AUTOMATIC RECOGNITION SYSTEM FOR DOCUMENT DIGITIZATION IN NUCLEAR POWER PLANTS

Elisa Ou
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
Madison, WI 53706, USA

Minhee Kim
Department of Industrial and Systems Engineering
University of Florida
Gainesville, FL 32603, USA

Po-Ling Loh
Department of Pure Mathematics and Mathematical Statistics
University of Cambridge
Cambridge, UK

Todd Allen
Department of Nuclear Engineering and Radiological Sciences
University of Michigan
Ann Arbor, MI 48109, USA

Robert Agasie
Department of Engineering Physics
University of Wisconsin–Madison
Madison, WI 53706, USA

Kaibo Liu*
Department of Industrial and Systems Engineering
University of Wisconsin–Madison
Madison, WI 53706, USA
kliu8@wisc.edu

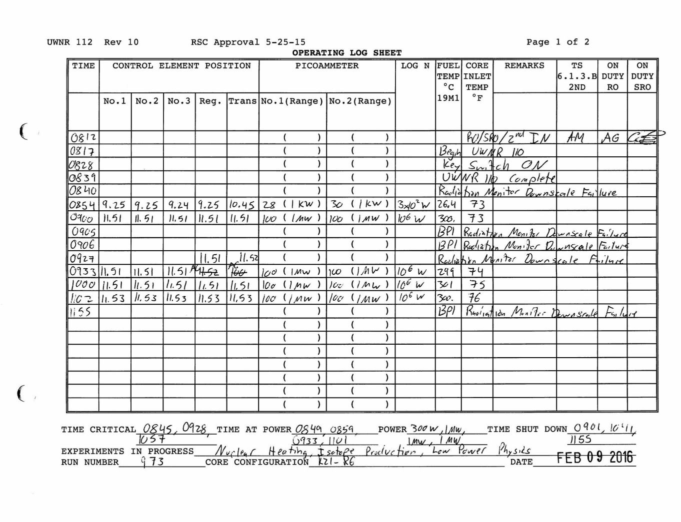
September 12, 2022

ABSTRACT

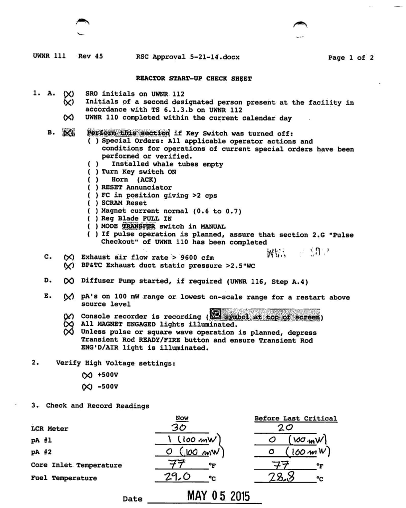
With the increasing number of data-driven models in nuclear applications, large volumes of numerical data are requiblack to accurately model and pblackict the health status of a plant component. However, many historical operation logs that contain useful information are not fully utilized due to the lack of a systematic approach of digitization. To overcome this issue, this study proposes an automatic pipeline for extracting information from handwritten tabular documents collected from nuclear power plants. In our pipeline, we first denoise scanned documents with morphological operations, and then extract relevant parts from individual pages using both traditional computer vision and neural network methods. Handwriting recognition is applied to obtain text and numbers. As the most challenging step is how to crop only relevant information, the main focus of our paper is to detect tables and cells from scanned handwritten documents. We evaluate the efficiency and accuracy of our proposed method on handwritten operational reports obtained from a real-world case study. The results demonstrate the high accuracy and practicality of our proposed method.

Keywords nuclear power plants · computer vision · image processing · neural networks · table detection · handwritten documents

*Corresponding author.



(a) Bordered table



(b) Borderless table

Figure 1: Examples of handwritten operational reports

1 Introduction

With the advent of machine learning, data-driven approaches for ensuring reliable and efficient operation of nuclear power plants (NPPs) have been attracting more attention (IAEA [2013]). However, large amounts of operation data are critical for these approaches. While some data in NPPs are traditionally acquired and stored digitally, other data are more often recorded manually in handwritten form. For example, handwritten operational logs contain useful information such as the times associated with startup, scrams, and shutdown. Digitization of such data is crucial for accurate and reliable data-driven decision making in NPPs. To the best of our knowledge, labor-intensive manual processes are currently required to digitize such handwritten NPP data, and there are no existing automatic approaches. Please note that the proposed method can also contribute to the diagnostics and prognostics of an NPP that only collects information in digital form. In particular, a new NPP which collects fully digitized information may not have experienced many critical anomalies. Without sufficient data samples of historical anomalies, the existing data-driven approaches cannot achieve accurate anomaly detection performance. In such cases, one strategy is to train a data-driven model using not only the data samples collected from the new NPP, but also those from other similar NPPs. The proposed method enables us to obtain historical anomaly data samples from a wider range of NPPs, including those that have collected non-digital and handwritten information.

To make full use of historical handwritten reports, we propose an automatic system to extract information from reports. Two examples of handwritten reports are presented in Figure 1. The main challenge in building this system is to crop only relevant information. For bordered tables as in Figure 1(a), relevant information is written in cells surrounded by lines; to match handwritten cells with table headers, we need to detect horizontal and vertical lines. Borderless tables (cf. Figure 1(b)) are more difficult to handle, as the documents are scanned in full size. Most parts of the document are non-informative, such as instructions and table titles. To efficiently extract only informative data, we first need to locate and crop the handwritten parts. A common assumption for extracting handwritten information is that the locations of relevant parts are perfectly aligned with each other in the same types of documents or tables. However, due to misalignment caused by manual handling, this is often not true in practice. Another challenge is that most modern machine learning techniques require knowing the ground truth of informative data locations in order to train a complex model, which is difficult to obtain in our case. In this paper, we effectively solve those problems by using a combination of traditional computer vision and state-of-the-art neural network methods. Notably, we also utilize a pretrained model for the neural network.

The rest of our paper is organized as follows: Section 2 gives a literature review of existing work on data digitization and table extraction. Section 3 describes our system pipeline and methodology. Section 4 presents both analytic and numerical experimental results on a real-world dataset. In Section 5, we summarize our work and list several future directions.

