Hand-drawn draft to code - towards an AI application to industrial automation

Inacio Fonseca[1]
inacio@isec.pt
Claudio Fonseca[1]
a21220051@isec.pt
Fernando Lopes[1,2]
flopes@isec.pt,flopes@co.it.pt

[1] Polytechnic Institute of Coimbra, Coimbra Institute of Engineering,
Coimbra, PT
[2] Instituto de Telecomunicações,
Coimbra, PT

Abstract

The field of industrial automation is undergoing a rapid transformation, driven by the integration of artificial intelligence (AI). Among the most innovative applications of AI in this field is its remarkable ability to convert hand-drawn images of application drafts into functional code. Drafts can be from new or old projects, paper or digitally acquired. This groundbreaking advancement serves as a bridge between design and execution, enabling more flexible and efficient programming. In this paper, we will delve into the potential and impact of this AI-driven process on industrial automation.

1 Introduction

The field of automation can be likened to languages that enable communication and control within industrial processes. A SCADA system (Supervisory Control and Data Acquisition) functions as a conductor, harmonising these languages to ensure efficient and coordinated operations, while also providing a visual interface for monitoring and analysing the results.

Different programming languages can be utilised in industrial automation, depending on the specific requirements and systems involved. For instance, PLC (Programmable Logic Controller) languages such as Ladder, ST (Structured Text), FBD (Function Block Diagram) and SFC (Sequential Function Chart), are commonly used. Robotics languages, on the other hand, are typically script-based, such as Rapid from ABB. Additionally, process control languages and motion control languages may also come into play. Humanoid industrial robots are also emerging, with the Apptronik Apollo1 serving as a notable example in this field [1].

The SCADA system serves as the interface for monitoring and controlling industrial processes, facilitating efficient and coordinated operations by utilising visual representation and industrial protocols to ensure seamless information flow.

Converting hand-drawn images of drafts into usable code has traditionally been a labour-intensive and error-prone human task. Engineers often face difficulties in translating intricate designs into executable instructions, resulting in inefficiencies and project delays. However, the convergence of AI and industrial automation offers a revolutionary solution. By harnessing advanced machine learning algorithms, AI systems can now comprehend hand-drawn images, extract relevant information, and generate precise and functional code. This groundbreaking advancement bridges the gap between design and execution, enabling more efficient and accurate programming in the field of industrial automation.

In the future, this research will explore the potential of leveraging AI LLMs (Large Language Models) in code generation, with a particular focus on mapping hand-drawn images and image-based drawings into an intermediate representation and then converting it into a web-based SCADA system or PLC system code.

This paper presents an initial investigation and associated findings related to image block detection using the YOLO (You Only Look Once) object detection algorithm [2].

2 Impacts on industrial automation and the conversion process

AI-driven code conversion from hand-drawn drafts boosts industrial automation with improved speed, accuracy, and collaboration. It speeds up development cycles, minimizing human errors and resulting in precise, consistent code. AI allows engineers to focus on high-level design and strategy, promoting collaboration and innovation in the process.

AI-driven hand-image-to-code conversion has numerous potential benefits, but some challenges must be overcome: (1) Image Context: AI needs to recognize the context behind hand-drawn images; image recognition technology must be improved. (2) Ambiguity Handling: Hand-drawn drafts may contain ambiguities which can be a challenge for AI to generate correct code; robust algorithms to manage these situations must be created. (3) Domain Adaptation: Each industry has its own terminology and needs; AI must be able to adapt to multiple domains.

AI can evaluate hand-drawn images of drafts using the following steps: image recognition, semantic understanding, code generation and code validation. Image recognition converts the visual content to a digital format, allowing AI to analyze it. Semantic understanding leverages machine learning and natural language processing to interpret the underlying purpose of the design. Code generation creates code segments that match the functionality. Validation ensures the accuracy and proper functioning of AI-generated code through simulation and testing.

3 Identification of hand-drawn items with YOLO

Figure 1 illustrates hand-drawn items that must be accurately detected and classified for SCADA visual design. AI-based YOLO was used for labeled item detection. Similar studies on diagram detection have been documented in [3].

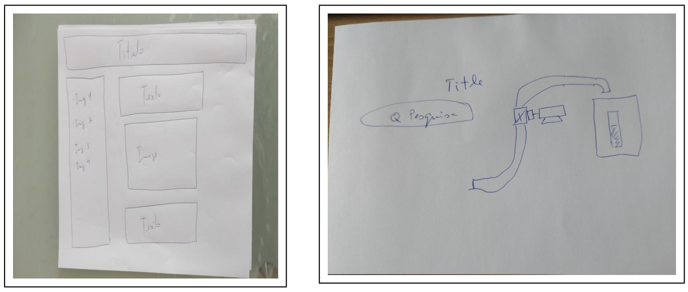

Figure 1: Two examples of hand-drawn images for the training dataset.

