(a) Inner fill algorithm (b) Remove line from the diagram  
Figure 21: (a) Inner fill algorithm (b) Remove line from the diagram

Then those symbols were detected and removed from original diagram.



(a) Detected symbols. (b) After removing detected symbols from the diagram.  
Figure 22: (a) Detected symbols. (b) After removing detected symbols from the diagram.

The resulted diagram in the previous process is the input for the next iteration. In the second iteration the process is the same as in sections 3.2-3.7.

### 5. DATASET

The dataset used in this paper is SESYD which is a database of synthetic documents with the corresponding ground truth produced using the 3gT system (Delalandre, 2016). This database is composed of many collections (symbol bags, floor plans, electronic diagrams, International Symbol Recognition Contest 2011 dataset, lowers-floorplans, sketched symbols, font character, segment characters, word bags, and text/graphics) and it is available as a public domain dataset (Delalandre, 2012). In this paper, the focus is on electronic diagrams' part of the dataset which consists of 1000 diagrams of 14100 symbols divided into 10 folders each consisting of 100 diagrams of binary TIFF image file format (Table 2). Additionally, each TIFF image file is accompanied by corresponding Generic Object Model (GOM) and Extensible Markup Language (XML) documents that provide more information on symbols, the X and Y coordinates and the size of each symbol. In each folder of the dataset, the electronic diagram wire connection is the same while the location of the symbols is swapped with other symbols. In other words, each diagram wire connection is similar but electronic symbols are swapped among 100 diagrams.

Table 2: The SESYD Dataset. There are 100 binary TIFF images per folder.

Folder	Name	Resolution	#Symbols
#01	diagrams21-01	1720*1760	700
#02	diagrams21-02	2400*1333	900
#03	diagrams21-03	1740*1200	800
#04	diagrams21-04	3998*3600	1500
#05	diagrams21-05	2981*1463	1300
#06	diagrams21-06	1688*1727	1400
#07	diagrams21-07	4198*1871	1900
#08	diagrams21-08	2100*2075	1600
#09	diagrams21-09	4433*2100	2600
#10	diagrams21-10	3628*1500	1400

### 6. Experimental Setup

In this research, MATLAB is used as the preferred programming language because it contains a huge library of functions that helps in reducing time for coding. Due to the big number of samples (images) in the dataset, it is divided into 10 folders each of which contain 100 diagrams (images). Our code is prepared to read and process images in batches. The developed code processes each folder one by one and each diagram is processed and the extracted statistical data (detected symbols and missed symbols rates) are saved in a two-dimensional matrix. After processing all of the 1000 diagrams, the number of detection and miss detection of symbols is counted and saved into an excel file.

To evaluate the results, Precision, Recall, and F-measure are used as the performance evaluation criteria.

$$Precision = \frac{TP}{TP+FP} \tag{10}$$

$$Recall = \frac{TP}{TP+FN} \tag{11}$$

$$F - measure = \frac{2TP}{2TP+FP+FN} \tag{12}$$

where

Precision: Is the retrieved relevant documents

Recall: Is the relevant documents that are successfully retrieved

F-Measure: Is the harmonic mean of the precision and recall, is a measure of a test's accuracy

### 7. Experimental Results

The proposed preliminary algorithm detects 12151 symbols out of 14100 symbols, that is 86.2% accuracy (Table 3). While the enhanced algorithm detects 12864 symbols out of 14100 symbols, that is 91.2% accuracy. Figure 23 shows the number of detection and miss detection of symbols in each diagram of each folder dataset.

