applied system innovation

MDPI

Article

Deep Learning Method to Detect Missing Welds for Joist Assembly Line

Hamed Raoofi [1], Asa Sabahnia [1], Daniel Barbeau [2] and Ali Motamedi [1,*]

[1] Construction Engineering Department, École de Technologie Supérieure, Montréal, QC H3C 1K3, Canada; hamed.raoofi@groupecanam.com (H.R.); asa.sabahnia.1@ens.etsmtl.ca (A.S.)
[2] Groupe CANAM, Montréal, QC G0M 1T0, Canada; daniel.barbeau@groupecanam.com
* Correspondence: ali.motamedi@etsmtl.ca

Abstract: Traditional methods of supervision in the construction industry are time-consuming and costly, requiring significant investments in skilled labor. However, with advancements in artificial intelligence, computer vision, and deep learning, these methods can now be automated, resulting in time and cost savings, as well as improvements in product quality. This research focuses on the application of computer vision approaches to monitor the quality of welding in prefabricated steel elements. A high-performance network was designed, consisting of a video capturing station, a customized classifier based on a YOLOv4 detector and an IoU tracker, and a user interface software for any interaction with quality control workers. The network demonstrated over 98% accuracy in identifying steel connection types and detecting missed welds on the assembly line in real-time. Extensive validation was conducted using a large dataset from a real production environment. The proposed framework aims to reduce rework, minimize hazards, and enhance product quality. This research contributes to the automation of quality control processes in the construction industry.

Keywords: computer vision; machine learning; weld inspection; quality control; object detection; DNN; construction quality management; deep learning

Citation: Raoofi, H.; Sabahnia, A.; Barbeau, D.; Motamedi, A. Deep Learning Method to Detect Missing Welds for Joist Assembly Line. *Appl. Syst. Innov.* **2024**, *7*, 16. https://doi.org/10.3390/asi7010016

Academic Editors: Patrícia Ramos and Jose Manuel Oliveira

Received: 11 December 2023
Revised: 19 January 2024
Accepted: 24 January 2024
Published: 13 February 2024

1. Introduction

The fast-paced nature of industrial mass-production demands that a substantial quantity of products be manufactured quickly while meeting the corresponding required quality standards. To ensure products meet the targeted level of quality, a rigorous quality control mechanism needs to be in place. Yet, if this process is inefficient, quality tasks may affect production, for instance, by slowing it down or quality may suffer, for instance, errors might go on undetected in the final product.

In the construction industry, steel joists are one of the most widely used structural components. Steel joists are assembled using a Gas Metal Arc Welding (GMAW) process. Because of the variety of steel joist types, the process uses manual welding, which is somewhat error-prone and can lead to the omission of welds. If such defects go undetected by the manual and visual quality control process, the performance and stability of the structural element will be seriously affected, leading to possible hazardous safety issues.

Traditional methods of quality control based on visual inspection have proven to be inefficient in terms of required time, accuracy, and cost. However, machine learning (ML) techniques can potentially provide an efficient monitoring system in a fast, accurate, and cost-effective way [1]. ML can be used to detect welding defects in various stages of the welding process.

While most research in this domain focuses on identifying welding defects, our study pivots to the critical task of detecting the absence of welded parts. This focus is vital for ensuring structural safety, as missing welds pose significant risks to product integrity. Furthermore, the study focuses on missing welds as detecting them involves a complex

process that typically relies on extensive manual labor. Automating such detection can enhance the efficiency of quality control process.

In this study, we present a novel deep learning-based framework for real-time detection of missing welds in steel joists during the shop fabrication process. Our proposed framework integrates a comprehensive hardware setup, including a data-capturing station that streams real-time assembly line videos. This hardware setup is integrated with advanced computer vision algorithms and machine learning techniques that are utilized to accurately detect, track and classify the various types of connections and nodes and identify missing welds on prefabricated steel joists. Additionally, a user-interface alerts workers in real-time when a defect is detected, enabling prompt corrective actions.