2 Literature review

We begin by mentioning several previous studies which focused on how to obtain digital data in NPPs and how to automate data acquisition procedures in NPPs. Rashdan and Germain [2019] reviewed various sources of data in NPPs and proposed advanced methods to automatically obtain digital data from NPPs. Deng et al. [2020] analyzed the applications and challenges of wireless sensor networks in NPPs. However, one crucial aspect that has not been widely studied is how to convert existing manually-recorded data into digital form. For instance, in healthcare applications, several studies have been conducted on the digitization of health records, including Milewski et al. [2009], who proposed an automatic digitization system for handwritten medical documents. In particular, a novel approach was developed to reduce lexicons consisting of medical and pharmacology corpora. Unfortunately, these methods from the healthcare domain cannot be directly applied to NPPs, where the major challenge is to learn the complicated structure of documents involving complex structures such as tables and checklists.

One crucial step in NPP document digitization is to crop relevant parts (e.g., certain cells in table or the handwritten portion of a checklist) from the entire scanned document image, which can be considered as a table detection problem for bordered (with lines) and borderless (without lines) tables. The goal in table detection is to locate table blocks from pictures of digital documents, e.g., images of academic journals or books. Many studies have been done on detecting tables from scanned documents, and earlier attempts use traditional image processing methods to tackle this problem. Gatos et al. [2005] proposed a workflow containing three steps: image preprocessing, horizontal and vertical line detection, and table detection. In addition, they proposed a new approach for line detection in bordered tables. Mandal et al. [2006] utilized the fact that for blocks with tables, the white spaces are larger than those of text lines, to distinguish table blocks from text or image blocks. Shafait and Smith [2010] proposed a pipeline to analyze document layout with line detection and connected component analysis, which can identify text, image, and table blocks. This implementation is also a part of the widely-used optical character recognition (OCR) engine called Tesseract.

Besides traditional image processing methods, more recently, neural networks have been widely applied to table detection by treating it as a special case of object detection. Object detection is one of the most popular topics studied in computer vision, and aims to identify and locate a certain class of objects from a digital image. Other examples of object detection applications include face recognition, computer vision systems in self-driving cars, and gesture recognition. The common practice of object detection is to draw a rectangular box around the target and compare the coordinates of predicted corners and human labels, which requires substantial work.

Works studying neural network methods for table detection include DeepSeSRT (Schreiber et al. [2017]), TableNet (Paliwal et al. [2019]), and Graph Neural Networks (Qasim et al. [2019]). However, all of these studies discuss how to extract tables from scanned images of digital documents such as academic journals, which contain no noise or handwritten elements. They also utilize a large training dataset with locations of tables annotated by humans. In our study, we focus on noisy scanned images with no prior annotations, which are very common in NPPs.

3 Methodology

3.1 System pipeline

Figure 2 shows our proposed pipeline for digitizing handwritten operating records, which consists of three steps:

1. First, we preprocess the images with denoising algorithms including thresholding and morphological operations, as scanned images of handwritten documents contain noise. This will be discussed in Section 3.2.
2. Second, we crop relevant patches from the denoised images using both traditional computer vision algorithms and neural network methods for table detection. For example, for the bordered table in Figure 1(a), we first crop only the table in the center of the page, removing titles and other information, and then crop each cell from the table as separate pieces. This will be elaborated in detail from Section 3.3 to Section 3.5.
3. Third, we apply handwritten recognition algorithms, explained in Section 3.6, to the cropped handwritten segments and fill in digital tables with recognized numbers.

3.2 Denoising with morphological operations

As our dataset contains manually scanned documents, the images are not as clean as digital documents and require preprocessing before applying detection algorithms. Morphological operations are simple operations used widely in image processing, and can be used to remove dotted noise in our scanned images. In morphological operations, a predefined shape is used as a mask (usually involving simple shapes such as a circle or square). The whole image is

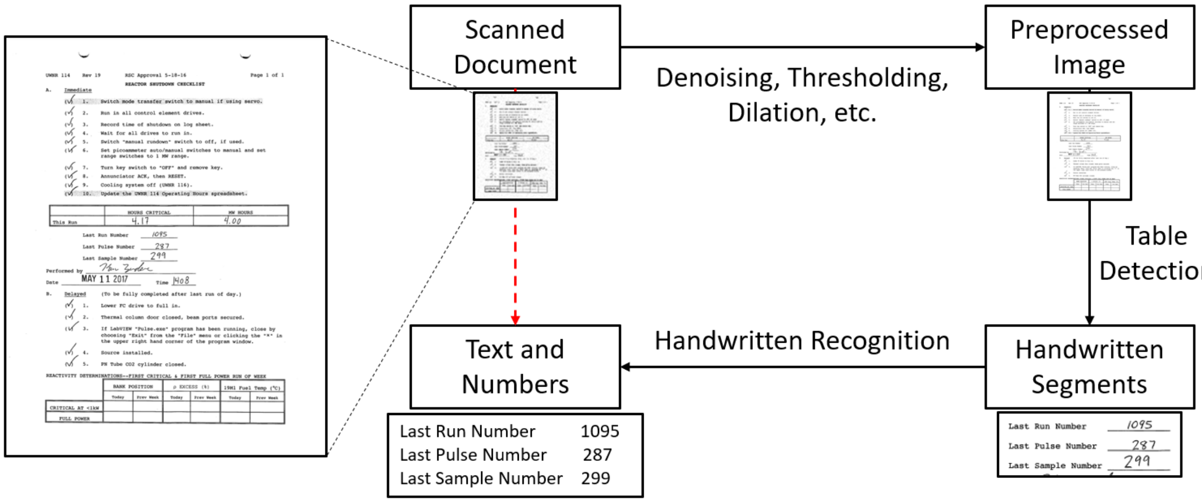
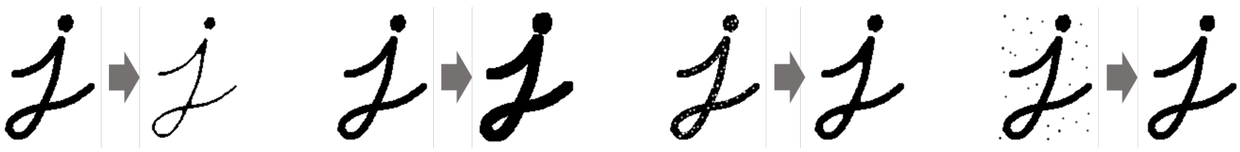


