Systems & Control Transactions

Research Article - Peer Reviewed Conference Proceeding
ESCAPE 35 - European Symposium on Computer Aided Process Engineering
Ghent, Belgium. 6-9 July 2025
Jan F.M. Van Impe, Grégoire Léonard, Satyajeet S. Bhonsale,
Monika E. Polańska, Filip Logist (Eds.)

PSE PRESS

Automated Interpretation of Chemical Engineering Diagrams Using Computer Vision

Maged Eid[a], Giancarlo Dalle Ave[b]

[a] W Booth School of Engineering Practice and Technology.
[b] Chemical Engineering Department, McMaster University, Canada.
Email addresses: eidm6@mcmaster.ca, dalleagf@mcmaster.ca

ABSTRACT

This paper presents state-of-the-art object detection and object identification algorithms for digitizing and interpreting chemical engineering diagrams, including, Block Flow Diagrams (BFDs), Process Flow Diagrams (PFDs), and Piping and Instrumentation Diagrams (P&IDs), using computer vision techniques. These diagrams are essential for visualizing plant processes and equipment but are often stored as image-based PDFs, making manual digitization/interpretation labor-intensive and error-prone. The proposed algorithm automates tasks such as detecting unit operations and identifying them using a set of rule-based and predefined approaches including edge and contour-based rules, spatial arrangement rules, and geometric rules. This method avoids data requirements and computational requirements of deep learning approaches, offering a scalable and efficient solution for preliminary extraction of complex process information.

Keywords: Chemical Engineering Diagrams, Object Detection, Object Identification, Computer Vision for Chemical Engineering, Optical Character Recognition (OCR), Features Extraction in Diagrams.

1. INTRODUCTION

This paper introduces an object detection and identification system designed to interpret chemical engineering diagrams including Block Flow Diagrams (BFDs), Process Flow Diagrams (PFDs), and Piping and Instrumentation Diagrams (P&IDs), which are essential for visualizing material flow, energy transfer, and equipment in chemical processes. The process of manually interpreting these diagrams often requires significant effort, as it is both labor-intensive and susceptible to errors. This not only increases the time and resources needed for analysis but also raises the likelihood of inaccuracies [1]. To address this, the proposed algorithm leverages computer vision techniques to detect key unit operations and identify them based on textual elements. By automating these tasks, the system aims to improve accuracy and speed, reducing reliance on expert analysis and enhancing process design workflows.

A BFD is a simplified representation of a process that outlines the main unit operations and their flow in a highly abstract way. BFDs are often used in the early stages of process design to visualize the overall structure of a process without focusing on details. PFDs, on the other hand, are more complex and detailed than BFDs. They display the relationships between specific unit operations and the flow of materials, PFDs are crucial for understanding the interconnectedness of unit operations and for simulating both individual operations and the entire process. The most detailed level is the P&ID, which reflects the "real" operational state of a process. P&IDs go beyond PFDs to include every piece of equipment, instrument, and control detail, providing a comprehensive view of how the process will operate in practice. The goal is to create a single multipurpose algorithm that can digitize key information by extracting features from these diagrams including automated detection and identification of unit operations, which is the focus of this work.

Predefined computer vision approaches use fixed rules or templates to recognize patterns or symbols in diagrams, offering fast and reliable detection in standardized industries with consistent representation of unit operations, flow paths, and equipment. However, they may struggle with low-quality diagrams, and non-standard, or irregular symbols [2], such as those often present in chemical engineering diagrams. Rule-based approaches, such as edge and contour-based rules, spatial arrangement rules, and geometric rules, leverage domain-

specific knowledge to define tailored rules for detecting and identifying objects, thereby improving detection accuracy. By combining expert insights with predefined, rule-based methods, these approaches ensure flexibility and reliability across diverse scenarios [3]. The proposed algorithms introduce an approach for automating the detection and identification of unit operations in BFDs, PFDs, and P&IDs. Unlike deep learning methods, it does not require manual labeling, making it more easily implementable and adaptable [4][5][6]. The algorithm leverages predefined rules and rule-based approaches to recognize patterns and symbols, enabling consistent performance across diverse and non-standard representations. This approach enhances accuracy and flexibility while streamlining workflows, addressing limitations of past attempts that relied heavily on manual-labor-intensive methods.

2. PROJECT METHODOLOGY

This section outlines the workflow for processing chemical process diagrams, from test data acquisition and preprocessing to object detection and identification. Key steps include object detection to isolate elements like unit operations, using OCR to extract labels, and combining these for unit identification.