Research Objectives and Contributions

This research explores state-of-the-art methods and develops a deep-learning-based framework, focusing on achieving high accuracy and reduced processing time in monitoring welding quality during the shop fabrication process. The project focuses on the detection of missing welds on prefabricated metal joists in a factory setting. The objectives are to: (1) Investigate a method for automatic real-time detection of missed welds on the assembly line, using video streams from the cameras in the data-capturing station and machine learning algorithms, (2) Investigate and propose an innovative solution to tackle challenges such as tracking, occlusion, and processing time, (3) Train and validate the network on datasets provided by the partner company and test the developed network in the real assembly line environment, and (4) Fine-tune the network parameters to further increase performance and compare the proposed network results with existing procedures.

In our study, we have developed a deep learning-based framework to address the challenge of detecting missing welds in steel joists during the fabrication process. This framework incorporates a comprehensive hardware setup including a data-capturing station for streaming assembly line videos in real-time. Our approach integrates advanced computer vision techniques and machine learning algorithms to accurately detect, track, and classify various types of connections and nodes in steel joists. This system also includes a user interface to alert workers in real-time when a defect is detected, thus enabling prompt corrective actions.

This research presents a significant advancement in the field by automating the process of detecting missing welds, which traditionally relies on manual inspection. Our method aims to reduce rework, minimize hazards, and enhance product quality in the construction industry. The deployment of our framework is expected to yield high efficiency and accuracy in quality control processes for steel joist fabrication.

Our work distinguishes itself by providing a holistic approach to welding quality control. Our framework seamlessly integrates hardware, software, and machine learning techniques for real-time defect detection in a factory setting. While recent advancements in welding quality control have shown promising results, there remain significant challenges and gaps in current methodologies. Our study aims to address these gaps by introducing an innovative approach.

Furthermore, our framework is designed to achieve high accuracy and efficiency in weld defect detection, representing a potential advancement in automated quality control processes. Detailed comparisons and evaluations of our method in the context of existing research will be presented in the results and discussion sections.

This paper outlines an innovative approach to enhancing welding quality control in steel joist prefabrication through the application of advanced computer vision and machine learning techniques. The ensuing sections will delve into our methodology and the specifics of our framework, detail its integration of hardware and software for real-time defect detection and discuss its practical effectiveness in an industrial setting.

2. Previous Works

To report on the state-of-the-art computer vision research on welding quality control, the research project was divided into two parts, based on when the quality control occurred in the welding process: In-Process welding quality control, and post-process welding quality control.

2.1. In-Process Welding Quality Control

Using Machine learning techniques, weld quality can be monitored during the welding process. Various types of data can be gathered from the welding pool, such as the welding current, the arc spectrum, and Infrared images. Subsequently, this data can be used to create a dataset to train an ML-based model and afterward, the trained networks can identify possible defects that occurred in the welding process.

W. Jiao et al., 2020 [2] used a convolutional neural network (CNN)-based model to predict the penetration of the GTAW process using the top-side image of the weld pool. Additionally, a pre-trained model based on the residual neural network (ResNet) was proposed to improve the predictions' accuracy. By using transfer learning, they reduced the training time. Zhao et al., 2019 [3] developed a framework based on a cooperative awareness of the arc spectrum, vision data, and electrical parameters to control the quality of the welding process in wire-arc additive manufacturing. They found that the self-emitted radiation coming from the weld pool mainly consists of near-infrared light. Thus, to capture a better-quality image, an 850 nm high-pass filter was utilized to block high-frequency arc interference. Different classifiers were used depending on the welding current and wire material. For the welding process features, they monitored welding speed, shielding gas flow rate, and weld defects by analyzing multi-source data from the weld pool. Finally, a K-Nearest Neighbor (KNN) classification algorithm was used to classify the weld features extracted from vision and spectral data. Tang et al., 2020 [4] proposed a framework to detect defects in the fiber laser welding of stainless steel. By capturing the image of the keyhole, the features based on the shape and edges of the keyhole were extracted using grayscale projection distribution and the Poisson matting technique. Then, a Hidden Markov Model (HMM) was applied to correlate the geometry features of the keyhole and welding defects. To obtain desirable weld characteristics, the information provided by the vision-based model could be further utilized to adjust the significant welding variables. Xiong & Zou, 2019 [5] used a fuzzy logic-based controller to actively monitor and enhance the penetration quality of MIG welding in a thin aluminum sheet by manipulating the welding current. In a similar study conducted by Peng et al., 2019 [6], a proportional-integral controller was proposed to provide feedback control over the penetration quality of the GTAW process by adjusting the welding current.

