Translating an electrical circuit schematic into a SIMULINK model

Andreas Boltres
Karlsruhe Institute of Technology
Kaiserstr. 12, 76131 Karlsruhe, Germany
andreas.boltres@student.kit.edu

Cem Gülşan
Hamburg University of Technology
Am Schwarzenberg-Campus 1, 21073 Hamburg, Germany
cem.guelsan@tuhh.de

Christian Alexander Vecsei
RWTH Aachen
Templergraben 55, 52062 Aachen, Germany
christian.vecsei@rwth-aachen.de

Thomas Frei
University of Applied Sciences and Arts Northwestern Switzerland
Bahnhofstrasse 6, 5210 Windisch, Switzerland
thomas.frei1@students.fhnw.ch

Abstract

We present a functionality developed within the computing environment MATLAB, which analyses schematics of electrical circuits and transforms them into an electrical circuit model of MATLAB's SIMULINK environment.

1. Introduction

1.1. Motivation

Students and engineers often use schematics while studying or developing. A schematic is a drawing of an electric circuit which should contain all information about the circuit but should also be as simple as possible.

With increasing size, electrical circuits often become more complicated and confuse less experienced engineers, rendering the estimation of the circuit's behaviour hard or impossible.

A program that confidently identifies the components of a schematic with their respective connections and translates them into a user-friendly digital model can substantially accelerate the understanding of the given circuit. Analyzing, modifying and storing a schematic becomes easy once it is properly digitized.

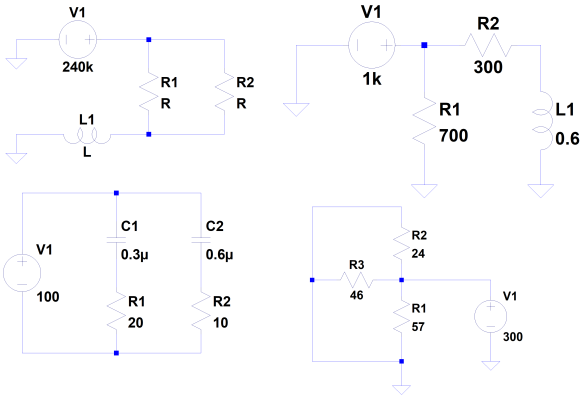

Figure 1. Example circuits

1.2. Program overview

The aim of this student project is, as stated in the abstract, to create a functionality developed within the computing environment MATLAB, analyzing schematics of electrical circuits and transforming them into an electrical circuit model within MATLAB's Simulink environment. There, it can be used to facilitate calculations or behaviour analyses. The process of obtaining the Simulink model from the given image can be divided into three phases which

will be briefly described in the following sentences and in detail in the following sections.

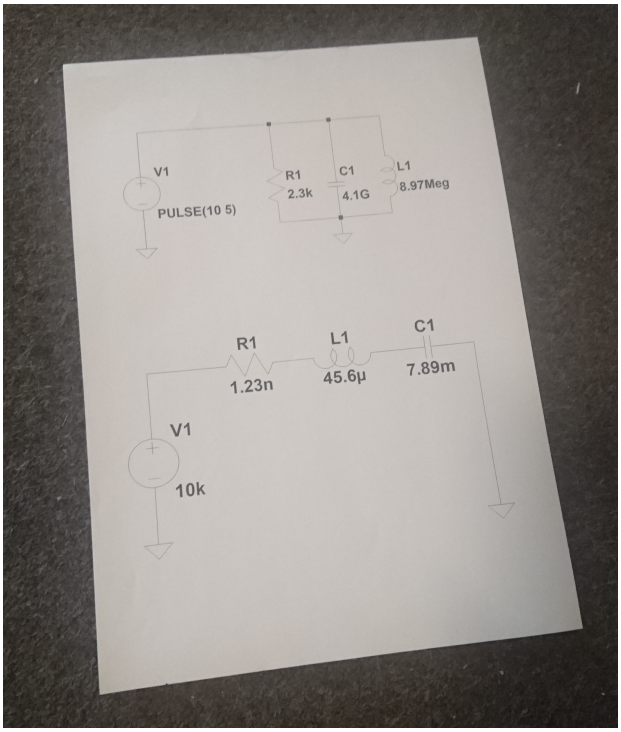

Figure 2. Example photography of a circuit

Initially, the program expects a digital image to be selected by the user. This image can be either a photography of a schematic on a sheet of paper or the image of a digitally created schematic. If the image is a photography, the program will first detect the boundaries of the sheet of paper in order to then rectify the image such that the sheet of paper lies entirely in the image plane. This implies that the entire paper is clearly included in the photography. After rectification, the photo can be treated in the same way digitally created schematic pictures are to be treated.

The user selects a region of interest (ROI) within the provided (rectified) image; the following analysis will be limited to that specific ROI. The first step of the image analysis algorithm is **Optical Character Recognition** (OCR), as every electrical element comes with electrical properties textually encoded next to them in the image. Recognized text is stored and deleted from the image. Hereby OCR is running in a loop: After each OCR iteration, the program asks the user to confirm the removal of all text within the selected ROI. If this is not the case, he is asked to manually select new regions of interest so that the OCR routine can obtain and remove the remaining text.