2.1 Data Collection and Preprocessing

The dataset used in this study comprises 100 high-quality diagrams, including PFDs, BFDs, and P&IDs, sourced from various online resources. The primary selection criterion was image clarity and resolution, ensuring minimal noise or distortion. This diverse dataset allowed for comprehensive testing of the model's robustness across diagram types, with a Git repository link provided for access [7].

Performance evaluation focused on object detection and identification efficiency, analyzing metrics such as processing times for the 100 images. To assess the accuracy, a subset of 50 images was manually validated, covering a wide variety of diagrams and characteristics (e.g., different unit operations and types of diagrams). Object detection accuracy was determined by comparing manually counted objects (e.g., unit operations, control loop symbols) with detected ones, while object identification accuracy was validated using domain knowledge of the unit operations and compared to the textual labels assigned by the algorithms.

2.2 Algorithm Development

The algorithms presented in this paper aim to process and analyze images by detecting and identifying the main objects within the images. It follows a series of steps designed to identify regions of interest, remove unwanted noise, and highlight the main unit operations in the image. The process includes multiple image processing techniques such as edge detection, contour detection, and the use of OCR for text recognition to effectively detect and identify the different chemical unit operations within the different diagrams.

Object detection and identification focuses on isolating key features and objects within images and labeling them using text in the images, with OpenCV being one of the tools employed for this process [8]. Figure 1 presents pseudo code for the Object Detection algorithm which begins with Canny edge detection to enhance critical boundaries (line 6) [9], followed by contour detection to isolate individual objects based on shape, area thresholds, aspect ratio, and solidity (line 12) [10]. To clean and refine the mask of detected objects, morphological operations like opening and dilation are applied, removing noise, and closing gaps in contours (line 19) [11]. For separating connected components, a distance transform and the watershed algorithm are used, enabling precise segmentation, even in complex scenarios where objects overlap [12].

Algorithm 1 Image Processing and Contour Detection

```
1  Input: image_dir (directory containing image files)
2  Output: processed images with highlighted contours and separated objects
3  function DETECT_EDGES(image)
4      convert the input image to grayscale
5      smooth the image using Gaussian blur to reduce noise
6      apply Canny edge detection to identify prominent edges
7      return edge-detected image
8  end function
9  function PROCESS_IMAGE(image_path)
10     load the image from the provided file path
11     edges ← DETECT_EDGES(image)          ▷ identify edges in the image
12     contours ← find contours in the edge-detected image
13     for each contour do
14         if the contour meets size and shape criteria then
15             highlight the contour on a separate mask image
16         end if
17     end for
18     use morphological operations to refine the mask and remove noise
19     apply watershed or similar techniques to distinguish overlapping objects
20     return the original image with contours and the refined mask
21 end function
22 function MAIN(image_dir)
23     for each image in image_dir do
24         processed_image, mask ← PROCESS_IMAGE(image)
25     end for
26 end function
```

Figure 1. Pseudocode for the object detection algorithm

Figure 2 presents the pseudocode for the Object Identification algorithm. PaddleOCR is employed to extract textual elements from diagrams, while the initial Object Detection algorithm identifies the unit operations [13]. Small objects detected next to one another are merged to account for the possibility that they are different parts of the same object (lines 4-6). The Euclidean distance between the detected text and the detected unit operations is calculated to establish associations, enabling accurate mapping of labels to unit operations. This method allows for localization and detection of components across chemical engineering diagrams.

Algorithm 2 Object Identification

```
Input: image directory
Output: processed images with identified objects
initialize PaddleOCR model
MERGE_BOUNDING_BOXESbox1, box2
merge boxes by computing min/max coordinates
ARE_BOXES_CLOSEbox1, box2, threshold
return true if distance between centers is below threshold
MERGE_NEARBY_BOUNDING_BOXESword_boxes, threshold
merge nearby boxes based on threshold
DETECT_EDGESimage
apply Canny edge detection after grayscale conversion and blur
PROCESS_IMAGEimage_path, word_boxes
edges ← DETECT_EDGES(image)
contours ← findContours(edges)
word_boxes ← MERGE_NEARBY_BOUNDING_BOXES(word_boxes, threshold)
for each contour in contours do
  if contour area > threshold then
    process contour and match with word boxes
  end if
end for
morph mask, generate main object image
return original_image, main_object, words_cleaned_image
MAINimage_dir
for each filename in image_dir do
  if filename ends with .jpg or .png then
    result ← OCR.OCR(image_path)
    word_boxes ← extract from result
    original_image ← PROCESS_IMAGE(image_path, word_boxes)
    main_object ← PROCESS_IMAGE(image_path, word_boxes)
    words_cleaned_image ← PROCESS_IMAGE(image_path, word_boxes)
  end if
end for
```

Figure 2. Pseudocode for the object identification algorithm

3. RESULTS AND DISCUSSION

Table 1 shows the computation times for the object detection and object identification algorithms. Overall, the algorithm is fast, offering quick initial results which can speed up digitization workflows.

