Automatic Information Extraction from Piping and Instrumentation Diagrams (P&ID)

Jiandao Zhu (jiandaoz@gatech.edu)

July 2020

Abstract

A piping and instrumentation (P&ID) data on offshore platforms is presented as connection point, often stored in PDF format. In order to automate equipment association of digital data, the P&ID information needs to be efficiently digitized for purposes of inventory management and update, and easy reference to different components of the schematics. This project will target the digitization of P&ID systems into graph representations using state-of-the-art deep learning networks as well as image processing techniques. The developed algorithm is tested using a dataset of simple P&ID illustration and achieved satisfactory results.

1 Introduction

A piping and instrumentation diagram (P&ID) is a detailed diagram in the process industry that shows the piping and process equipment together with the instrumentation and control devices. Over the years, P&ID diagrams have been manually generated, scanned, and stored as image files. Recently, artificial intelligence (AI) technology and big data science have been developing rapidly. As a result, digitization of data in a scanned image or hard-copy format is required for the application of technology to the plant operations worldwide. This enables a virtual plant model - so-called digital twin which is possible to locate all important data in mere seconds. Compared with traditional plant documentation, a plant digital twin is much easier to comply with operator obligations, accelerate plant modifications and allow for entirely new efficiency enhancement applications through a connection to the Internet of Things (IoT).

However. the conversion of plant documents into a digital form is costly in terms of both time and money. To improve the productivity, automatic drawings digitization is studied by a lot of researchers which aiming to eliminate the simple, repetitive tasks of manually extract components with different techniques. Generally, processing and analyzing these drawings is similar to any typical image-processing task, where the aim is to find the object of interest and then classify these objects. However, there are several unique challenges to digitize P&ID as follows:

- It is estimated that a single P&ID drawing contains numerous shapes including text, symbols, lines, etc. Representing a single section of a plant could require 100 to 1000 drawings.
- Likely, the digitization process involves a combination of several ML and shape recognition algorithms. The overall accuracy is dependent on the accuracy of each small step.
- Engineering standards are not hard fixed rules. There are different representations of the same component. For example, Figure 1 shows three possible representations of a ball valve which could occur in different drawings. A person with certain engineering knowledge would know that they are the same, but this may become problematic if the developed algorithm is not robust enough. The way of positioning the text/shape could also be different within the same drawing.
- Sometimes, mistakes are unacceptable because of the criticality of the drawings. Therefore, most of the developed process is still a human-centered approach that requires human involvement to review the results.

Figure 1: Three possible representations of a ball valve.

2 Related Work

Engineering drawings are very common in many industries, such as oil and gas, construction, planning, etc. These drawings can be defined as schematic representations, which depict the constitution of a circuit, device, process, or facility. Some examples of these drawings include logical gate circuits, mechanical or architectural drawings, P&ID drawings. There is an increasing demand in different industries for developing digitization frameworks to process and analyze these diagrams. Having such a framework will provide a unique opportunity for relevant industries to make use of large volumes of diagrams in informing their decision-making process and future practices.

In the 1990s and 2000s. The approach developed at that time was mainly based on vectorization. Lu [5] developed a rule-based text/graph separation approach to erase nontext regions from mixed text and graphics engineering drawings, Lu's algorithm could extract Chinese and English, dimensions, and symbols, but the accuracy performance was limited by the quality of the drawing and noise level. Luo and Liu [6] presented another framework to recognize engineering drawings using a case-based approach based on geometric constraints. The method was efficient and effective to learn geometry from given examples, but it required customization for different component types and it was challenging to extend to a "one size fits all" solution.

Since 2010, researchers have begun to use state-of-art AI tools, such as deep neural network and image processing techniques as well as pattern recognition technologies to recognize text, lines, and symbols from scanned images or PDFs. Fu et al. [2] utilized Convolutional Neural Network (CNN) as a symbol recognizer and the connectivity analyzer between line and symbol to convert network-like, image-based engineering diagrams into engineering models. Elyan et al. [1] presented a semi-automatic and heuristic-based approach to detect and localize symbols within these drawings. It included generating a labeled dataset from real-world engineering drawings and investigating the classification performance of three different supervised machine learning algorithms random forests (RF), support vector machine (SVM), and CNN. The results indicated that CNN performs better than the rest of the two on the original data set.