Following up is the **retrieval of geometric primitives** within the images which will be further explained in subsection **??**. Obtained through the binarized and thinned image, these primitives are used to retrieve the scale of the

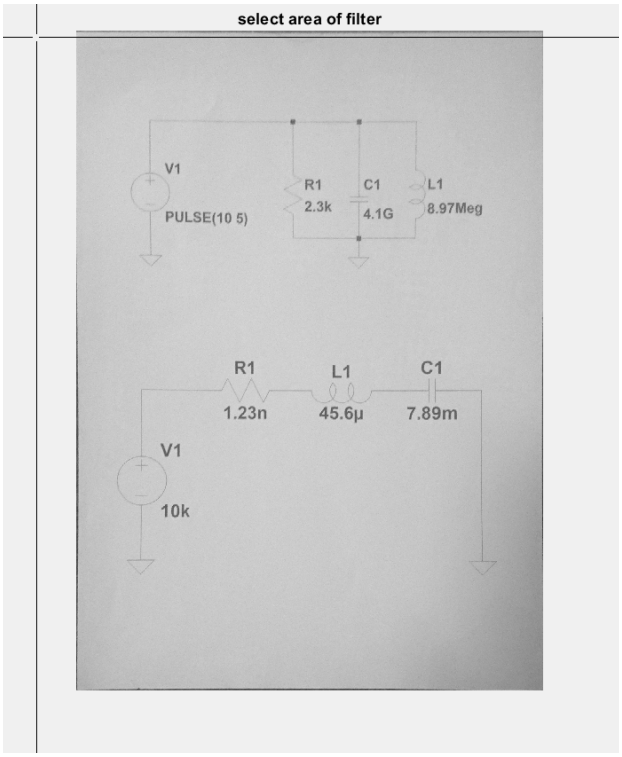

Figure 3. ROI selection within a rectified photography

electrical components of the schematic in the image.

Using the knowledge of the components' scale, the **object detection** is done using a sliding window to retrieve the location and orientation of electrical elements in the image. Following up is the **detection of wires** based on the image after deletion of the detected elements. See subsections **??** and **??** for more detailled descriptions.

Finally, after having detected elements and the connections in between, the **output generation** model uses the objectified results of the image analysis and creates and fills a Simulink model. This model will then be shown and saved for further usage. Refer to section **??** for further imformation.

2. Preprocessing

Before being able to obtain the electrical circuit elements from the provided image, several steps have to be taken in order to facilitate the information retrieval.

2.1. Rectification

When dealing with photographies, the depicted schematic may be distorted due to the surface not being aligned to the image plane. This has to be taken care of in order to use the object recognition approach presented in this work. Therefore, asserting that the entire surface and its corner points are included in the image, a rectification

step is taken to bring the schematic onto the image plane.

In order to do this, the image needs to be processed first. After binarizing the image with a predefined threshold, the biggest shape gets extracted, assuming that the paper takes up the biggest part of the image. After additional filtering such as edge detection, dilation and erosion of the result, the remaining holes (mostly text) will be filled so that only a tetragon remains.

By taking the boundaries of the image and calculating its mean values, the resulting polar coordinates can be transformed to cartesian coordinates.

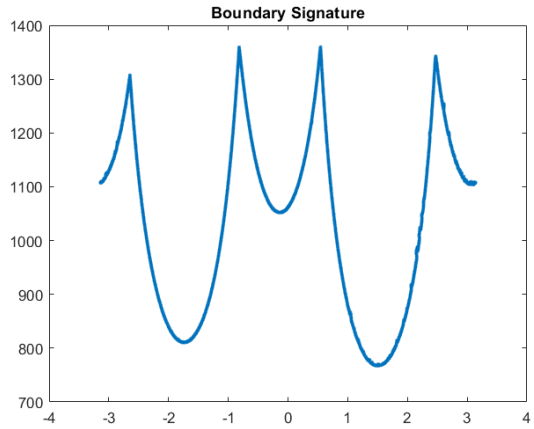

Figure 4. Boundary signature of a detected paper

Figure **??** shows the result of that transformation. Taking the value of the peaks with the highest prominence, the location of the edges can be found. With the resulting points, the corresponding points need to be generated. This is done by taking the mean value of the point pairs, which are defined by the pair that is closest together. Thereafter, the aspect ratio will be corrected to be identical to that of a *A4* paper. After a simple warping of the image the edges get detected again to verify the result and the image gets cropped according to the result.

2.2. Optical Character Recognition

In depictions of non-distorted or rectified schematics, MATLAB's OCR functionality can be used to retrieve text and its position within the image. After retrieval, the corresponding text regions in the image are erased, simplifying the following recognition of electrical elements and connections.