Preliminary Algorithm

$$Precision = 12151 / (12151 + 761) = 0.94$$

$$Recall = 12151 / (12151 + 1188) = 0.91$$

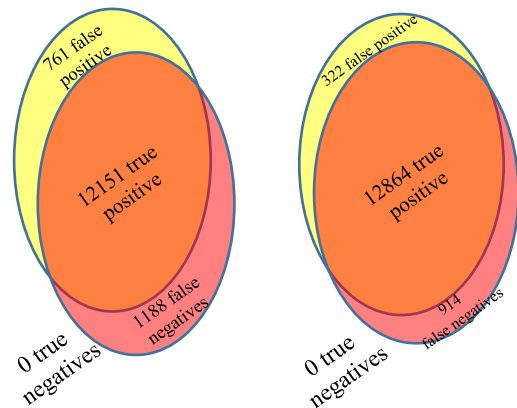
$$F - measure = 2 * 12151 / (2 * 12151 + 761 + 1188) = 0.92$$

Enhanced Algorithm

$$Precision = 12864 / (12864 + 322) = 0.97$$

$$Recall = 12864 / (12864 + 914) = 0.93$$

$$F - measure = 2 * 12864 / (2 * 12864 + 322 + 914) = 0.95$$



(a) Preliminary Algorithm (b) Enhanced Algorithm

Figure 23: Results of the proposed algorithm on the SESYD dataset.

Table 3: Statistical results

Dataset / 100	# Symbols per folder	Preliminary Algorithm			Enhanced Algorithm		
		# Detections	# False Detection	# Missed Detection	# Detections	# False Detection	# Missed Detection
#01	700	700	0	0	700	0	0
#02	900	849	0	51	900	0	0
#03	800	651	149	0	692	60	48
#04	1500	1400	100	0	1500	0	0
#05	1300	643	512	145	1030	103	167
#06	1400	1400	0	0	1400	0	0
#07	1900	1402	0	498	1507	159	234
#08	1600	1600	0	0	1600	0	0
#09	2600	2357	0	243	2243	0	357
#10	1400	1149	0	251	1292	0	108
<b>Total</b>	<b>14100</b>	<b>12151</b>	<b>761</b>	<b>1188</b>	<b>12864</b>	<b>322</b>	<b>914</b>



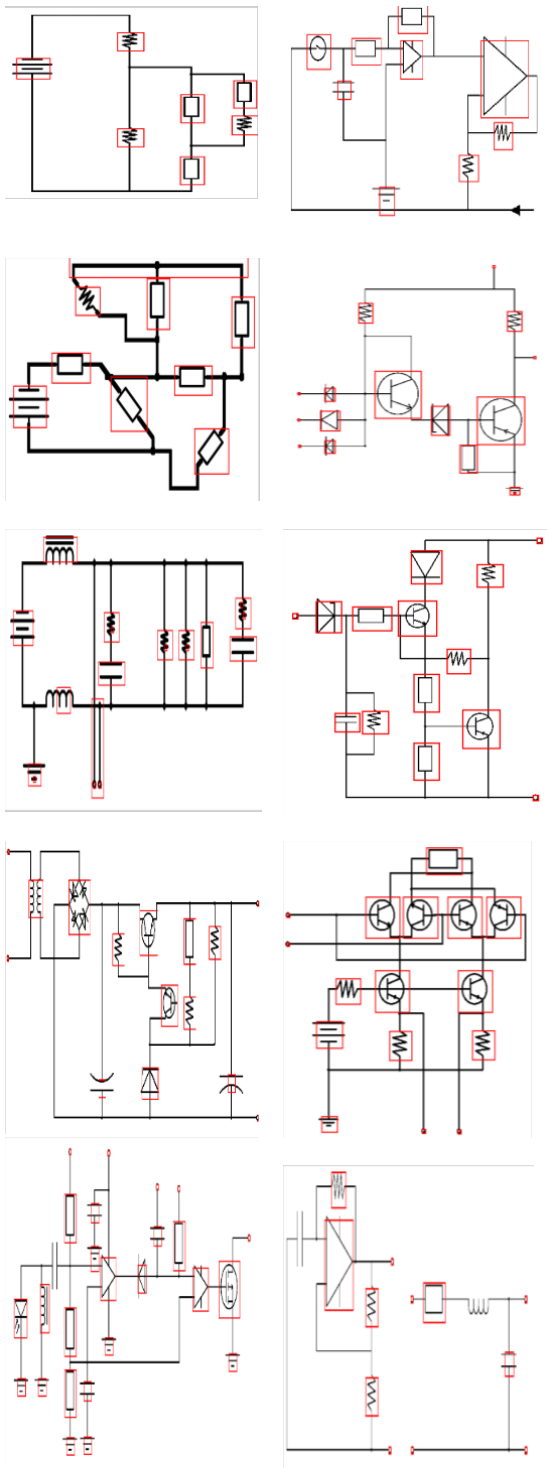


Figure 24: Sample images of preliminary algorithm with detected symbols wrapped in red rectangles.