2.2. Post-Process Welding Quality Control

The quality of the weld beads can be verified after the welding process is finished. In this approach, information about the surface of weld beads is extracted by utilizing devices such as profile sensors and cameras. Using machine-learning techniques, a model is designed and trained to detect irregularities and defects on the weld surface by creating a labeled dataset from the acquired raw data.

Soares et al., 2019 [7] presented a passive vision-based system to detect irregularities in the weld bead texture. The Principal Component Analysis (PCA) technique was deployed to acquire the main characteristics of each class while reducing the dimension of data. Then, a Support Vector Machine (SVM) model was trained based on the class characteristics to assess the weld surface quality. Han et al., 2020 [8] utilized a structured light sensor to measure the geometry of the weld bead. The laser stripe was projected onto the weld bead surface, and a camera captured the laser stripe through a narrow-band filter to decrease the environmental noises. Through image preprocessing and baseline extraction, a laser strip profile was extracted from the raw image data. Furthermore, the weld profile was classified as the filling weld or the capping weld using a threshold-based method. By extracting

features points from the stripe profile, the weld bead size could be measured. Additionally, the bead shape data was compared with acceptable thresholds to detect geometrical defects. Chu & Wang, 2017 [9] developed a passive machine vision-based framework to detect the undercut on shell-tube welding. By applying preprocessing algorithms, the quality of the image was enhanced, and the edges in the picture were extracted. Median [10] filter and Otsu segmentation [11] were utilized to segment the metal tube, and morphological operations were employed to improve the segmentation integrity. Finally, the undercut was detected by counting white pixels in each angle inside the tube. Chu & Wang, 2016 [12] presented an automated passive quality control system that used structured light to measure bead size and detect corresponding defects such as undercut and plate displacement. The laser centerline was determined in the image by applying preprocessing techniques such as Median filter, thresholding, and morphological operations. Then, feature points were extracted by fitting lines to the laser stripe reflecting from parent metal using the RANSAC technique. Afterward, a sliding window was applied to determine the extremums of the laser profile. Once feature points were calculated, the quality of welding could be controlled by comparing them with thresholds. Sun et al., 2019 [13] proposed a vision-based defect-detection system to inspect the welding quality of thin-walled metal canisters. They developed a modified Gaussian mixture model to extract defect feature areas from gray images. An area thresholding technique was applied to filter out false detections, which mainly consisted of small areas. Finally, using the feature image as the input, three main types of defects were classified by utilizing brightness thresholding and curve detection algorithms. Hartl et al., 2019 [14] developed a deep-learning framework to monitor the quality of friction stir welds. They utilized the one-stage object detector YOLOv2 [15] to detect and localize the weld seam and compared several convolutional neural networks such as DenseNet-121 [16] to classify the possible defects. They created 112 stir-weld samples that were used in the dataset to train and test the framework. The proposed framework input were RGB and topology images. Cruz et al., 2020 [17] designed a vision-based system that used the convolutional neural network to detect the misalignment of the metal parts before welding. In addition, the developed system could identify geometrical nonconformities of weld beads after the welding process was completed using laser triangulation and image processing techniques. Using image processing and deep learning techniques, Muniategui et al., 2019 [18] proposed a framework to detect defects such as Lack of Fusion (LoF) in mass-produced welded cylinder parts for flame sensors. They developed 1D-pDFT and Gabor filters that help highlight features related to the defect. Finally, a deep learning model utilized the output of the mentioned filters to detect defects. Haffner et al., 2016 [19] developed a weld inspection system that relies on image processing techniques to extract features such as weld edges. After feature extraction, the welds were evaluated by neural networks. They ran the developed system on a single-board computer (SBC) and employed a cloud-based solution to store the system output. Exploiting the laser triangulation technique, Spruck et al., 2020 [20] proposed a deep-learning-based inspection system to monitor the quality of weld seams. The laser scanner was mounted to a robot providing profile data of the weld seam surface. After preprocessing, such as gamma correction and normalization, a network based on Inception-v3 architecture [21] is responsible for the classification task on the weld quality. Xue et al., 2019 [22] used structured light to capture the weld surface profile. The 3d model of the weld seam was reconstructed by applying image processing filters, extracting the center of the laser stripe, and curve-fitting. After 3d model reconstruction, the weld geometry was measured and compared with the manual measurement. Dong et al., 2021 [23] developed an unsupervised approach to inspect weld quality in the aerospace industry using unannotated real and synthetic X-ray images as input. The unsupervised process is based on the U-Net [24] segmentation encoder-decoder and a k-means clustering technique. Finally, the random-forests (RF) classier was trained on the features extracted by the encoder.