For the OCR function, a specific language was trained using the *MATLAB* training function. This yields more precise results, as only letters contained in *RLC*-filter circuits are allowed (and component values between p and G). All values that have been found to be above a predefined confidence (depending on photo or screenshot) will be extracted and the text will be erased from the image. If not all text

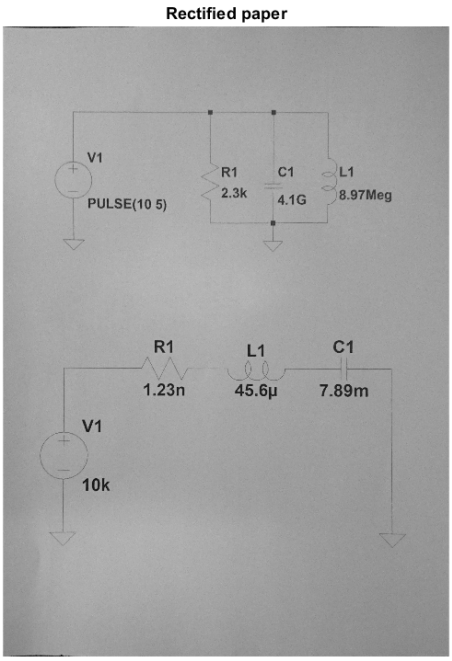

Figure 5. Post-rectification of image in figure **??**

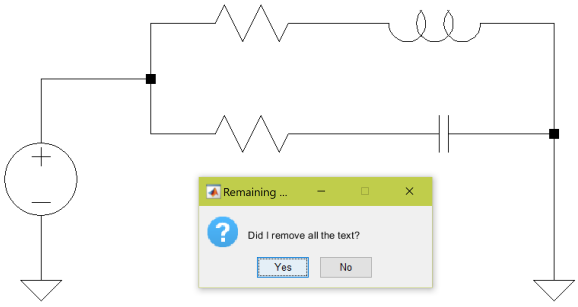

Figure 6. Successful OCR result

was found successfully (if the text is too close to a component), the user can define the region where the rest of the text lies (with the confidence lowered). If the image is a photography, some adaptive and noise filtering will be done before the OCR and some line thickening afterwards, since the image will have more grey scale values compared to the screenshot. Figure **??** shows the successful result of the algorithm.

2.3. Binarization, Morphology

After all text has been recognized and erased from the image, a binarized and thinned-out version of the image is prepared to further facilitate the following retrieval of lines, circles, corners and element template matchings. Both operations are done using pre-defined functions within MAT-

LAB.

3. Image Analysis

The goal of the functions within the image analysis package is to retrieve all objects from the pre-processed image. This includes the electrical components themselves as well as the wires connecting them. The approach used for the detection of the electrical components is a sliding-window approach, implying that a robust estimation of the scale electrical components is essential in order to find the objects quickly. This in turn calls for the usage of geometric primitives in order to determine that scale.

3.1. Geometric primitive retrieval

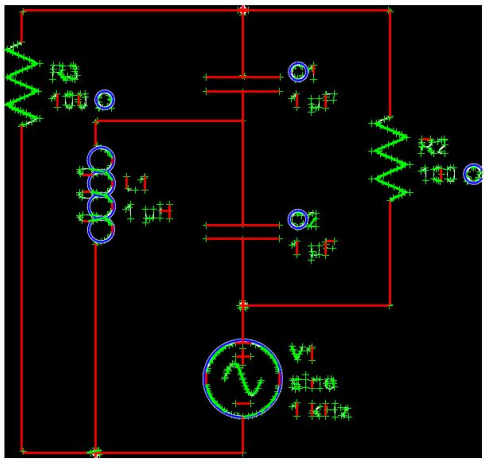

Figure 7. Retrieved geometric primitives: circles in blue, lines parallel to the axes in red, skewed lines in green and corners as green crosses.

For lines, a parametrized hough transformation is used. The results are rearranged into a data structure including length and angle of all lines, using the given start- and endpoints of the lines.

For circles and corners, which are unused in the current version of the program, predefined MATLAB functionality is used to detect the harris features of the image and the centers and radii of circles are found using a circular hough transform.

In order to retrieve the scale of the elements, skewed lines are used because in rectified schematics conforming to LTSpice, they are used in resistors only. They can be used to retrieve their dimensions, when assuming that every schematic has at least one resistor and every resistor is conforming to the LTSpice regulations. Then, the greater one of the coordinate differences of full-length lines (of which one is highlighted in Image **??**) along the axes corresponds to the shorter side of the resistor whereas four times the coordinate difference along the other axis corresponds to the longer side of the resistor.

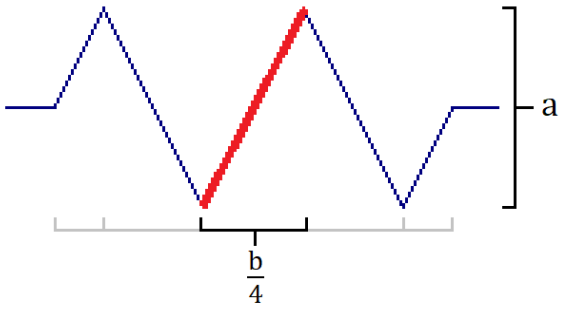

Figure 8. Resistor template with highlighted full-length line and resistor dimension explanations

Using this heuristic, the size of a resistor can be estimated within tolerance of a few pixels. Furthermore, including the stretched half-length lines which lie at the end of resistors, the algorithm becomes more robust.