Figure 2: System pipeline for handwritten document digitization



(a) Dilation (b) Erosion (c) Opening (d) Closing

Figure 3: Example of basic morphological operations

then probed with the mask. Two of the basic operations are dilation and erosion: Dilation increases the size of objects around boundaries, while erosion decreases the size of objects. Figure 3(a) shows the dilation of an image and Figure 3(b) shows the erosion, where the foreground is in white (pixel value 1) and the background is in black (pixel value 0). To effectively remove noise in our scanned images, we need to combine those two operations. The opening operation first applies erosion and then dilation:

$$\text{Opening}(A) = f(g(A)), \tag{1}$$

where A is an image, $f(\cdot)$ denotes dilation, and $g(\cdot)$ denotes erosion. In contrast, the closing operation first applies dilation and then eroding:

$$\text{Closing}(A) = g(f(A)). \tag{2}$$

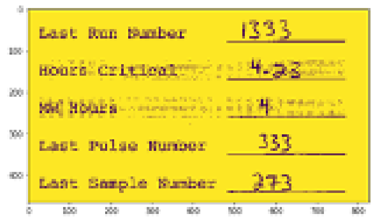
Examples of opening and closing operations on simple images are presented in Figures 3(c) and (d).

Figure 4(a) shows a part of a scanned image with noise, mostly consisting of small dots in gray. These kinds of "holes" can be removed using the closing operation, since we have a black foreground and white background (shown as yellow in plots). We fill holes (black dots) with values close to 0 by first using dilation, as shown in Figure 4(b). Since the strokes in dilated images are finer than the original image, erosion is then applied to restore the image after dilation, which could be seen as a dilation of the black parts. The final denoised image is shown in Figure 4(c).

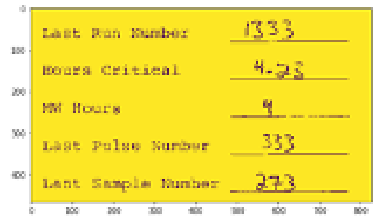
3.3 Methods for bordered tables

For bordered tables (cf. Figure 1(a)), after the proper preprocessing steps mentioned in Section 3.2, it is possible to utilize contour detection to identify cells, or the Hough line transform to detect lines.

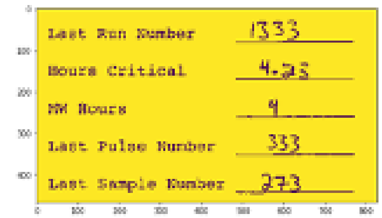
Contour detection is a method for object detection which is designed to extract continuous boundaries from images. First, we need to "embolden" the handwritten text to make the text within each cell connected. We use morphological operations to process images, as described in Section 3.2. The result of erosion is shown in Figure 5(b). Then we apply one of the most well-known contour detection algorithms, proposed by Suzuki et al. [1985], where the erosion image is scanned to find the rectangular boundaries of cells. The final result is shown in Figure 5(c). Finally, we extract cell content from scanned tables with detected rectangular contours.



(a) Original image

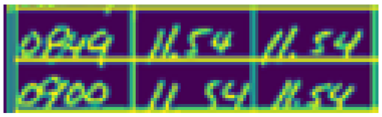


(b) Dilation of (a)

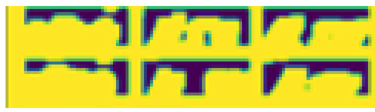


(c) Erosion of (b), which is equivalent to closing of (a)

Figure 4: Image denoising with morphological operations



(a) Original image



(b) Erosion of original image

(c) Applying contour detection on (b)

Figure 5: Results with morphological operations and contour detection

Another method to detect lines from a bordered table is Hough transform as proposed by Duda and Hart [1972]. Hough transform detects shapes by transforming digital images into a parameter space, and then finds local maxima in this space. For example, to detect a line, we could transfer the image from the Cartesian coordinate system to the polar coordinate system. Then a line in the original image is a dot in the polar coordinate image, which could be identified easily by finding points with maximum pixel values. By choosing appropriate parameters, this algorithm can detect lines from images, and we can further identify cells by detecting both horizontal and vertical lines from scanned tables. To remove noise that is not in the form of vertical or horizontal lines, morphological masks are applied before the transform. We present detailed implementations and results of these two algorithms in Section 4.2.

3.4 Methods for borderless tables

For borderless tables (cf. Figure 5(b)), we propose two different methods to crop relevant parts: template matching and neural networks with pretrained models.

Template matching is an approach in image processing that identifies parts of a source image that are similar to a given template image. Most template matching methods are based on sliding the template above the source image (windowing) and comparing the similarity of each position between the template and the source image patch. The metric we use to measure similarity is the normalized correlation coefficient, which is defined in Szeliski [2010] as

$$\mathbf{R}(x,y)=\frac{\sum_{x',y'}\left(T'\left(x',y'\right)\cdot I'\left(x+x',y+y'\right)\right)}{\sqrt{\sum_{x',y'}T'\left(x',y'\right)^{2}\cdot\sum_{x',y'}I'\left(x+x',y+y'\right)^{2}}} \tag{3}$$

for coordinate (x, y) in source image I'. Here, T' is the template image and (x', y') are coordinates of pixels in the template image and the corresponding window in the source image. $\mathbf{R}(x, y)$ refers to the similarity between the template image T' and the cropped patch from source image I' centered at (x, y). As we only require a single-center coordinate that is most similar to the given template, we then obtain the location of the template patch in the source image by applying an argmax function to the matrix $\mathbf{R}$.

Using template match, we could locate a given text. And with positions of several surrounding texts, we can further locate and crop the handwritten part. For example, as illustrated in Figure 6, if our desired area is the handwritten number in the blue box, we can use the green box "Now" and the red box "LCR Meter" as vertical and horizontal locators, respectively.