### 8. Discussion

In our research and after performing the proposed preliminary algorithm on the dataset, the accuracy is 86.2% that is 12151 out of 14100 symbols are detected, which is a good result for our algorithm in that it can detect symbols without any complex calculations. The Precision ratio is 0.94 which indicates that the system retrieved most of the symbols. A Recall of 0.91 is the proportion of real symbols that are correctly spotted. The F-Measure is an overall measure of a model's accuracy that combines precision and recall. An F-Measure value of 0.92 means that the results has low false positives and low false negatives. The algorithm

can detect symbols of closed shapes for example (Fixed Resistor, Zener Diode, Pn Junction Diode, NPN transistor, and Not Gate). On the other hand, the algorithm cannot detect some symbols (Non-Polarized Capacitor and Inductors of Table 4) for the following reasons. First of all, there is no closed shape in these symbols which complicates the detection process. Secondly after performing morphological image closing the process is not enough for getting some thickness of the shape to be detected in the next process. The other issue is miss detection in some diagrams that are being shown in Fig. 25 due to a non-optimal SE size.

Table 4: Symbols not detected by preliminary method.

Symbol	Symbol Name
	Non-Polarized Capacitor
	Inductors

The accuracy of the preliminary algorithm shows that 86.2% of symbols were detected. In order to improve the detection accuracy, an enhanced algorithm of two iterations is proposed. In the first iteration, closed shape symbols were detected and in the second iteration those symbols are removed from the original diagram and made it an input image to the second iteration. In the second iteration, the same processes in sections 3.2-3.7 are applied. In comparison to preliminary algorithm, it was noticed in Fig. 26 that symbols are not recognized from preliminary algorithm but are detected by enhanced algorithm. This is due to the size of disk SE for morphological image closing that is used in both algorithms. The image closing in both algorithms is used to restore the broken symbols. In the preliminary method, a value of 30 for SE is used but it was not enough to restore the broken symbols. In the enhanced algorithm, an increased size of SE to 40, the broken symbols are restored and detected. The enhanced algorithm showed a detection of 12864 symbols out of 14100 with accuracy 91.2% which is an improvement of the detection process over the preliminary algorithm.