3.2. Object Detection

The general approach used for object detection is a sliding-window approach. The structure of object detection goes as follows:

- Specific pre-processing for object detection
- Sliding window: Generation of an error-image (difference from template to reference)
- Get position of minimum error/maximum correlation
- Ranking and selecting candidates
- Return found elements

Preprocessing for Object Detection

Before the sliding window can be applied, the image needs to undergo specific preprocessing, which is an integral part of object detection. The template (i.e. a resistor) and the reference image (i.e. the circuit) are first binarized, scaled down for performance and then dilated. The dilating factor is an essential parameter and influences how tolerant the algorithm is of dissimilarities between the template image and actual representations of the template in the reference image. Increasing the dilation factor increases the change to find high-correlation areas but also decreases accuracy and precision. In figure **??** and figure **??** we can see a template and a reference image being (heavily) dilated. A scaling factor between 0.1 and 0.4 proofs to be both relative fast and still returns good results.

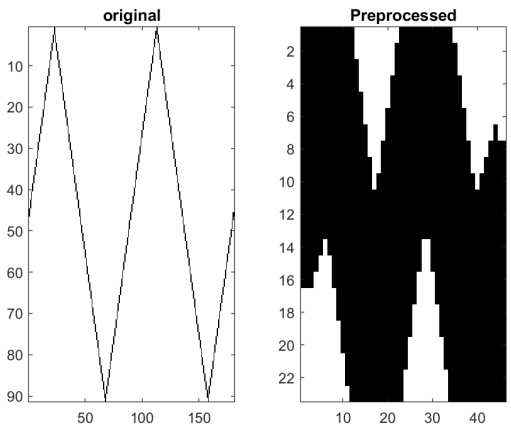

Figure 9. Dilated template

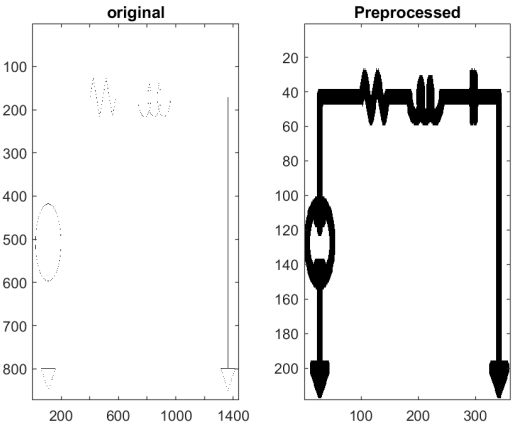

Figure 10. Dilated reference image

Sliding window and error-image

The preprocessed template is now sliding over the reference image, while a normalized mean-squared error (or difference value) is calculated at each position. The error values form the error image. Figure **??** and**??** each show a dilated reference image on the left and the corresponding error visualization on the right. Hereby, figure **??** shows much lower dilation values than figure **??**. Observing the error patterns, it is clear that the highly dilated image yields better results, but also more maxima where none should be.

Get Maximum Correlation

Using the error image (or inverted: the correlation image), the algorithm tries to find significant maxima in the image. Noise and similarities between objects (e.g. grounds and resistors, or capacitors and wires) make the search for maxima difficult. The algorithm first supresses all values below a certain dynamic threshold and then binarizes again, using another dynamic threshold. Occasionally, the final binarization fails to produce good results and higher dilation values must be used.

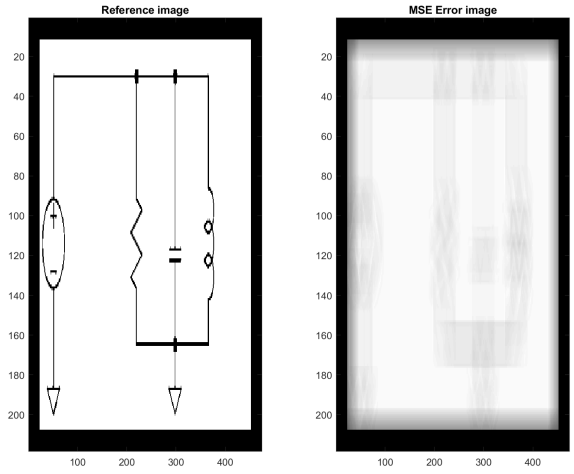

Figure 11. Slightly dilated reference image and resulting error image

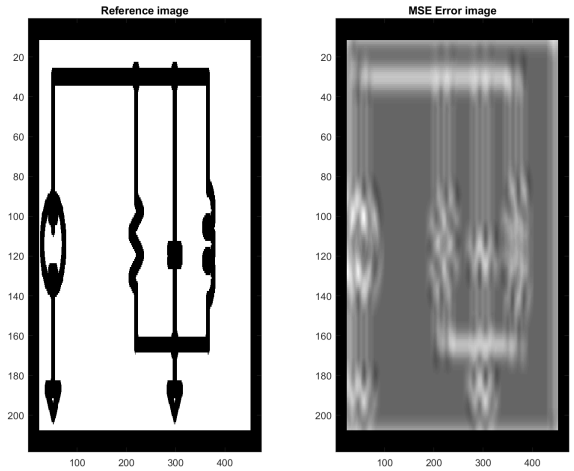

Figure 12. Heavily dilated reference image and resulting error image

Figure **??** to **??** show the different attempts to identify a capacitor, which is particularly hard to detect with certainty due to its very simple geometric shape. In figure **??**, the target maximum is not significant enough and its value is too close to the rest of the error-image. The final binarization results in an all positive image. In figure **??**, other areas have significant - although faulty - maxima aswell, resulting in a faulty final binarization. Figure **??** shows a successful suppression of minima and a successful binarization, with an isolated, significant dot as result.