3.5 Neural network-based methods

As mentioned in Sections 3.3 and 3.4, traditional computer vision methods can obtain decent results in table detection, with effort given to selecting features and tuning parameters. However, for large datasets involving different types of documents feature engineering requires a great deal of manual work. Convolutional neural networks (CNNs) are a

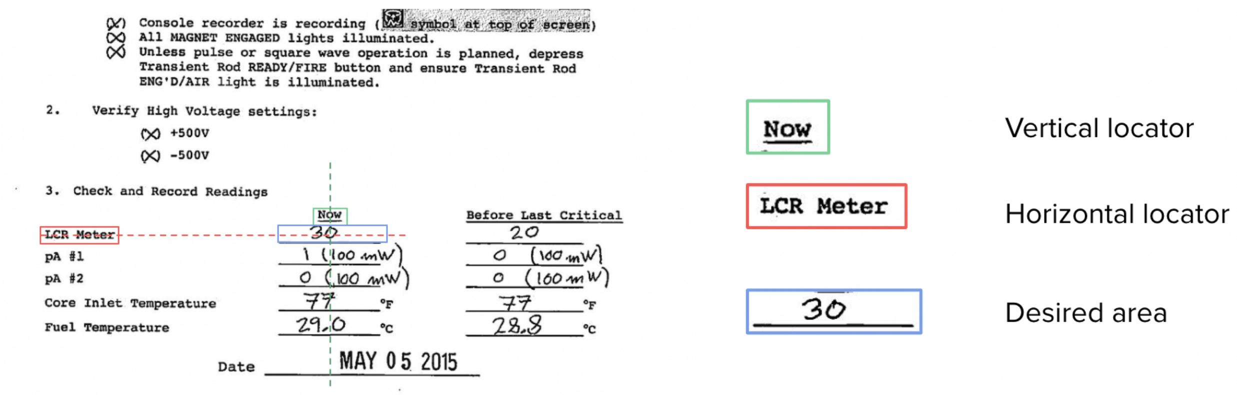


Figure 6: Template matching for locating

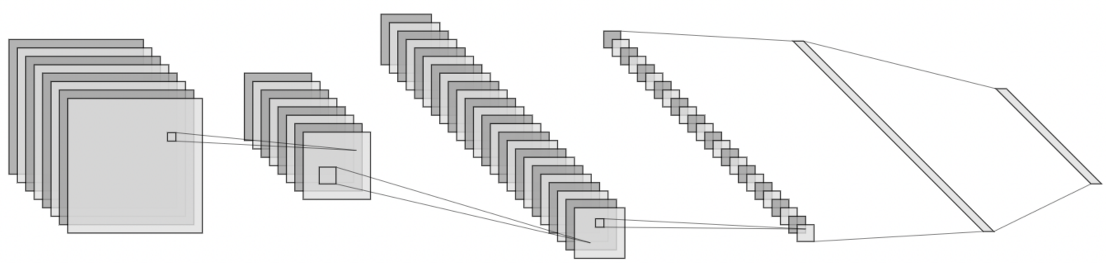


Figure 7: A typical CNN structure

special class of neural networks which can resolve these issues by using convolutional layers to directly extract features from digital images.

Figure 7 presents a typical structure of a CNN. Each convolutional layer maps a small patch from the previous layer to a single pixel. Thus, by stacking several convolutional layers, it is possible to transfer a large image into a relatively short vector. The parameters in each layer can be trained by back-propagation with properly labeled data.

To train a CNN, labels are required for the training set of images. In table detection tasks, the training data are image-block (tables and cells) pairs. Human annotators draw borders around targeted tables in scanned document images. This type of manual annotation is expensive to obtain, and the table positions are often inaccurate. Fortunately, with the development of deep learning in computer vision fields, many open-source pretrained models are available to address this issue. The one we adopt in our system is called CascadeTabNet (Prasad et al. [2020]), which is the latest end-to-end model with state-of-the-art performance on table detection with pretrained data on both bordered and borderless tables. This model is based on the Cascade mask R-CNN HRNet (Cai and Vasconcelos [2019]), a complex version of a CNN which stitches together the Cascade R-CNN and HRNet networks. We present more details on this method in Section 4.3.

3.6 Handwriting recognition

Once the handwritten text segments are isolated following the above procedure, we can apply an existing Application Programming Interface (API) for handwriting recognition to digitize the text information. Examples include the state-of-the-art model presented in Toiganbayeva et al. [2021], which achieves 93.48% accuracy on characters. We will

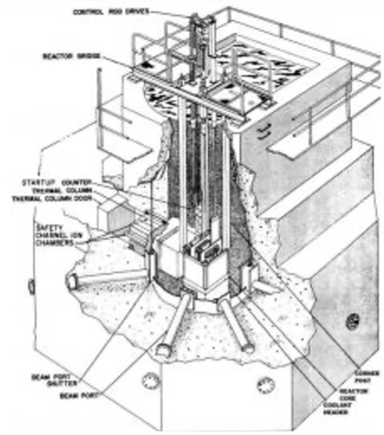


Figure 8: Reactor schematic diagram

not elaborate on this topic, as it is not the main focus of our study. We refer our audience to the survey in Memon et al. [2020] for a systematic review of state-of-the-art results and techniques for handwriting recognition.

4 Experiments

In this section, we apply the proposed methodology as described in Section 3 to the two types of documents (containing either bordered or borderless tables) collected from a real-world nuclear reactor and present both qualitative and numerical results.