YOLO is a state-of-the-art, CNN-based, real-time object detection system. Joseph Redmon and co-authors developed YOLOv1 [2], YOLOv2 [4] and YOLOv3 [5]. Alexey Bochkovskiy and co-authors developed YOLOv4 [6] and YOLOv7 [7]. YOLOv5 [8] and YOLOv8 [9] were developed by Ultralytics and YOLOv6 [10, 11] was developed by Neituan. The official releases are YOLOv1,v2,v3,v4 and v7 while YOLOv5,v6 and v8 are unofficial (released by private companies).

Table 1: Image dataset composition, with labels frequency and mAP50-95 results in the training phase. V(*)=Validation

	Train Set	Test Set	V(*) Set	Total	mAP50-95 (YOLO)		
					v8s	v8n	v7
N.º of Images	105	5	10	120			
Button	51	7	4	62	0.676	0.590	0.615
Footer	33	1	3	37	0.808	0.667	0.813
Header	88	3	8	99	0.751	0.759	0.628
Image	33	0	4	37	0.933	0.904	0.933
Motor	12	0	1	13	0.796	0.697	0.597
Pipe	18	0	2	20	0.895	0.746	0.945
Text	318	16	37	371	0.760	0.731	0.796
Total	559	28	59	646	0.803	0.728	0.761

The work in this paper used YOLOv7, YOLOv8n (nano) and YOLOv8s (small) versions. Epochs were set to 500 and patience to 50 (except for YOLOv8s). No other default parameters were changed.

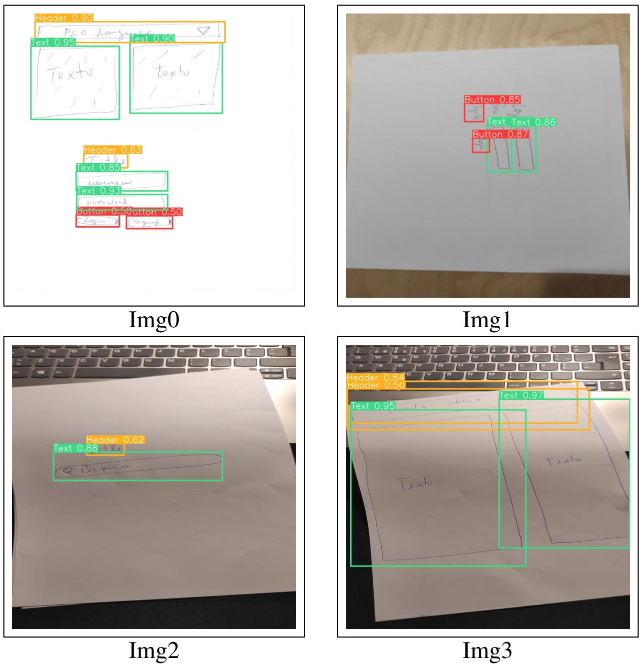

Figure 2: Detection in the test dataset with YOLOv8n.

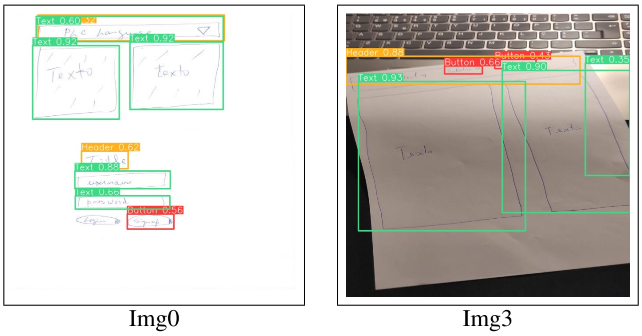

Figure 3: Results with YOLOv8s (images with differences to v8n).

The images in Figure 1 are two representative examples in the training set. All images were annotated with 7 labels: "Button", "Footer", "Header", "Image", "Motor", "Pipe" and "Text". Table 1 details the dataset and COCO mAP50-95 metric results. All three YOLO versions achieved a VOC mAP50 score of 0.94 or higher for all labels.

Example visual results on the test dataset are presented in Figure 2 for YOLOv8n, in Figure 3 for YOLOv8s and in Figure 4 for YOLOv7. Detection was successful due to the consistent annotation across all images taken with two mobile phones and a scanner (despite varying resolutions).