The algorithm detects those isolated dots and decides whether it has found an object at concerned position,

whether it needs to use different parameters or whether there is no element of this kind to be found.

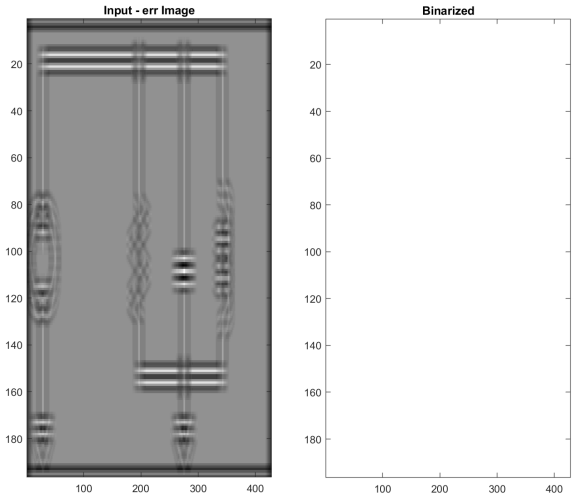

Figure 13. Faulty binarization - main maximum too insignificant

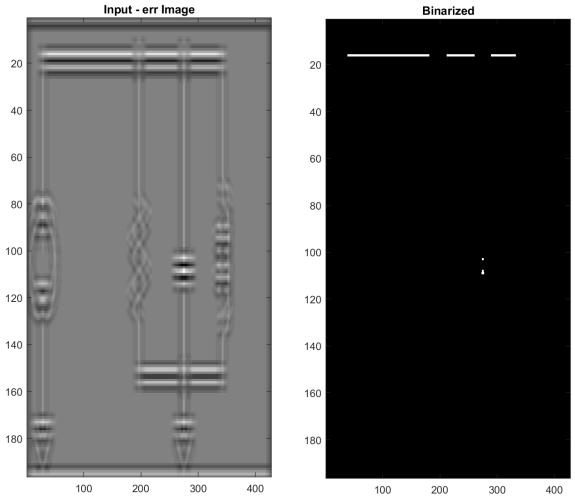

Figure 14. Faulty binarization false maxima detected

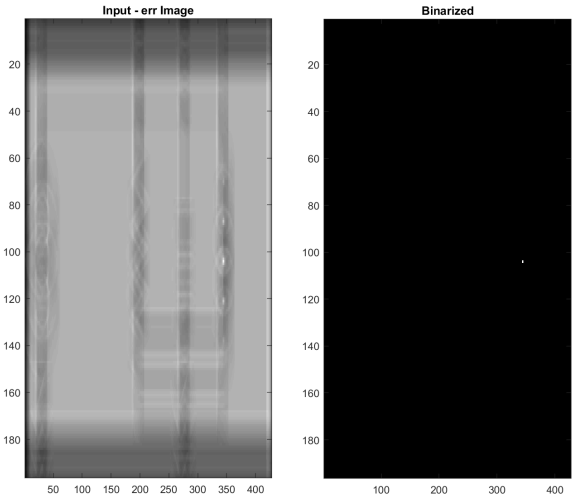

Figure 15. Successful binarization

the algorithm searched for all elements, it checks whether the found elements are out of bounds (the edges of the reference image are prone to faulty maxima) and compares their store against each other.

Figure **??** shows all found candidates for this particular example. Attached to each found element is a confidence score, with lower scores being better. When two or more elements occupy the same area, the element with the lower score gets priority and the others are deleted. The result of this selection process can be seen in figure **??**.

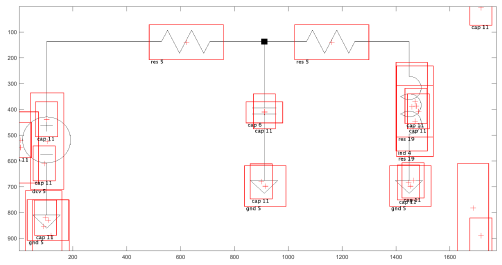

Figure 16. All candidate elements with scores

Ranking and Selecting Candidates

As mentioned before, the algorithm is prone to confuse different elements with each other, provided they share geometric similarities. When dilating template and reference image, this confusion gets worse as the tolerance for non-perfect matches increases. Frequently, multiple elements are detected (with varying tolerance) in the same area in the reference image. To counteract this, the algorithm assigns a score to each found element relating to its certainty. After

Return Found Elements

The found and selected elements are stored and returned to the main image analysis function for further processing. In the image, the areas of the found elements are erased to enable the following wire detection.

3.3. Wire Detection

After having detected the electrical elements and after having erased them from the image, cables and connections

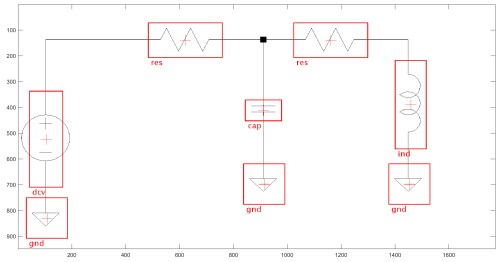

Figure 17. Selected candidates (final result)

are the only entities left in the image. Therefore, a second hough transformation application retrieves the endpoints of the connections between elements.