Table 1. Accuracy Distribution for Object Detection and Object Identification.

Metric	Object Detection	Object Identification
Average Time per Image (s)	0.38	4.86
Minimum Time (s)	0.24	2.23
Maximum Time (s)	0.68	15.15
Standard Deviation of Time (s)	0.1	2.33

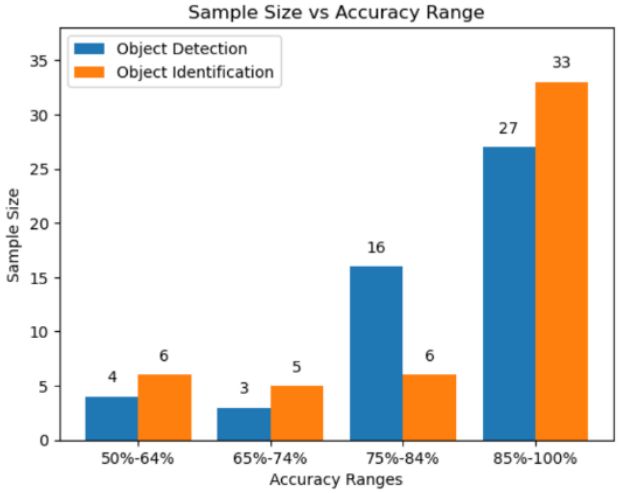


Figure 3. Cumulative Time Metrics for Object Detection and Object Identification

Figure 3 summarizes the performance metrics from the manual validation phase of the computer vision models, evaluating accuracy for object detection and identification across the dataset. These results demonstrate the model's effectiveness in detecting and identifying objects within the diagrams. The model achieved strong performance across different accuracy ranges, with the highest sample sizes in the 85%-100% accuracy range. The reason that object identification can have a higher score than object detection is that often multiple objects are mistakenly grouped together (lowering the detection score), however, when analyzing the text of the images multiple labels are assigned to this aggregated space identifying that those operations are all within that space. This evaluation offers insights into the model's consistency for object detection and identification tasks. Specific examples of results are shown below to highlight features of the algorithms' performance as well as situations where the algorithm could be improved.

3.1 Object Detection

Figure 4 displays the original and processed images from a BFD. Subfigure 4A shows the original image, while Subfigure 4B presents the results of object detection applied to the same image. The contours and masks are overlaid on the original image, illustrating how the system isolates and identifies key objects. The detected contours are visible, highlighting the effectiveness of the edge detection and contouring methods used. In this example, the accuracy of Object Detection is 100% as it captured all 9 objects within the diagram.

Another example, Figure 5, displays the original and processed images from a PFD. Subfigure 5A shows the original image, while Subfigure 5B highlights the accuracy of the object detection algorithm in handling different object shapes and sizes within the PFD. This visual representation confirms that the system can effectively

manage variations in object appearance and image noise. Some of the drawbacks of the algorithm are visible here as well. The crossing of the streams exiting the central heat exchanger form a closed loop which the algorithm mistakenly identified as belonging to the heat exchanger.

Figure 6 displays the original (A) and processed images (B) from a P&ID. This subfigure demonstrates the algorithm's capability to identify the control aspects of the P&ID and delineate components specific to P&ID diagrams. However, it lacks the ability to capture certain unit operations, such as valves, which will be addressed using template matching techniques in future work. The assigned accuracy of this diagram is 54% as it identifies 19 control icons but misses all 16 valves present in the diagram. P&IDs consistently showed the worst performance in the dataset reflecting their increased complications relative to the other diagrams. Despite some shortcoming, the algorithm shows great flexibility to quickly (<0.5s per image on average) and accurately detect process operations across a wide variety of diagrams with no additional fine tuning required to transition between document types.