4.1 Data description

The University of Wisconsin Nuclear Reactor (UWNR) is a 1 MW TRIGA (Training, Research, Isotope Production, General Atomics) reactor that has operated as a teaching and research reactor since 1961 (see Figure 8). Note that the reactor uses the record management system (RMS) required by the US Code of Federal Regulations, specifically 10 CFR 50.71.[2]

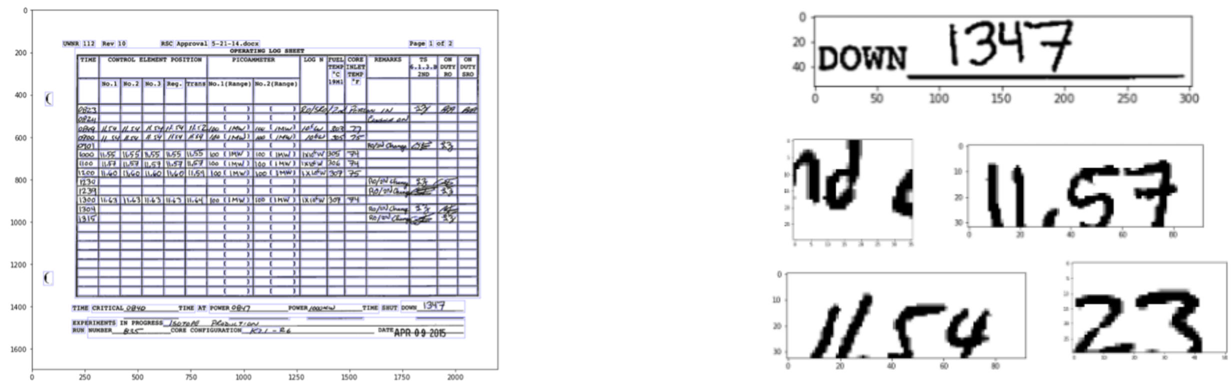
Our study involves two types of handwritten documents (one operation log and one checklist), collected from the UWNR over 40 operations from April 2015 to August 2018. The documents were acquired utilizing a Fujitsu ScanSnap iX500 scanner. The PDF is automatically produced using the ScanSnap Folder application with 1200dpi resolution for black and white images. Figure 1 shows the examples of these two types of handwritten documents used in our experiments. In particular, Figure 1(a) is an example of an operational log which is a bordered table (a table with vertical and horizontal lines that separate cells) and Figure 1(b) is that of a checklist which is a borderless table (a table without lines). The data include 98 pages of documents with bordered tables and 48 pages of documents with borderless tables.

4.2 Results for bordered tables

In this section, we discuss experimental results for applying our method to scanned images of bordered tables, as shown in Figure 1(a). Our goal for this type of document is to crop scanned images into small rectangular cell patches and then locate every cell. Here, we implement the traditional methods such as morphological operations, contour detection, and Hough line transforms with the Python library OpenCV-Python by Bradski [2000], as well as the neural network-based method, CascadeTabNet Prasad et al. [2020], with the MMDetection framework[3].

For contour detection, we first dilate the original image with a square kernel of size 15×15, and then obtain rectangle cell patches that bound handwritten texts with `cv2.findContours` and `cv2.boundingRect`, as in Figure 5. Figure 9 shows an example of the result on one scanned page using contour detection, and Figure 9(b) shows several examples of extracted cells. Although this method identifies some cells correctly, it makes errors when the handwriting within one cell is separated, and the accuracy greatly depends on the content within cells.

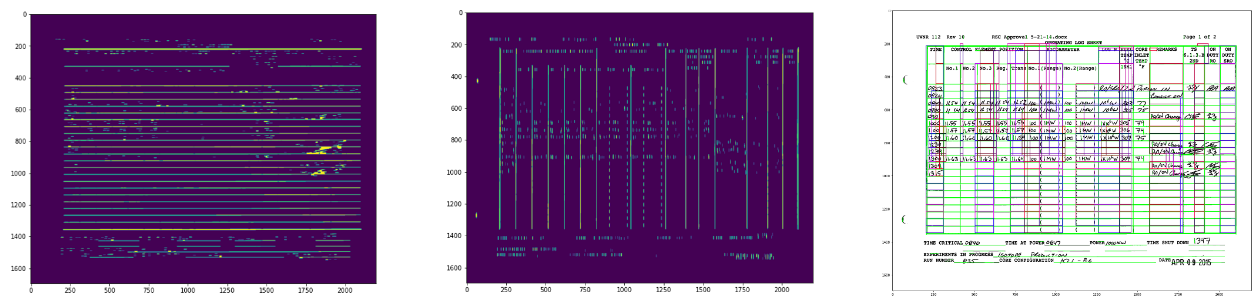
[2]https://www.nrc.gov/reading-rm/doc-collections/cfr/part050/part050-0071.html
[3]https://github.com/open-mmlab/mmdetection



(a) Results of contour detection

(b) Example of extracted contents

Figure 9: Detection of table cells with contour detection



(a) Horizontal line detection

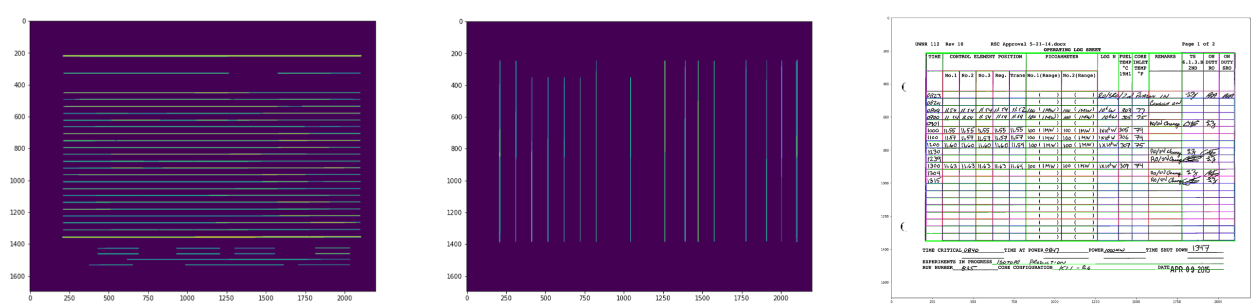
(b) Vertical line detection

(c) Results on images

Figure 10: Results without tuned parameters

The results of the Hough line transform are shown in Figure 10. The figure shows that Hough line transform detects letters (either attached or detached to boundaries) within cells as lines, thus making mistakes when we crop cell content. To address this issue, we tune the parameters in morphological masks to increase the length of those masks. By applying these tuned masks, we filter out shorter lines that are not table cell separators. The results after tuning the parameters are presented in Figure 11, which is more accurate that those in Figure 10. The Hough line transform is shown to outperform contour detection, as it is more robust to various cell content.