In recent years, more complicated real-world data sets have been tested and the development efforts have been shifted towards a production mode under software environment. Rahul et al. [7] used a combination of traditional vision techniques and state-of-the-art deep learning models to identify and isolate pipeline codes, inlets, and outlets and to detect symbols and texts. The extracted information could be further populated to a tree-like data structure for capturing the structure of the piping schematics. Overall, it achieved more than 90% accuracy for pipeline-code detection and some of the shape detection. But pipeline detection and object association tends to have less performance. Kang et al. [4] developed a prototype software tool "ID2" to digitize the P&ID. It used a sliding window method to recognize and extract line which achieves a better recognition rate. All recognition result is generated and stored as XML format.

3 Overview

In this paper, a novel P&ID digitization process is proposed which is shown in Figure 2. This process could be mainly divided into three parts: shape and text detection, line detection, and graph generation. Shape detection utilizes different object detection approaches including contour detection, Hough circle transform, text detection, and template matching to identify inlet/outlet, square, circle, text, and other symbols (such as reducer,valve, etc). Then the drawings are masked based on the location of the shapes and text. Then a line detection and merging algorithm is applied based on Hough line transform. All those identified text regions and shapes are associated and stored together based on their relative position. Finally, a network graph is generated to visualize the results.

All methods are coded using Python and its associated library under a modularized structure. Due to the limited time and large amount of potential work, the main purpose of the project is to deliver a proof of concept (POC) to digitize several simple P&ID by automating equipment association using state-of-art deep learning networks and image

processing techniques. The approach is mainly based on Rahul et al.' work [7]. Several improvements are identified and implemented based on his approach.

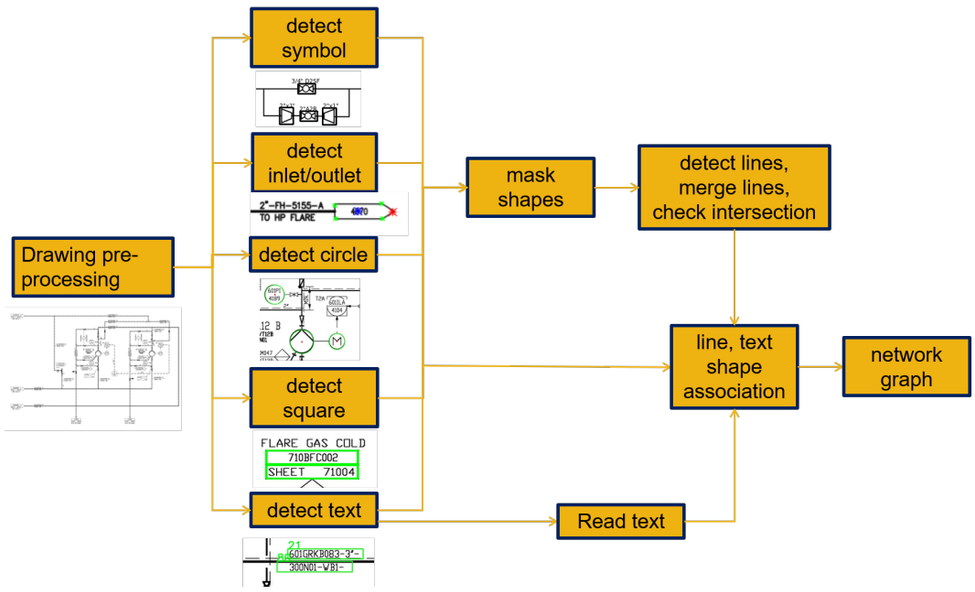

Figure 2: Overall proposed process of P&ID digitization.