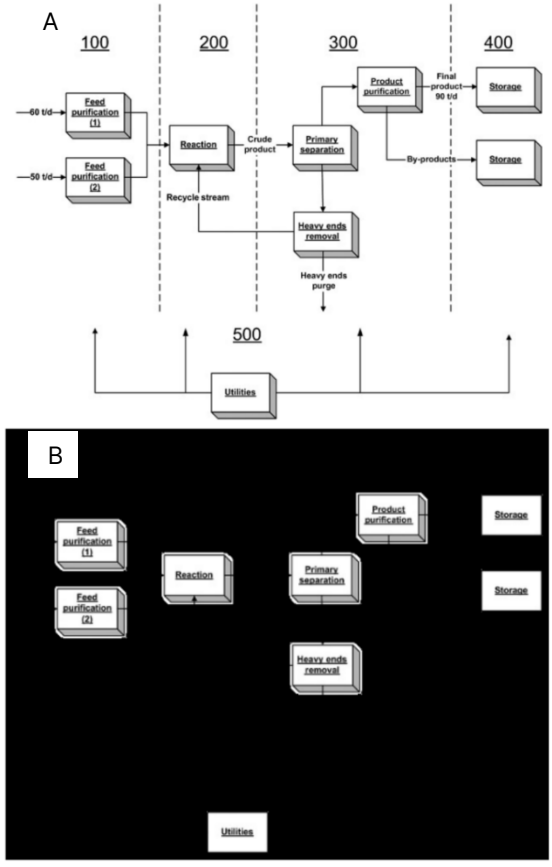


Figure 4. (A) Original image of a BFD. (B) Objects detection algorithm output

3.2 Object Identification

The object identification algorithm is based on spatial proximity, associating each detected object with its nearest detected word by calculating the Euclidean distance between their center points. In the visual representation (Figures 7 & 8), red dots represent detected words, green dots represent detected objects, and the blue lines are drawn between them to show their assignments. These lines visually indicate which word is linked to which object, based on how close their center points are to each other. A simple example result is shown in Figure 7. In this case, the algorithm clearly connects the text inside the boxes with the boxes themselves. An example of the identification algorithm applied to a PFD can be seen in Figure 8. In this case the object identification algorithm can connect text to label adjacent objects. Lines are drawn between the text box and each of the sub objects which are merged within a unit operation (e.g., each of the tubes in a heat exchanger) to determine which object the label is most likely to belong to. The drawback to the algorithm is that if a unit operation label is too far from the object, it is not connected to it (for example the absorber column on the right). Also, textual elements on the diagrams are often encoded with cryptic names (e.g., REC for the heat exchangers). In practice, these names are often encoded in a standard format and could be converted to more meaningful names with a dictionary.

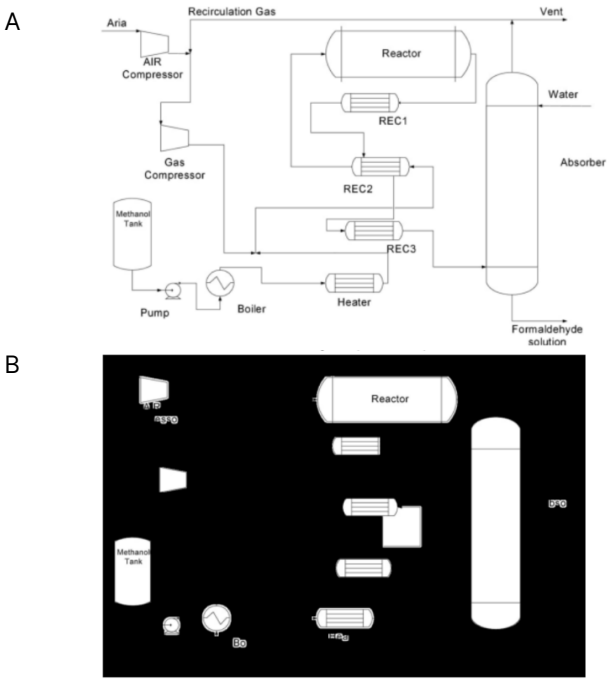


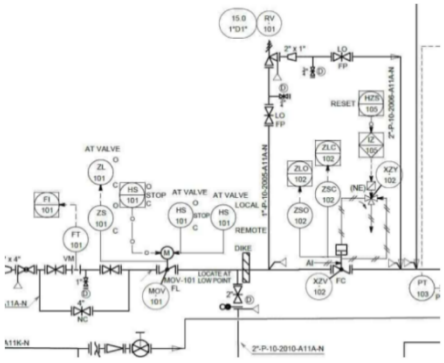
Figure 5. (A) Original image of a PFD. (B) Objects detection output from the PFD.