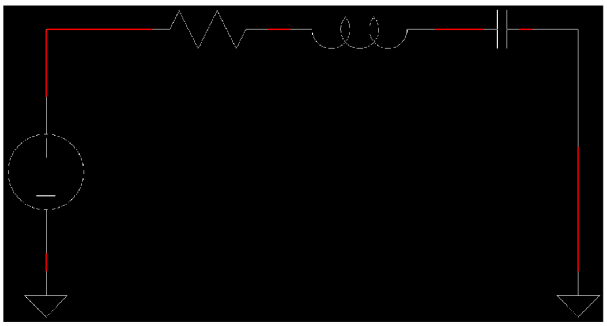

Figure 18. Circuit from figure **??** with recognized cables in red.

After the wire detection, all detection results are passed onto the output generation module.

In this module, the results of the image analysis routines are reframed and sent to SIMULINK in order to build an electrical circuit model. After its creation, the user can interact with the model which is also saved to enable further usage.

4. Output Generation

In this the step the problem takes all the data which has been gathered in the previous steps and generated the Simescape model from it. Simscape is a special library for Simulink to simulate physical systems.

4.1. Object Generation

The input to the Object Generation are on one hand the Elements detected in the Object Detection. On the other hand the OCR input is used to find the values and labels corresponding to the input. LTSPICE models have the property, that the Label is always above the value. So to assign them to an object we search the labels closest to the detected object by taking the center coordinates of the object and then calculating the distance to the centers of all labels with the formula

$$d = \sqrt{(L_x - O - y)^2 + (L_y + O_y)^2}$$

From the results the two closest labels are taken and ordered by y-coordinates to find out which one is the value and which one the value. From the Output Detection the position and the detected object type are taken and an object of the type is instantiated. To save the objects the following class structure was introduced: The most general class is

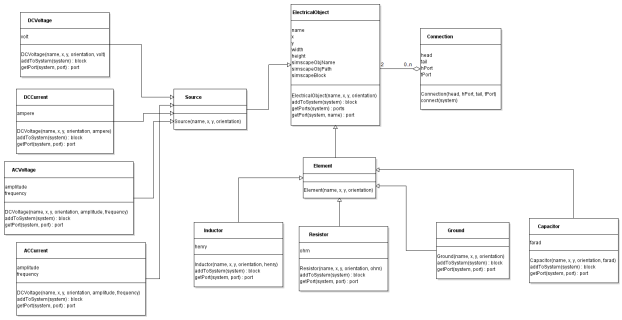

Figure 19. Class diagram of classes that store electric elements

ElectricObject which holds information and methods which exist in all objects, like position, name and methods to add them to the system. From this class the subclass Source and Element exist. The sources, which can be stored are AC Voltage, AC Current, DC Voltage and DC Current. The big difference between AC and DC sources is, that the first one have a frequency and amplitude as parameter, while the second ones have just a constant value which the supply. For the elements classes for resistors, capacitors, inductors and grounds exist. A special case is the Connection class, which represents a connection between Objects, while also specifying to between which ports the connection exists. This are the basic models from which the system is built.

4.2. Wire Preprocessing

This step simplifies the wire next work so it is usable by the next step Wire Processing. Since the hugh transform gives lines which can be overlapping, those lines are merged to single wires. After that lines which form nodes are connected, by moving the end point of the wires which touches another wire to an endpoint of said wire.

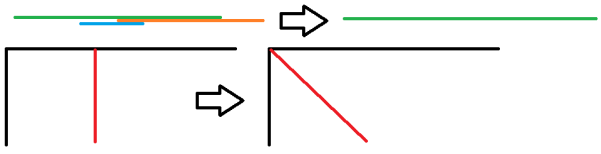

Figure 20. Steps done in Wire Preprocessing: Merging on top and intersection detection on bottom

4.3. Wire Processing

In the Wire Processing all the objects are taken and a list of all ports is generated. From these ports a depth first search is used to find all other ports connected to this wire tree. Then all Connections are instantiated between all Ports of the Objects found and the root port. After that the found ports are removed from the list and the search continues until no ports are left.

4.4. Output Generation

In this step a new Simulink system is created. First all the objects are passed the system with the call to add themselves to it. After this is finished the same is done with the connections. The user will now have the finished model of the system, which he wanted to analyze, on his screen and can add Elements like voltage sensors and scopes to analyze his system.

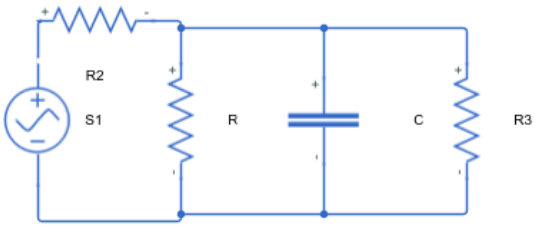

Figure 21. Finished system as displayed in Simulink

5. Evaluation

Extensive evaluation is a key technique to determine the feasibility of the approach and its parts presented in this work. Therefore, every functionality should be tested qualitatively as well as quantitatively. For this purpose, a bunch of electrical circuit schematics have been created; the algorithm's parts will be tested on each of these images.