4 Approach

4.1 Shape Detection

The purpose of shape detection is to identify all small components on P&D including valves, reducers, Squares, inlet and outlet. Different shape detection approach is adopted based on characteristics of shapes.

1. **Detection of Inlet, Outlet and Square**: The inlet or outlet marks the starting or ending point of the pipeline. There is a standard symbol representing the inlet or outlet as shown in Figure 3. It is a polygon having 5 points or 7 points with one point at the arrow tip and the rest of 4 or 6 points has at least one line of symmetry crossing arrow tip point. Those properties are used to identify the inlet and outlet. To implement this, we use Ramer-Douglas algorithm [10] to approximate all polygons from images. After detecting each polygon, we find out whether it is an inlet or outlet based on the property above. The detection of square-like table can be implemented with the same approach above.

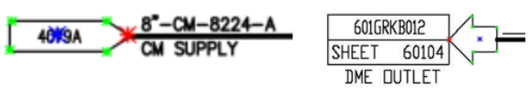

Figure 3: Example detection of inlet and outlet.

2. **Detection of Circle**: Circle detection is mainly used to identify motors or instrument alarms. Hough circle transform can be used which is very similar to Hough line transform. The initial output from Hough circle transform tends to provide a lot of false positives. An additional algorithm is developed to check the pixel color within a ring area around the circle edge.

3. **Detection of Other Symbol**: Other symbols such as valve and reducer are detected using a template matching technique. It is a technique in digital image processing to find small parts of an image that matches a template image. This relies on the extraction of image features such i.e., shapes, textures, colors to match in the target image. This algorithm is further extended to accommodate multiple scales of images. The proposed matching is based on the matching score exceeds a certain threshold value.

4.2 Text Recognition and Identification

The purpose of text detection is to identify text regions from the drawings and potentially associate lines and shapes to those identified texts. The state-of-art Connectionist Text Proposal Network (CTPN) [9] is used to localize text sequences. The architecture of this network is shown in Figure 4. It uses VGG16 as a base net to extract feature maps. A 3×3 spatial window is used to slide the feature maps of the last convolutional layer which allows it to share convolutional computation. The sequential windows in each are then recurrently connected by a bi-directional Long Short Term Memory (LSTM). This LSTM layer is connected to a fully-connected layer, followed by the output layer, which jointly predicts text/non-text scores, y-axis coordinates, and side-refinement offsets of k anchors.

The CTPN method is suitable for scene text detection with a wide range of scales and aspect ratios by using a vertical anchor mechanism. However, due to the same reason, it is not particularly robust for a vertical text orientation which is common in a P&ID drawing. Therefore, an image rotation step is adopted to identify vertical text and an additional algorithm is developed to merge text proposals from two oriented images and to avoid overlapping.

Once text regions are proposed, Tesseract 4.0 [8] is used to read text from proposed regions using the pre-trained model. It is identified that the CTPN text typically does not follow any dictionary which creates an additional challenges on text recognition. One way to improve this is to limit characters predicted based on engineering knowledge.

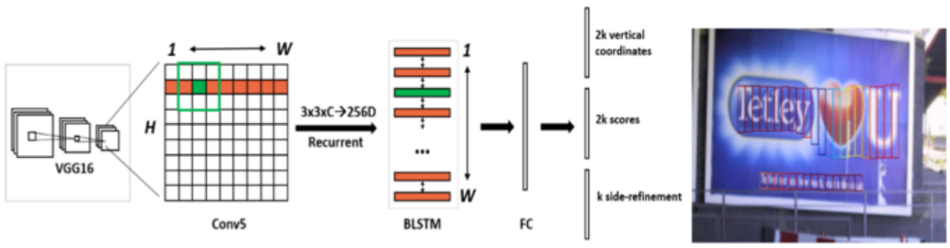

Figure 4: Architecture of the CTPN.