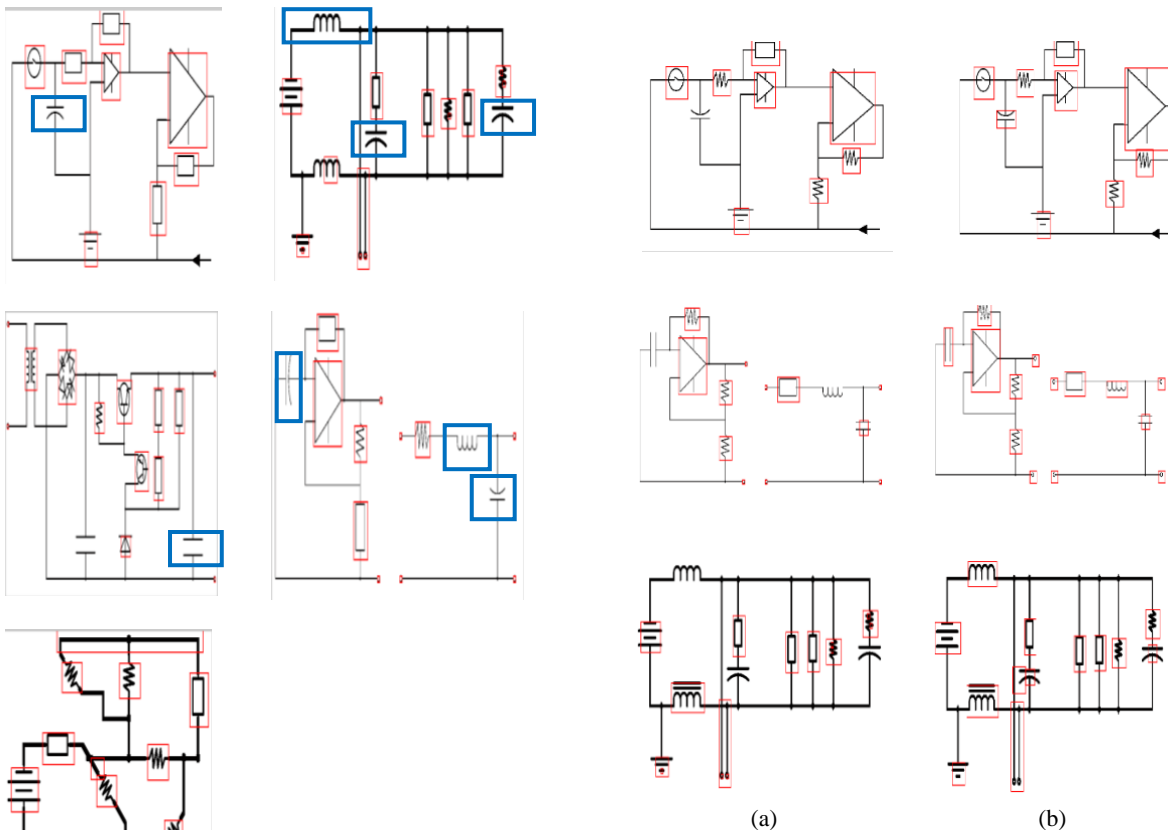


Figure 26: Detection in preliminary algorithm vs. enhanced algorithm. (a) Result sample from preliminary algorithm. (b) Result sample from enhanced algorithm.

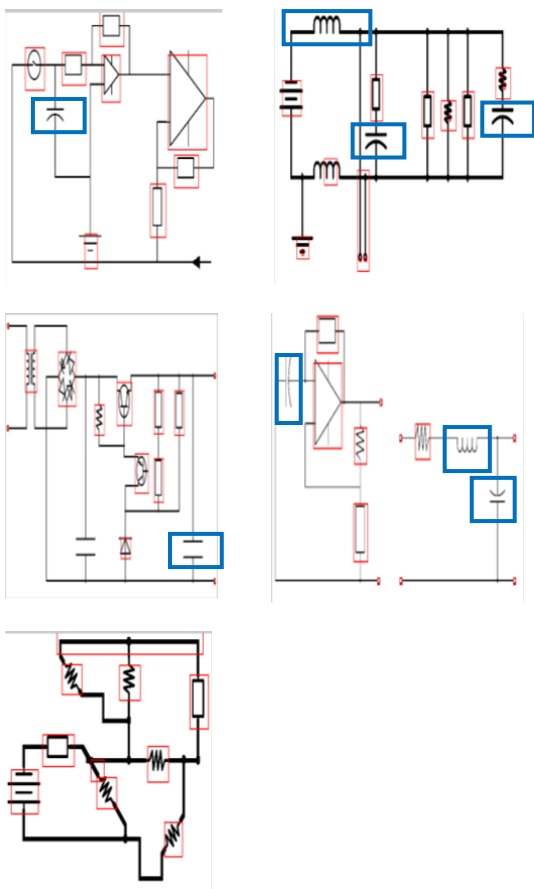


Figure 25: Undetected and miss detected symbols of preliminary algorithm shown wrapped in blue boxes.