5.1. LTSpice and assertions

In order for the presented approach to produce meaningful results, a few assertions have to be made concerning the design of the circuits and the appearance of its elements:

- Circuit images are generated with LTSpice[1], which uses normed representations of all relevant circuit elements. These representations match the templates used in the sliding-window approach.
- circuits are drawn with black colors on a white background. In photographies: The contrast between circuit and background should be sufficient.
- In photographies: All four corners of the rectangular surface on which the circuit is depicted have to be visible. That shape is the biggest on the photography.
- Values of electrical properties of elements range from $1p$ $(1 * 10^{-12})$ to $999G$ $(999 * 10^{9})$.
- Every circuit has at least one resistor. All circuits are RLC-Circuits with one source of voltage.
- All electrical circuit elements and connections are oriented either vertically or horizontally.
- There is sufficient white space between elements and pieces of text.

[1] http://www.analog.com/en/design-center/design-tools-and-calculators.html?domain=www.linear.com#LTspice

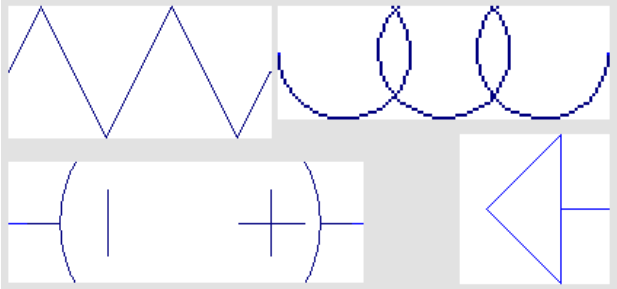

Figure 22. Examples of LTSpice-conform circuit elements

5.2. Rectification

To validate the rectification, 15 photographs of papers were collected and rectified. Then the number of pixels that are not part of the paper inside the rectified image is being counted. The average result is 99.5%, with the smallest measured value being 98.8%. The pixels which are contained in the rectified image but are outside of the paper are a consequence of the paper not being flat on the ground, which in turn results in small curls. Since the user has to select the area of the filter, this proportion can be neglected.

5.3. OCR

To evaluate the OCR algorithm, the number of detected text patches compared to the input image was calculated.

Table 1. Evaluation of OCR algorithm

Found w/o user	Found w/ user	Correct
73.62%	100%	95.7%

Instead of listing every single of the 30 tested images, only the average result is displayed in table **??**. The values that were evaluated is the text that was found without the user specifying a region, the text that was able to be detected and the amount of text that was correctly detected. The score of the found text without user interaction is somewhat low because the OCR algorithm supplied by MATLAB has

trouble finding text if it is close to the components. Also, the correctly detected text is high, but the wrongly identified test is mostly caused by similar letters like upper- and lower case P.
Additionally, the same procedure was made for several images of two printed schematics. The text was always correctly detected, but the detection without user-interaction is 59.4%, which is expectedly lower compared to screenshots. This is due to noise in the image and a lower resolution.

5.4. Scale Retrieval

In order to evaluate the performance of the scale retrieval routine, the dimensions of the resistors of each image has been measured manually. Thereafter, each image was sent through the program until the resistor dimensions were retrieved. In table **??**, the dimensions of resistors in each image (column *Measured*) can be found alongside the dimensions retrieved by the algorithm for each picture (column *Calculated*) and a maximum deviance of the resistor size for each dimension. A manual measurement tolerance of one pixel is included in this maximum deviance metric.

File	h/v	Measured	Calculated	Deviance
01.png	horiz.	180×91	180×88	$< 1 \times 4$
02.png	vert.	91×180	90×194	$< 2 \times 5$
03.png	horiz.	180×90	176×87	$< 5 \times 4$
04.png	horiz.	140×71	140×69	$< 1 \times 3$
05.png	vert.	46×92	46×96	$< 1 \times 5$
06.png	vert.	46×92	46×96	$< 1 \times 5$
07.png	horiz.	206×103	208×102	$< 3 \times 2$
08.png	vert.	108×215	106×216	$< 3 \times 2$
09.png	vert.	85×169	83×168	$< 3 \times 2$
10.png	vert.	127×255	126×256	$< 2 \times 2$
11.png	horiz.	216×108	216×107	$< 1 \times 2$
12.png	vert.	109×216	108×212	$< 2 \times 5$
13.png	horiz.	199×100	200×98	$< 2 \times 3$
14.png	vert.	94×185	91×184	$< 4 \times 2$
15.png	horiz.	265×133	264×130	$< 2 \times 4$
16.png	horiz.	215×108	216×106	$< 2 \times 3$
17.png	vert.	98×195	94×192	$< 5 \times 4$
18.png	vert.	108×215	108×224	$< 1 \times 10$
19.png	vert.	118×235	114×232	$< 5 \times 4$
20.png	horiz.	245×122	248×120	$< 4 \times 3$

Table 2. Resistor size measurements and estimations (in px)

A maximum deviance of 5 pixels is considered satisfactory for an effective usage of the object detection algorithm. With the exception of one case, the scale retrieval algorithm could retrieve the resistor size in all images it got called on. This corresponds to an accuracy of 95%.