Since we do not have the exact coordinates of each cell box (ground truth), to present some numerical results, we use an alternative measure to evaluate the overall performance on scanned documents. In particular, we measure the number of intersection points, horizontal lines, and vertical lines as the ground truth, and compare these with the numbers detected by the algorithm. Table 1 presents the resulting average count and its standard deviation (in parentheses) in each of

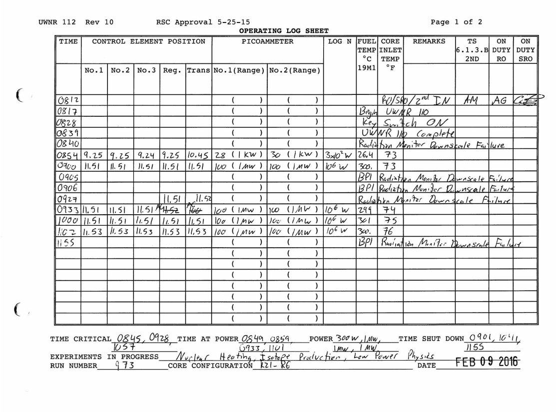


(a) Horizontal line detection

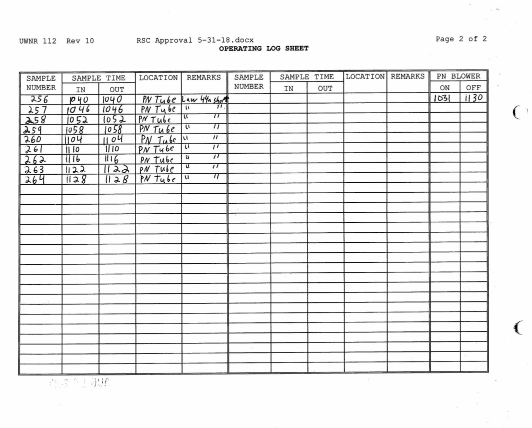
(b) Vertical line detection

(c) Results on images

Figure 11: Results after tuning parameters



(a) 112a



(b) 112b

Figure 12: Two types of bordered tables

Document type	Methods	Intersections	Horizontal lines	Vertical lines
112a	Proposed Method	356.66 ± 22.80	27.94 ± 0.48	15.77 ± 0.66
	Ground Truth	382	28	16
112b	Proposed Method	355.42 ± 23.30	30.44 ± 0.76	12.91 ± 0.53
	Ground Truth	390	30	13

Table 1: Results for bordered tables by document type

two types of bordered tables: Document 112a and Document 112b, as shown in Figure 12. Results of the proposed method are generated with Hough line transforms and parameter tuning. Documents 112a and 112b each consist of 49 pages. Table 1 shows the numbers of lines and intersections detected by our algorithm are close to the ground truth. In addition to that, we checked that the locations of the detected intersection points, horizontal lines, and vertical lines are well-aligned with their true locations through manual visual inspection.

For the neural network-based method, a set of training data is often required which contains ground truth about the exact locations of tables. To resolve the issue of not having annotated data, we use a pretrained model "ICDAR 19 (Track B2 Modern) table structure recognition" from Model Zoo of CascadeTabNet[4]. All pretrained models were trained on digital images of PDF files instead of handwritten scanned documents, which therefore contain less noise than our dataset. For bordered tables, CascadeTabNet can detect the outer boundary of the table but not the cells. The result of the pretrained model on our bordered table document is presented in Figure 13. This table border detection can be regarded as a filtering mask if we only want to extract information within a bordered table (in our case, the yellow rectangle in the plot). This filter is especially important if the table is surrounded by other texts. In all 98 pages, the model successfully detects the table borders.

4.3 Results for borderless tables

In this section, we present results for borderless tables as shown in Figure 1(b), using both the traditional method (template matching) and the neural network method.

For template matching, we present an example of the results in Figures 14 and 15. In order to better display the details, we show a zoomed-in version of the original document. we first compute correlation coefficients for the original image and template image, yielding the normalized correlation coefficient matrix shown in Figure 14(b). Results of applying different thresholding methods to binarize the document (i.e., pixel values above the threshold are replaced by 1 and those below the threshold are replaced by 0) are also shown in Figure 15. The larger the threshold, the fewer white dots we have in the plot, which correspond to possible positions of template centers. The location with maximum pixel value is selected as the position of the template. By choosing 7 template patches of header texts to locate 5 rows and 2 columns, we can identify locations of all cells.

[4]https://github.com/DevashishPrasad/CascadeTabNet#6-model-zoo

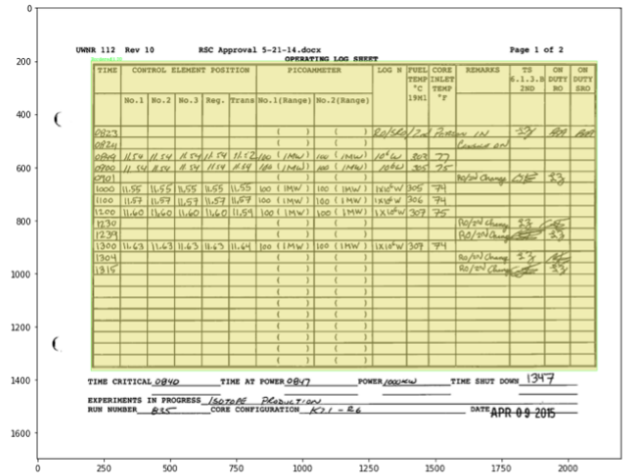
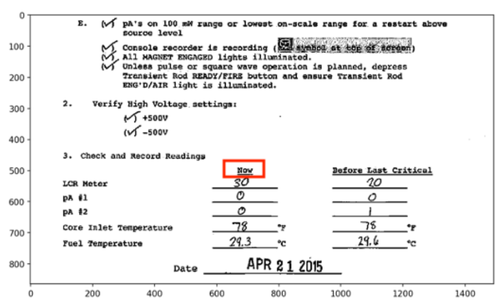
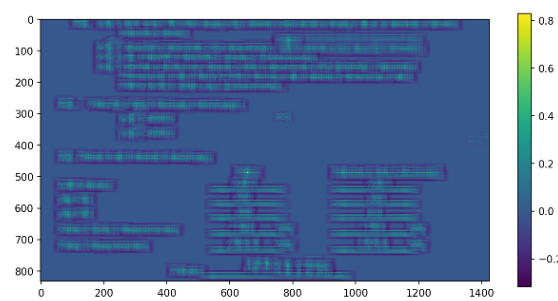