## 9. CONCLUSIONS

The study concludes that morphological operation is powerful in detecting symbols in electronic diagrams. The study supports the idea that combining morphological operation with Hough transform improves the detection rate. Our preliminary algorithm detects 12151 symbols out of 14100 (i.e. 86.2% recognition rate). The proposed enhanced algorithm (it adds an extra iteration to the preliminary spotting algorithm to increase spotting rate) rises from 86.2% to 91.2% and the number of detected symbols increase from 12151 to 12864. Symbols like Non-Polarized Capacitor and Inductor are not spotted in the preliminary algorithm but spotted by the enhanced algorithm. It was shown in this work that combining morphological operations with Hough transform can detect open symbols like Ground as well as symbols with closed shape parts like Inductors. For future research, it is suggested to add more stages to the method to increase the detection accuracy.

## REFERENCES

- Datta, R., Mandal, P. D. S., & Chanda, B. (2015). Detection and identification of logic gates from document images using mathematical morphology. *2015 Fifth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*.
- Datta, R., Mandal, S., & Biswas, S. (2019). Automatic Abstraction of Combinational Logic Circuit from Scanned Document Page Images. *Pattern Recognition and Image Analysis*, 29(2), 212–223.
- Delalandre, M. (2012). *Systems Evaluation SYnthetic Documents*. Mathieu.delalandre.free.fr. <http://mathieu.delalandre.free.fr/projects/sesyd/>
- Delalandre, M. (2016). *generation of graphical ground Truth*. Mathieu.delalandre.free.fr. <http://mathieu.delalandre.free.fr/projects/3gT.html>

- Edwards, B., & Chandran, V. (2000). Machine recognition of hand-drawn circuit diagrams. *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*.
- Efford, N. (2000). *Digital image processing : a practical introduction using Java*. Addison-Wesley.
- Fan, Z., Zhu, L., Li, H., Chen, X., Zhu, S., & Tan, P. (2021). FloorPlanCAD: A Large-Scale CAD Drawing Dataset for Panoptic Symbol Spotting. *ArXiv:2105.07147 [Cs]*.
- KaiBin, O., & Hooi, Y. K. (2021). Critical Literature Review of Named Entity Recognition in Symbol Spotting. *2021 International Conference on Computer & Information Sciences (ICCOINS)*.
- Moetesum, M., Waqar Younus, S., Ali Warsi, M., & Siddiqi, I. (2018). Segmentation and Recognition of Electronic Components in Hand-Drawn Circuit Diagrams. *ICST Transactions on Scalable Information Systems*, 5(16), 154478. <https://doi.org/10.4108/eai.13-4-2018.154478>
- Müller, S., & Rigoll, G. (2000). Engineering Drawing Database Retrieval Using Statistical Pattern Spotting Techniques. *Graphics Recognition Recent Advances*, 246–255.
- Najman, L., Gibot, O., & Barbey, M. B. L. (2001). *Automatic Title Block Location in Technical Drawings*.
- Qureshi, R. J., Ramel, J.-Y., Barret, D., & Cardot, H. (2008). Spotting Symbols in Line Drawing Images Using Graph Representations. *Lecture Notes in Computer Science*, 91–103. [https://doi.org/10.1007/978-3-540-88188-9\\_10](https://doi.org/10.1007/978-3-540-88188-9_10)
- Rezvanifar, A., Cote, M., & Albu, A. B. (2020). Symbol Spotting on Digital Architectural Floor Plans Using a Deep Learning-based Framework.
- Rezvanifar, A., Cote, M., & Albu, A. B. (2021). Geometry-based symbol spotting in born-digital architectural floor plans. *Journal of Electronic Imaging*, 30(04).
- Rezvanifar, A., Cote, M., & Branzan Albu, A. (2019). Symbol spotting for architectural drawings: state-of-the-art and new industry-driven developments. *IPSJ Transactions on Computer Vision and Applications*, 11(1).
- Tabbone, S., Wendling, L., & Zuwala, D. (2004). A Hybrid Approach to Detect Graphical Symbols in Documents. *Document Analysis Systems VI*, 342–353.
- Zhang, W., & Liu, W. (2007). A New Vectorial Signature for Quick Symbol Indexing, Filtering and Recognition. *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*.
- Zuwala, D., & Tabbone, S. (2006). A Method for Symbol Spotting in Graphical Documents. *Document Analysis Systems VII*, 518–528.