5.5. Object Detection

To evaluate the object detection algorithm, perfectly prepared images (screenshots of circuits without text) were used. The algorithm was tested on how well it detected all objects correctly and on the number of phantom detections, which are detections without actual objects being at concerned places.

Through tables **??** (collected data) and **??** (main results), the following conclusions can be made:

- There is a 4.8% miss rate; The chance that an object is misclassified.
- There is a 3.9% not-detection rate; The chance that an object is not recognized at all.
- There is a 7.4% phantom rate; The chance that additional objects are recognized, where none actually are. This is in relation to total actual objects.
- The ratio between total number of candidates and total number of selected objects is 25%, ranging from 13% to 35%.
- The correct identification rate, that is the number of correctly identified objects in relation to the number of actual present objects, is 90.9%

The detection algorithm performs well in certain circumstances. The detection algorithm has severe difficulties when the input images are not well rectified or cleaned (no text, no noise). The most common error sources were ground- or capacitor elements being recognized where none were, or where other sources were. This can be explained by their very simple and common shape. Voltage sources were also often misidentified or placed where no objects actually were (phantom objects).

ID	EL	CC	SC	Cdet	Mdet	Ndet	Pdet
1	6	18	7	6	0	0	1
2	6	17	6	6	0	0	0
3	8	23	8	8	0	0	0
4	9	22	10	9	0	0	1
5	9	19	9	8	1	0	0
6	11	26	11	10	1	0	0
7	8	18	8	8	0	0	0
8	5	24	5	5	0	0	0
9	11	35	12	11	0	0	1
10	7	13	7	7	0	0	0
11	8	24	9	8	0	0	1
12	6	35	5	4	0	2	1
13	5	15	6	5	0	0	1
14	7	21	9	6	1	0	2
15	7	17	6	6	0	1	0
16	6	22	7	5	0	1	2
17	9	25	10	9	0	0	1
18	7	28	7	6	0	1	1
19	7	24	7	6	0	1	1
20	7	22	6	5	0	2	1
21	9	24	10	9	0	0	1
22	8	30	9	7	1	0	0
23	9	25	9	9	0	0	0
24	7	13	7	6	1	0	0
25	9	36	9	8	1	0	0
26	7	32	10	6	1	0	2
27	8	29	7	7	0	1	0
28	8	36	8	6	2	0	0
29	7	28	7	6	1	0	0
30	8	29	8	7	1	0	0
	REL				4.8%	3.9%	7.4%

Table 3. Collected data from tests. ID: Test image ID, EL: Nr. of correct elements, CC: Candidate Count, SC: Selected Count, Cdet: Correctly detected, Mdet: Misdetected, Ndet: Not detected, Pdet: Phantom detected

ID	SR%	ELEM	CORR%
1	28%	6	100%
2	26%	6	100%
3	26%	8	100%
4	31%	9	100%
5	32%	9	89%
6	30%	11	91%
7	31%	8	100%
8	17%	5	100%
9	26%	11	100%
10	35%	7	100%
11	27%	8	100%
12	13%	6	67%
13	29%	5	100%
14	30%	7	86%
15	26%	7	86%
16	24%	6	83%
17	29%	9	100%
18	20%	7	86%
19	23%	7	86%
20	21%	7	71%
21	29%	9	100%
22	23%	8	88%
23	26%	9	100%
24	35%	7	86%
25	20%	9	89%
26	24%	7	86%
27	19%	8	88%
28	18%	8	75%
29	20%	7	86%
30	22%	8	88%
Avg	25%		90.9%

Table 4. Caption: Results of evaluation. ID: Test image ID, SR%: Selected Ratio, ELEM: Nr. of correct elements, CORR%: Correct ratio

6. Conclusion

This paper introduced a method to digitalize electrical circuits given in form of an image, using a sliding-window approach for the detection of the electrical elements of the circuit. Despite the staticness of this approach, using certain assumptions concerning the appearance of the circuit and its parts, the algorithm produced exact models and accurate topographies in a relatively stable manner.

Ideas for future work

Despite promising results, there are several measures which could improve this work:

- A limiting factor of this approach, especially when dealing with photographies, was the detection of lines using MATLAB's hough transformation implementa-

tion. Its poor performance also resulted in lines not being detected after the rectification has taken place, which explains why this part was not validated. Implementing a new and custom hough transformation is a promising way to make the retrieval of connections more robust.

- Similar to MATLAB's hough transform, its OCR functionality also struggled to correctly and robustly detect the text in the test schematics. A proper OCR solution would increase the robustness of the whole procedure just like an improved hough transform would.

- Within the object detection, instead of the comparison of image regions with the respective templates and the following calculation of error images, the process of finding electrical elements could be made more abstract, e.g. by comparing histograms of geometrical primitive imformations such as line length and angle instead of comparing color values pixel by pixel. Such an abstraction can potentially increase the tolerance of the object detection algorithm regarding the deviance of the found circuit elements from the template.

Contributions

Name	Proposal	Research	Pres.
Andreas Boltres	25%	25%	25%
Cem Gülşan	25%	25%	25%
Christian Vecsei	25%	25%	25%
Thomas Frei	25%	25%	25%
	100%	100%	100%

Table 5. List of members and their contributions to the project.