In all the previously mentioned research projects, the goal was to monitor the existing or in-process welding quality. However, the present research does not focus on the quality

of existing welds. The purpose of this research is to detect missing welds in specific areas of prefabricated joists (i.e., nodes) that require precise assessment. By leveraging detailed engineering drawings and the expertise of the industrial partner's quality control team, we aim to specifically determine if any segment of a weld is absent. This approach is essential for accurately identifying incomplete welding in these critical areas, thus ensuring the reliability and safety of the construction. Additionally, none of the available open datasets was suitable for the target elements of this study, hence, a targeted dataset needs to be created.

3. The Proposed Method

The proposed solution aims to automatically monitor the quality of the welding process on steel joists moving at a slow speed and to identify the missing welds and report them in real time to the quality control agents. Our proposed method employs two industrial IP cameras accompanied by LED ring lights that are placed at the end of the assembly line to provide a live video feed of the finished steel joists (as shown in Figure 1) as they are passing. The video data is then retrieved by the developed framework for processing. The final results are streamed on a screen and serve to notify the quality control agents when any welds have been missed in the final product.

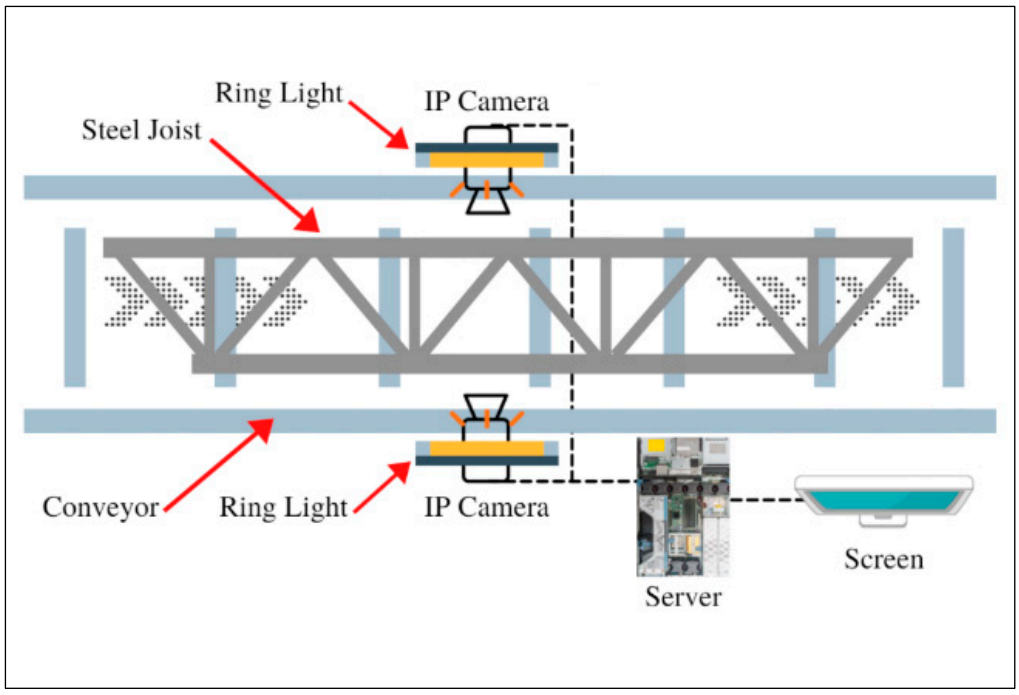


Figure 1. Schematic diagram (Top view).

3.1. Framework Architecture

As illustrated in Figure 2, the framework consists of three core modules: (1) detector: identifies welded nodes and provides details about their class and location based on the YOLOv4 technology [25], which is an advanced object detection algorithm that uses a deep convolutional neural network to detect and localize objects in real-time with high accuracy and speed; (2) tracker: utilizes information from the detector module to keep track of the welded nodes and their identity in the camera field of view (FoV) using an Intersection over Union (IoU)-based technique [26] which calculates the ratio of the intersection area between bounding boxes of objects in the current frame and the previous frame; and (3) classifier: receives the detection scores and the number of frames and identifies the final classification by ruling out the weak detections and giving an average score corresponding to the total number of remaining frames. Additionally, the framework utilizes four asynchronous

input/output modules responsible for reading, writing, and streamlining the data. An SQL database [27], which is a structured collection of data that can be easily accessed, manipulated, and queried using the SQL programming language, is used to store the framework predictions, while Ignition® [28], a web-based industrial automation software, is employed to develop the User Interface (UI).