Figure 13: Table border detection with CascadeTableNet



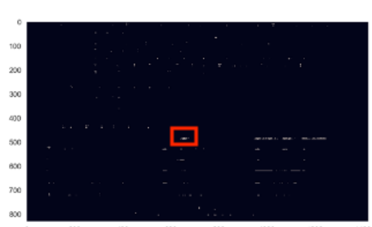
(a) Original image and template (highlighted in red rectangle)



(b) Normalized correlation coefficient matrix $\mathbf{R}$

Figure 14: Results of template matching

(a) Threshold 0.1



(b) Threshold 0.3

(c) Threshold 0.6

Figure 15: Correlation coefficient matrix with different threshold values

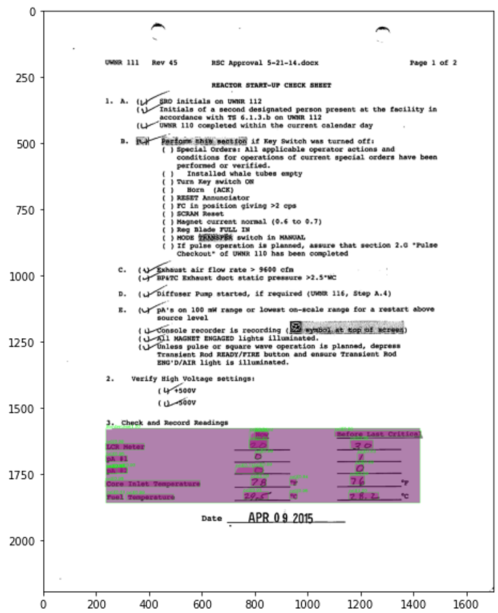


Figure 16: Table border and cell detection with CascadeTableNet

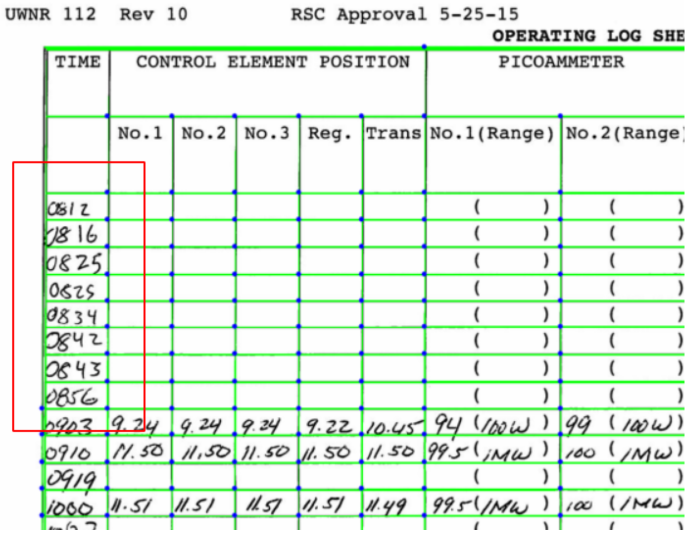
For the neural network-based method, we apply CascadeTabNet pretrained on borderless tables to the denoised scanned documents. The outputs of this neural network model are similar to the template matching methods by finding the location of each number (cell) as well as the outer boundary of the table from the document page. An example of the results is shown in Figure 16.

To obtain numerical results for borderless tables, we measure the number of errors in each document. There are 10 cells on each page and 48 pages in total. For template matching, we have 100% accuracy on cells, and for neural network methods, the average number of missed cells per page is 1.02, with a standard deviation of 0.97. In the following subsection, we will discuss more details about the results.

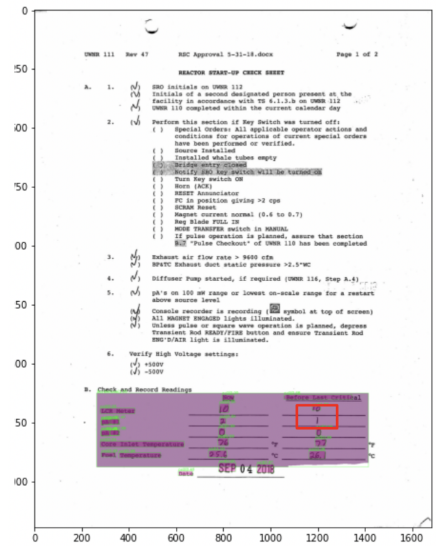
4.4 Discussion

For bordered tables, we observe that the number of intersection points detected is on average less than the ground truth, as in Table 1. This happens when the Hough line transform fails to detect a table border. For instance, Figure 17(a) presents an example where the Hough line transform fails to detect several intersections (blue dots) on the left-most vertical line, as it is a double line and handwriting is adjacent to table lines (highlighted in the red box).

In Section 4.3, we present results for borderless tables with both template matching and neural network methods. Compared to bordered tables, borderless tables are in general more difficult to detect, as the structure is not as clear as in bordered tables. Template matching has higher accuracy; however, it can only be applied to a certain type of document. Here, the certain type of document means the handwritten filled document from the same printed table with the same template. For example, in Figure 16, the handwritten numbers could be different, but the column and row names are fixed. Thus, the templates used in template matching remain the same for such type of document. For each new type of document, human effort must be expended to manually select templates. Results for the neural network-based method with pretrained models are not as good as template matching, but neural network methods do not need any templates and can be applied directly to any documents. Figure 17(b) shows an example of errors that may be incurred when the content in a cell is "slim" (highlighted in the red box), since the neutral network may fail to detect it as a cell. Template matching has the drawback of requiring human work on selecting and cropping templates, thus is recommended in cases when there are limited types of handwritten reports and human operators can easily process the documents manually. If many reports are filled out of the same blank table document, template matching is very efficient. On the other hand, the neural network-based method has the drawback of requiring annotated data. This method is preferable when there is a large amount of labeled training data available which may be used to train a reliable neural network, or, in our case, a pretrained neural network with a similar dataset.When the proposed method is employed in other real-world cases, it could be validated by examining random samples of outputs and errors like in Figure 17. It could also be validated by looking at the values contained in cells after handwritten recognition.