4.3 Pipeline Detection

All identified shapes and text are all removed first based on stored contour pixel coordinate location to reduce potential interference from lines in shape. We then use probabilistic Hough transform [3] on the skeleton version of the image which outputs a set of all lines including lines that do not correspond to pipelines. The outputs from the Hough line transform are small disconnected line segments which require to be merged. The line merge step is to check the coordinates from extreme points of each line and to see if any of the two line segments can be merged based on their orientations and distance. Several threshold criteria are setup to take account of the tolerance from gap or misalignment.

The last step is to check the line intersection type. As shown in Figure 5, there are two types of intersections: a valid intersection and an invalid intersection. The purpose is to identify all possible intersections and then exclude all invalid intersection. This is achieved by determining possible intersection points between any two lines by solving the system of linear equations. An invalid intersection is where an intersection where the solution of the two linear equations for the line has given us an intersection but there exists no such intersection as shown in Figure 5. A local square kernel is drawn with the center of the possible intersection points. The invalid intersection is discarded based on the following properties: 1) there exists at least one row with black color pixel occupying most of the area which indicates a line pass through the intersection; 2) there exists at least one row on each side of the line which has all-white color pixels. After excluding all invalid connections. the output is stored for later use to create a graph structure.

Figure 5: Different examples of pipeline intersection type. (only the leftmost type is not a valid intersection)

4.4 Network Graph Generation

The purpose of this step is to connect all components and information together and to generate a network-like graph to visualize the results. From the above steps, the location based on contour vertices coordinates for all identified texts, shapes, and lines as well as line connections are stored as ".txt" format. The text association is performed first based on minimum Euclidean distance between shape and text or center of line and text. Those identified shapes and lines become the graph nodal points. The associated texts become nodal labels. The association between shapes and lines are checked next based on minimum Euclidean distance again. All shape types are divided into the two categories: the first category only has a single line connected to the shape and the second category has two different lines connect the shapes. The association between lines and shapes becomes graph edges.

We use the networkX library which is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks to generate graphs based on graph node and edge information. After this, additional nodes contraction step is performed to merge connected lines together based on their connectivity.

5 Experiment Results

In this section, we present some results of this proposed end-to-end pipeline for P&ID information extraction. We use a combination of real-world P&ID and synthetic simple P&ID for the testing purpose. The results are shown in Table 1. Noting that the accuracy here only includes false negatives. A small amount of false positives are neglected which could be screened out by further tuning parameters. In addition, we only tested template matching from a limited synthetic dataset. Future work is needed to test template matching with a large dataset. It can be found that except circle detection, most of the shape detection algorithms achieves relatively high accuracy which indicates the robustness of the algorithm. An example of prediction result from circle detection is shown in Figure 6 which clearly missed two "incomplete" circles.

Figure 7 shows an example result from CTPN text detection. Green color bounding boxes are detected from the original image, while red color ones are detected from the rotated image. It can be seen that the application of CTPN did miss some of the vertically oriented text until image is oriented by 90 degrees. But the oriented image did generate additional false positives which could be potentially mitigated through Tesseract.

Table 1: Shape and text detection accuracy

Component	Sucessful Cases	Accuracy
Inlet/Outlet Detection	18/18	100%
Circle Detection	101/119	85%
Text Detection	98/102	97%
Template Matching	10/10	100%

Both line detection and component tag name association are mainly tested by using a smaller synthetic data set, as the current algorithm needs to be further developed to make it robust for a complicated P&ID graph. As illustrated by Figure 8, the bottom subfigure shows a generated graph based on top P&ID drawing. All identified shapes and lines become graph nodes, the line connection information becomes graph edges, and the recognized texts become labels of the nodes. Except Tesseract text recognition, we are able to achieve 100% accuracy from a synthetic dataset. Text recognition results from Tesseract did witnesses challenges of reaching a reasonable level of accuracy. This could be further improved to define the piping and equipment components' naming convention.