The object identification results demonstrate the algorithm's ability to accurately detect text and associate

it with objects in complex images. High-precision OCR effectively extracts text, with an internal confidence level for reliability of 100%. However, manual validation reveals that in some cases, there may be slight discrepancies between the automated results and the actual text (e.g., chemical formulae containing numbers can get mistaken for text which is “read” incorrectly). One of the major drawbacks to the algorithm is that object identification is based on textual elements in the diagrams, which are often encoded with company- or site-specific naming conventions.

Overall, the system demonstrates fast and accurate detection and text-based identification capabilities, streamlining the preliminary interpretation of chemical engineering diagrams and enhancing workflow efficiency. Compared to deep learning approaches, the proposed algorithm is less computationally intensive, enabling efficient large-scale preliminary data processing. The code processed 100 images with an average of 0.38 seconds for object detection and 4.86 seconds for object identification (on a i7-12700H laptop with 16 GB of RAM and no GPU-based calculations), demonstrating its ability to efficiently handle tasks of varying complexity within a limited timeframe, demonstrating it can help to streamline preliminary digitization workflows at minimal computational expense [14].

A



B

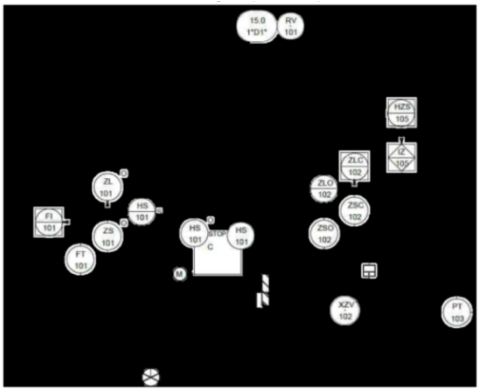


Figure 6. (A) Original image of a P&ID. (B) Objects detection output from the P&ID.

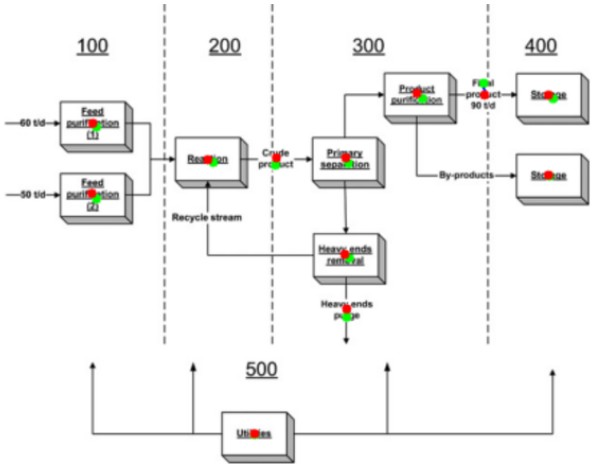


Figure. 7. BFD object identification: Red dots mark detected words, green dots mark detected objects, and blue lines show connections between them.

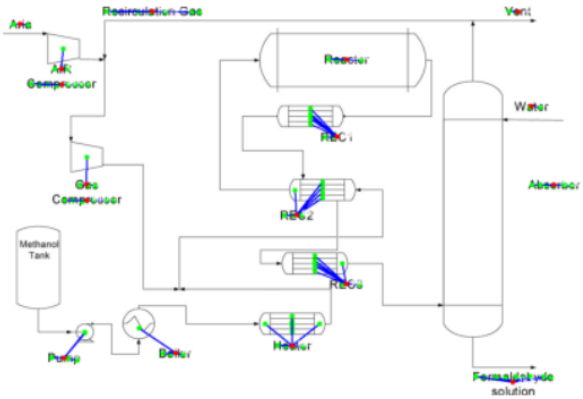


Figure 8. PFD object identification: Red dots mark detected words, green dots mark detected objects, and blue lines show connections between them.

4. CONCLUSION AND FUTURE WORK

This work presents an object detection and identification system tailored for chemical engineering diagrams, leveraging a combination of rule-based approaches, text extraction, and spatial association. The system quickly and efficiently detects unit operations, establishes connections between text and unit operations to identify objects, and is easily applicable to diverse diagram types, including BFDs, PFDs, and P&IDs. By eliminating the need for manual labeling and relying on predefined rules, the approach is quite flexible streamlining the preliminary interpretation of complex chemical engineering diagrams. Automatically interpreting chemical engineering diagrams reduces the need for manual analysis, significantly saving time and effort. It can improve accuracy by minimizing human errors in identifying process components.