(a) Errors in Hough line transform



(b) Errors in neural network method

Figure 17: Examples of errors

5 Conclusions and future work

In this study, we have proposed a systematic digitization framework for table-structured handwritten documents collected from NPPs. To the best of our knowledge, this is the first study focusing on digitizing NPP documents. In order to extract useful information from both bordered and borderless tables, we applied traditional computer vision methods and convolutional neural networks. For bordered table, we adopted Hough line transforms and conducted parameter tuning to detect desired areas. For borderless tables, as the document types are limited for NPP documents (all of the pages are from the same template and only the handwritten parts are different), template matching was shown to outperform the neural network-based method. In numerical studies involving a real handwritten operational log dataset, the proposed combination of neural network methods and traditional computer vision methods yielded good results for both bordered and borderless tables.

This paper is a companion to a broader study focused on predictive health of plant components. The overall goal is to enhance the capability to predict when a plant component (for instance, a critical pump) would reach a failure point. Because important operational data for nuclear plant operations are captured in handwritten logs, this study advanced the ability to rapidly digitize handwritten logs so that operator data can be included with digital data coming from sensors. This study could potentially benefit plant operators to extract information from handwritten records, and further, to construct data-driven predictive models for plant components' health.

There are still drawbacks of our methods and while a mistake occurs in table detection or handwritten recognition, it would result in a potential outlier for the study of our data. Several topics exist for future works to eliminate those drawbacks: One possible research direction is to improve the performance of table detection. Our current pipeline is using manually annotated documents and finetuning CascadeTabNet with specific data. In addition, as the tables are handwritten, there can be text written across multiple cells. One might merge adjacent cells to address such situation. The other research direction is to improve the handwritten recognition part of our pipeline, which has not been addressed in detail in our current research. Although the state-of-the-art recognition accuracy of handwritten recognition is reasonably high, there is still a possibility that the OCR API algorithms misread handwritten numbers. To eliminate this problem, we propose three steps after the handwritten API. First, use domain knowledge and outlier detection algorithms on the output number. For example, the method may alarm users when the digitized fuel temperature is negative. We could also apply outlier detection to alarm users when the number from one document is very different from those from other documents. Second, set a threshold on the confidence of the output number. The handwritten API will output the prediction confidence for each specific number. The system may automatically filter out the numbers with very low prediction confidence. Third, although the API was trained on handwritten numbers in many different styles, it is possible that the recorder wrote in a style that is not in the dataset and difficult to recognize. We could also finetune the API by adding a small dataset of numbers written specifically by the operators to learn their unique handwriting styles.

Acknowledgment

The authors acknowledge the funding support in part by the Department of Energy under award number DE-NE0008805 and award number DE-NE0008993.

References

G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: High quality object detection and instance segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

Zhiguang Deng, Qian Wu, Xin Lv, Biwei Zhu, Sijie Xu, and Xuemei Wang. Application Analysis of Wireless Sensor Networks in Nuclear Power Plant. In Yang Xu, Yongbin Sun, Yanyang Liu, Yanjun Wang, Pengfei Gu, and Zheming Liu, editors, *Nuclear Power Plants: Innovative Technologies for Instrumentation and Control Systems*, pages 135–148. Springer Singapore, 2020. ISBN 978-981-15-1876-8.

Richard O Duda and Peter E Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

Basilios Gatos, Dimitrios Danatsas, Ioannis Pratikakis, and Stavros J Perantonis. Automatic table detection in document images. In *International Conference on Pattern Recognition and Image Analysis*, pages 609–618. Springer, 2005.

IAEA. *Advanced Surveillance, Diagnostic and Prognostic Techniques in Monitoring Structures, Systems and Components in Nuclear Power Plants*. Number NP-T-3.14 in Nuclear Energy Series. INTERNATIONAL ATOMIC ENERGY AGENCY, Vienna, 2013. ISBN 978-92-0-140510-4.

Sekhar Mandal, SP Chowdhury, Amit K Das, and Bhabatosh Chanda. A simple and effective table detection system from document images. *International Journal of Document Analysis and Recognition (IJDAR)*, 8(2):172–182, 2006.

Jamshed Memon, Maira Sami, Rizwan Ahmed Khan, and Mueen Uddin. Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr). *IEEE Access*, 8:142642–142668, 2020.

Robert Jay Milewski, Venu Govindaraju, and Anurag Bhardwaj. Automatic recognition of handwritten medical forms for search engines. *International Journal of Document Analysis and Recognition (IJDAR)*, 11(4):203–218, 2009.

Shubham Singh Paliwal, D Vishwanath, Rohit Rahul, Monika Sharma, and Lovekesh Vig. Tablenet: Deep learning model for end-to-end table detection and tabular data extraction from scanned document images. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 128–133. IEEE, 2019.

Devashish Prasad, Ayan Gadpal, Kshitij Kapadni, Manish Visave, and Kavita Sultanpure. Cascadetabnet: An approach for end-to-end table detection and structure recognition from image-based documents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 572–573, 2020.

Shah Rukh Qasim, Hassan Mahmood, and Faisal Shafait. Rethinking table recognition using graph neural networks. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 142–147. IEEE, 2019.

Ahmad Al Rashdan and Shawn St. Germain. Methods of data collection in nuclear power plants. *Nuclear Technology*, 205(8):1062–1074, 2019.

Sebastian Schreiber, Stefan Agne, Ivo Wolf, Andreas Dengel, and Sheraz Ahmed. Deepdesrt: Deep learning for detection and structure recognition of tables in document images. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 1162–1167. IEEE, 2017.

Faisal Shafait and Ray Smith. Table detection in heterogeneous documents. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, pages 65–72, 2010.

Satoshi Suzuki et al. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

Nazgul Toiganbayeva, Mahmoud Kasem, Galymzhan Abdimanap, Kairat Bostanbekov, Abdelrahman Abdallah, Anel Alimova, and Daniyar Nurseitov. Kohtd: Kazakh offline handwritten text dataset. *arXiv preprint arXiv:2110.04075*, 2021.