6 Conclusion and Future Work

In this paper, a novel P&ID digitization pipeline is proposed to extract information from pdf or scanned drawings. Several state-of-the-art techniques have been used to achieve this including deep neural network, image processing, etc. We are able to connect each component together and to generate a network graph based on several synthetic P&ID data. Overall, it achieves satisfactory results as a POC project. Further improvements could include 1) to make text identification and association more robust; 2) Use real-world P&ID drawings to test algorithm; 3) Construct deep neural network to train the component recognition.

All relevant python codes for this project can be downloaded from following Github repository: `https://github.com/jiandaoz1/AB-PID-recog`

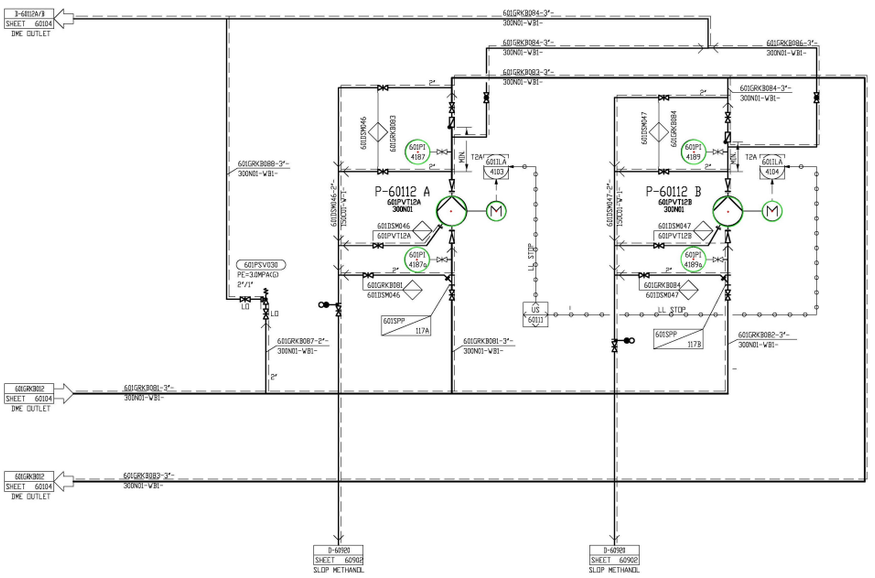

Figure 6: Example of circle detection results

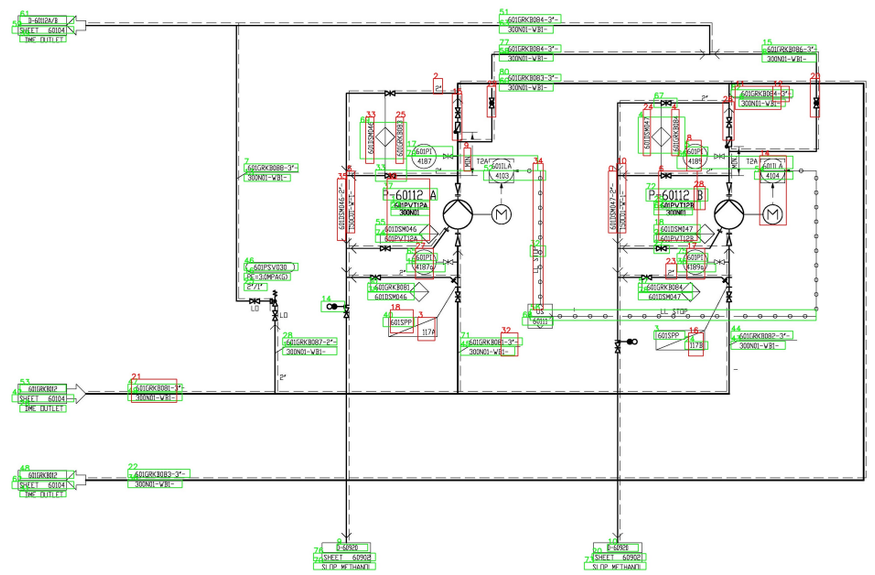

Figure 7: Example of ctpn text detection results

References

[1] Eyad Elyan, Carlos Moreno Garcia, and Chrisina Jayne. Symbols classification in engineering drawings. Institute of Electrical and Electronics Engineers, 2018.