Times for the three models in the detection phase were as follows. YOLOv8n: 0.2 ms preprocessing, 13.6 ms inference, 0 ms loss, 106.2 ms postprocessing per image and 193 epochs (patience=50). YOLOv8s: 0.2 ms preprocessing, 6.9 ms inference, 0.0 ms loss and 111.2 ms postprocessing per image. It took 298 epochs with a patience 80 or 122 epochs with a patience of 50, but with a reduction in performance in the test set. Differences for YOLOv8n can be observed in images Img0 and Img3 in Figure 3. YOLOv7: 22.1 ms inference, 1.4 ms NMS per image and 500 epochs. The machine used to run the algorithms was equipped with a NVIDIA TESLA T4 with 15360 MiB of RAM using drivers NVIDIA-SMI 525.105.17 and CUDA Version 12.0.

4 Conclusions

The integration of AI to convert hand-drawn drafts into code will be a transformative process in industrial automation. This concept can increase efficiency, accuracy, and productivity, fostering industrial advancements. AI's rapid growth means that hand-drawn designs can quickly become executable code, creating a new level of efficiency and collaboration. By blending human creativity and machine intelligence, industries can leap into an automated innovative future, dramatically changing the automation landscape.

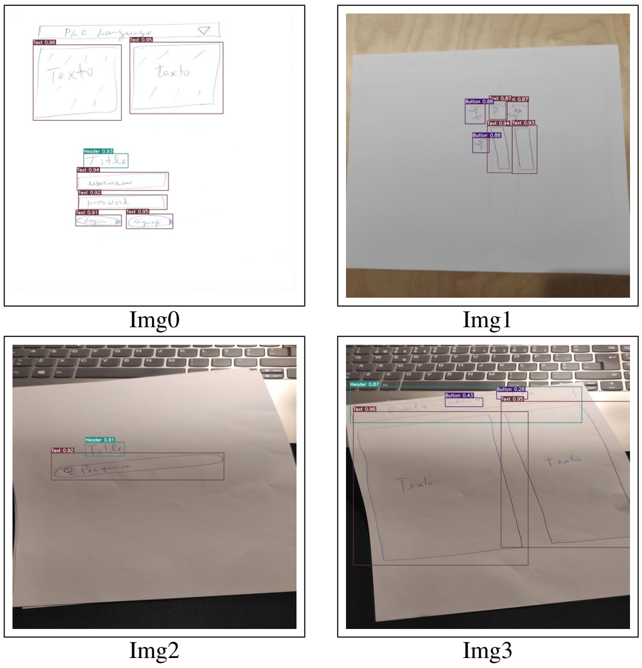

Figure 4: Detection in the test dataset with YOLOv7.

This paper presented a preliminary investigation and results for a proof of concept on the ability to detect hand-drawn images and convert them into symbolic representations. Such representations will be used in further investigations for the purpose of automation code generation.

Both YOLOv8 and YOLOv7 performances in object detection were evaluated against a test dataset and were found to be accurate when using a training set with well defined labels. However, careful considerations on label annotations and on the image dataset composition are crucial, in order to ensure a successful implementation. The results of this initial study suggest a high potential for fast code generation in Web SCADA systems and PLC programming, an objective that we are pursuing through our ongoing work. Both algorithms can detect user-drawn inputs in real-time, such as those acquired from a smartphone video stream.

References

[1] Apptronik. Why nasa is helping apptronik build a humanoid robot. *CNet*, February 2023. `https://www.cnet.com/videos/why-nasa-is-helping-apptronik-build-a-humanoid-robot`.

[2] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*, 2016. `https://arxiv.org/abs/1506.02640`

[3] Somaieh Amraee, Maryam Chinipardaz, Mohammadali Charoosaei, and Mohammad Amin Mirzaei. Handwritten logic circuits analysis using the yolo network and a new boundary tracking algorithm. *IEEE Access*, 10: 76095–76104, 2022. doi: 10.1109/ACCESS.2022.3192467.

[4] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016. `https://arxiv.org/abs/1612.08242`

[5] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. `https://arxiv.org/abs/1804.02767`

[6] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. `https://arxiv.org/abs/2004.10934`

[7] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*, 2022. `https://arxiv.org/abs/2207.02696`

[8] Ultralytics Inc. YOLOv5: Ultralytics open-source research into future vision ai methods. 2021. `https://ultralytics.com/`

[9] Ultralytics Inc. Introducing yolov8: Faster, simpler, more accurate. 2023. `https://ultralytics.com/`

[10] Chuyi Li, Lulu Li, Yifei Geng, Hongliang Jiang, Meng Cheng, Bo Zhang, Zaidan Ke, Xiaoming Xu, and Xiangxiang Chu. Yolov6 v3.0: A full-scale reloading. *arXiv preprint arXiv:2301.05586*, 2023. `https://arxiv.org/abs/2301.05586`

[11] Chuyi Li et All. Yolov6: A single-stage object detection framework for industrial applications. *arXiv preprint arXiv:2209.02976*, 2022. `https://arxiv.org/abs/2209.02976`,