This automation speeds up decision-making,

allowing engineers to focus on optimization rather than diagram interpretation. While the algorithm shows promise, results highlight areas for improvement. Without knowledge of unit operations, it sometimes misidentifies accessory shapes as objects (e.g., recycle loops forming closed shapes). Small objects like valves are often mistaken for noise and filtered out. The object identification relies on linking textual labels to nearby detected objects, but labels often follow site- or application-specific naming conventions.

The next steps involve refining rule-based and predefined approaches, using template matching for smaller objects, and enhancing preprocessing with multi-resolution bounding boxes. Improving OCR with multi-language support, adaptive thresholding, and noise reduction will boost text recognition and object detection. Additionally, the generated data can help integrate deep learning models, further enhancing detection and identification.

REFERENCES

1. Waller, A., LeDoux, J., & Newstetter, W. What makes an effective engineering diagram? a comparative study of novices and experts.. https://doi.org/10.18260/1-2--22751
2. Yang, L., Zhang, J., Li, H., Ren, L., Yang, C., Wang, J., & Shi, D. (2024). A comprehensive end-to-end computer vision framework for restoration and recognition of low-quality engineering drawings. Engineering Applications of Artificial Intelligence, 133, 108524. https://doi.org/10.1016/j.engappai.2024.108524
3. A rule-based approach for visual pattern inspection. (1988, January 1). IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/3867
4. Moon, Y., Lee, J., Mun, D., & Lim, S. (2021). Deep Learning-Based Method to Recognize Line Objects and Flow Arrows from Image-Format Piping and Instrumentation Diagrams for Digitization. Applied Sciences, 11(21), 10054. https://doi.org/10.3390/app112110054
5. Deep learning for text detection and recognition in complex engineering diagrams. (2020, July 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9207127
6. Kim, H., Lee, W., Kim, M., Yoochan, M., & others. (2021). Deep-learning-based recognition of symbols and texts at an industrially applicable level from images of high-density piping and instrumentation diagrams. Expert Systems with Applications, 183, 115337. https://doi.org/10.1016/j.eswa.2021.115337
7. DalleAveLab. (n.d.). Computer vision chemical simulation: Images. GitHub. https://github.com/DalleAveLab/ComputerVisionChemicalSimulation/tree/main/images
8. Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.
9. Canny, J. (1986). A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 8(6), 679–698. https://doi.org/10.1109/TPAMI.1986.4767851
10. Suzuki, S., & Abe, K. (1985). Topological structural analysis of digitized binary images by border following. Computer Vision, Graphics, and Image Processing, 30(1), 32–46. https://doi.org/10.1016/0734-189X(85)90016-7
11. Haralick, R. M., Sternberg, S. R., & Zhuang, X. (1987). Image analysis using mathematical morphology. IEEE Transactions on Pattern Analysis and Machine Intelligence, 9(4), 532–550. https://doi.org/10.1109/TPAMI.1987.4767941
12. Beucher, S., & Meyer, F. (1993). The morphological approach to segmentation: The watershed transformation. In Mathematical Morphology and Its Applications to Image Processing (pp. 433–441).
13. Du, Y., Li, C., Guo, R., Yin, X., Liu, W., Zhou, J., Bai, Y., Yu, Z., Yang, Y., Dang, Q., & Wang, H. (2020). PP-OCR: A Practical Ultra Lightweight OCR System. arXiv preprint arXiv:2009.09941. Available at: https://arxiv.org/abs/2009.09941 | PaddleOCR Contributors. (2023). PaddleOCR: Awesome multilingual OCR toolkits based on PaddlePaddle. GitHub repository. Available at: https://github.com/PaddlePaddle/PaddleOCR
14. Theisen, M. F., Flores, K. N., Balhorn, L. S., & Schweidtmann, A. M. (2023). Digitization of chemical process flow diagrams using deep convolutional neural networks. Digital Chemical Engineering, 6, 100072. https://doi.org/10.1016/j.dche.2022.100072

© 2025 by the authors. Licensed to PSEcommunity.org and PSE Press. This is an open access article under the creative commons CC-BY-SA licensing terms. Credit must be given to creator and adaptations must be shared under the same terms. See https://creativecommons.org/licenses/by-sa/4.0/