[2] Luoting Fu and Levent Burak Kara. From engineering diagrams to engineering models: Visual recognition and applications. *Computer-Aided Design*, 43(3):278–292, 2011.

[3] C Galamhos, J Matas, and J Kittler. Progressive probabilistic Hough transform for line detection. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 1, pages 554–560 Vol. 1, jun 1999.

[4] Sung-O Kang, Eul-Bum Lee, and Hum-Kyung Baek. A Digitization and Conversion Tool for Imaged Drawings to Intelligent Piping and Instrumentation Diagrams (P&ID). *Energies*, 12(13):1–26, July 2019.

[5] Zhaoyang Lu. Detection of text regions from digital engineering drawings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):431–439, 1998.

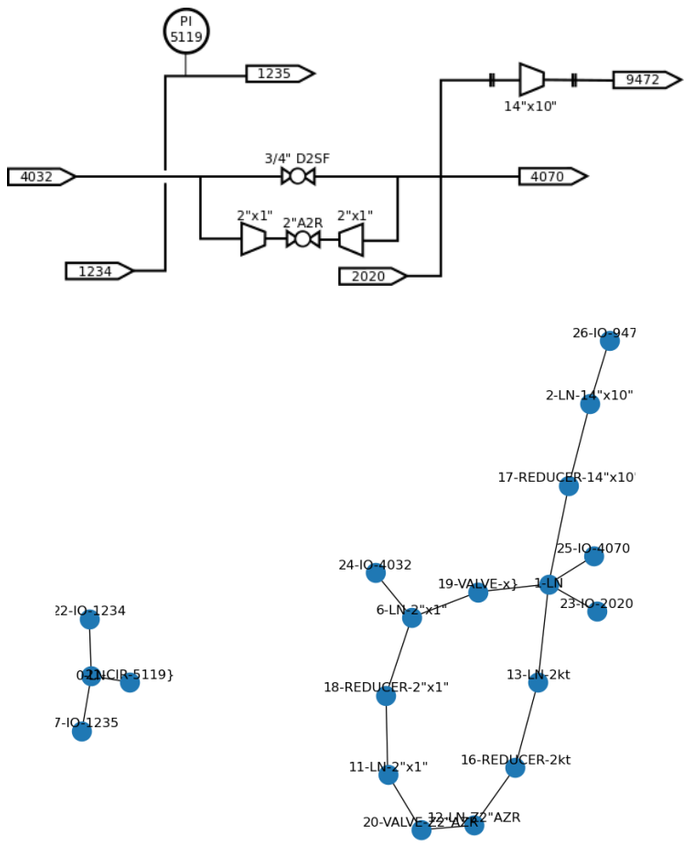

Figure 8: A example of P&ID (top) and generated network graph (bottom).

[6] Luo Yan and Liu Wenyin. Engineering drawings recognition using a case-based approach. In *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, pages 190–194 vol.1, aug 2003.

[7] Rohit Rahul, Shubham Paliwal, Monika Sharma, and Lovekesh Vig. Automatic Information Extraction from Piping and Instrumentation Diagrams, 2019.

[8] Ray Smith and Google Inc. An overview of the tesseract ocr engine. In *Proc. 9th IEEE Intl. Conf. on Document Analysis and Recognition (ICDAR*, pages 629–633, 2007.

[9] Zhi Tian, Weilin Huang, Tong He, Pan He, and Yu Qiao. Detecting Text in Natural Image with Connectionist Text Proposal Network, 2016.

[10] Wikipedia contributors. Ramer–douglas–peucker algorithm — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Ramer%E2%80%93Douglas%E2%80%93Peucker_algorithm&oldid=953709585`, 2020. [Online; accessed 22-July-2020].