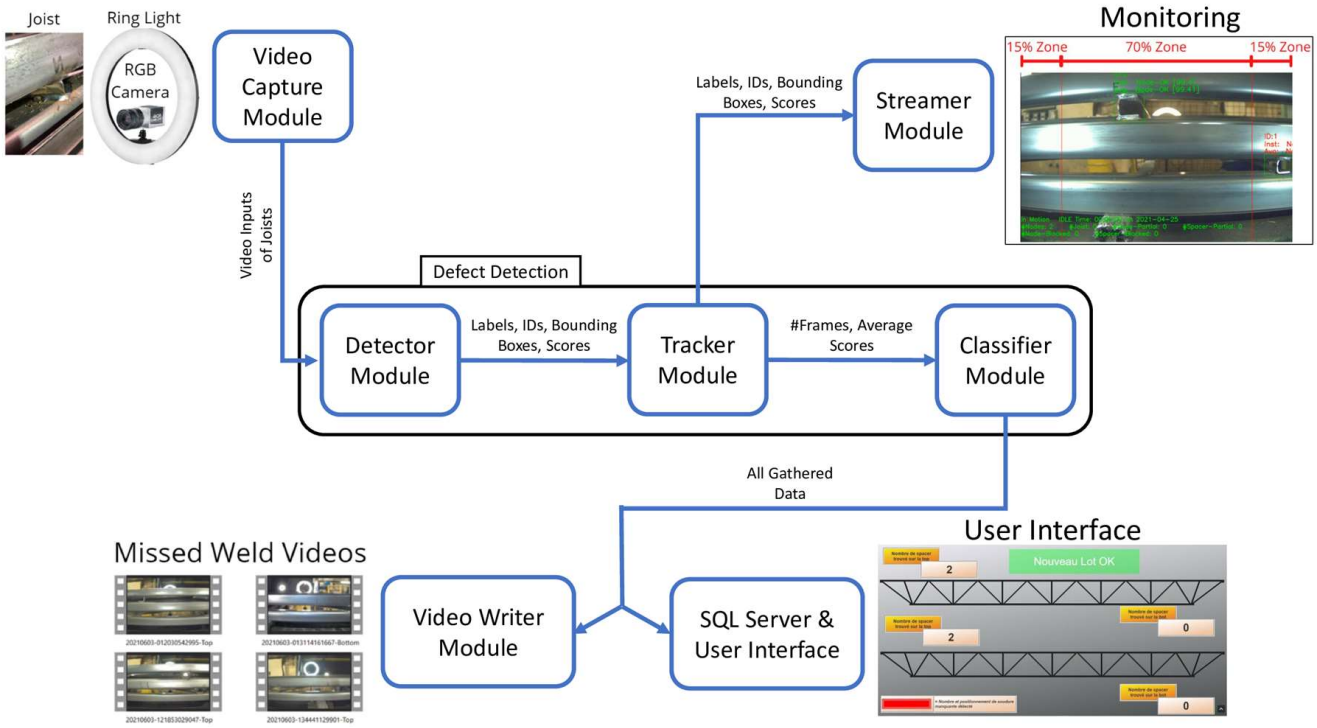


Figure 2. Framework architecture.

3.1.1. Video Capture Module

Pypylon 1.7.2 (2021) [29] the official python wrapper for the Basler pylon Camera Software Suite, is used to communicate with the IP cameras. Using the Python multi-threading feature, the video stream is grabbed every 1/FPS second from the cameras by an async module. On an ongoing basis, the agent returns the captured frame along with the camera connection status and stores it in a shared memory buffer. The content of the buffer is the input for the detector module.

3.1.2. Detector Module

Using deep-learning techniques, the detector module detects intended objects and provides information about their classes and locations accordingly. To achieve this capability, the detector module needs to be trained on an annotated dataset consisting of adequate images of the desired classes. As high speed and accuracy are necessary in such real-time applications, the YOLOv4 (You Only Look Once, version 4) algorithm [25] was selected as the frameworks detector module. YOLOv4 is a significant update in the YOLO series. It employs a convolutional neural network (CNN) optimized for speed and accuracy. YOLOv4 introduces several improvements over its predecessors, including better utilization of GPU capabilities for enhanced performance. It uses features like Weighted-Residual Connections, Cross-Stage Partial connections, and a more efficient backbone, which contributes to its improved object detection capabilities. These technical advancements make YOLOv4 particularly suited for tasks requiring fast and precise image analysis, such as industrial quality control [25]. YOLOv4 is a one-stage object detection algorithm that, unlike the two-stage algorithms (e.g., R-CNN [30] or Fast R-CNN [31]), is suitable for real-time applications [32]. The selection of YOLOv4 in this study, despite other potentially more complex models, is driven by its real-time detection capabilities, aligning with our study's goal to demonstrate

the feasibility of using computer vision for detecting missing welds, as a practical proof of concept. The choice was also influenced by YOLOv4's proven effectiveness in real-time object detection, its compatibility with our industrial partner's IT environment, and the potential for future enhancements by the partner's teams. In our proposed framework, the corresponding Python wrapper for C-based Darknet, which is a framework for developing deep neural networks in the C programming language, is used to Integrate the Darknet implementation into our Python-based Framework.

Novel Enhancements for the Detector Module

If for any reason the production line stops for a period of time, the system will continue creating multiple similar frames that do not add more helpful information, the static status mentioned above can induce errors in the classification based on the average score. To tackle this problem, a motion detection method is employed to enable the framework to pause the detection and tracking whenever the line stops. Figure 3 shows the flow chart of this motion detection technique.

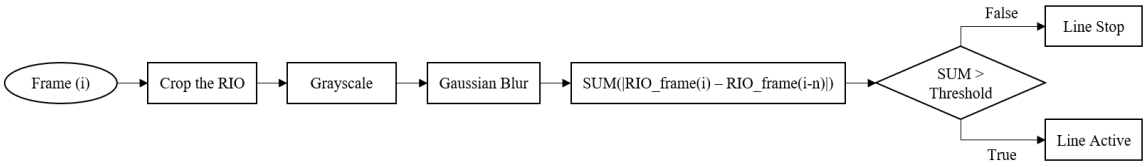


Figure 3. Motion detection technique.

Within the frame, a region of interest is selected that has minimal environmental noise. In grayscale mode, each frame is smoothed by a Gaussian blur filter to further reduce the noise and differentiate it from the previous nth frame. Then, the sum of arrays on the absolute-difference image is calculated and compared with a scalar threshold. A 12-frame gap and a threshold value of 0.006 were obtained for the assembly line environment from trial and error.

3.1.3. Tracker Module

Using label and bounding box information provided by the detector, an IoU-based [26] tracker module is deployed to track the nodes on the camera field of view. By employing a tracker module, the object" identities can be preserved while they are in the FoV. To track objects in the video, each frame detection is compared with its corresponding previous frame detection, and a metric called IoU [26] is calculated for each object by dividing the overlapping area by the combined area for two consecutive bounding boxes. If the IoU value is greater than a chosen threshold, the object in the current frame is considered the same as in the previous frame.

Aside from providing object-counting capability, tracking the objects offers further possibilities to develop postprocessing techniques to improve detection results and reduce errors. Within the Tracker Module, to prevent making errors by relying on a single detection score, the network scores for each node are averaged across the sequence of frames until the node exits the field of view of the camera.

Novel Enhancements for the Tracker Module

To improve the accuracy of the framework, objects near the two sides of the FoV are truncated as they may not be fully visible while entering or exiting the FoV. In the mentioned zones, the detector module often has difficulties in successfully detecting and classifying nodes. To resolve the above problem, three detection zones are defined in the FoV (Figure 4).

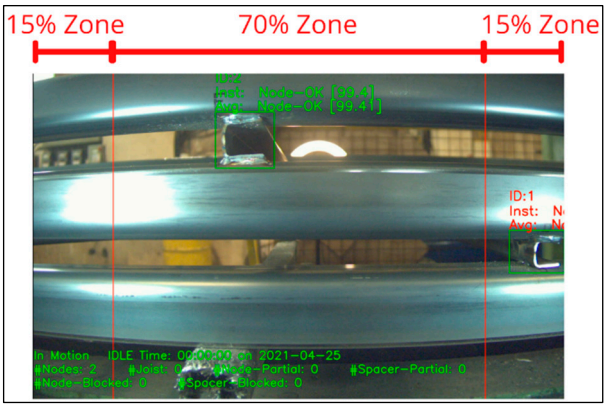


Figure 4. Three detection zones.

Thus, 15% of the frame width on either end is the area in which the framework detects and tracks nodes, but its corresponding prediction scores are not considered in the average score. The classifier module only receives the prediction score if the node occupies between 15% and 85% of the frame width; otherwise, the tracker will not pass the score to the classifier. To determine where the node position is within the three mentioned zones, the center point of the bounding box around the node is utilized.

It is possible that an object detected in the preceding frames is not detected in the current one. This scenario happens when the detector module cannot identify the present object in the current frame or simply because the object left the camera FoV. To resolve the uncertainty of whether the object reappears in the FoV or not, a transitional phase is designed to preserve object data for a certain number of object absences. Considering that the object may reappear in the subsequent frames in the mentioned phase, OpenCV Median Flow tracker is applied to increase tracking performance. Because the joists move slowly, the Median Flow tracker can effectively update the bounding box information during the frames in which the detector has difficulty finding the object. If an object is absent for more than a specific number of frames, the object is considered to be outside the FoV, and its corresponding detection data is removed from the tracker memory.

3.1.4. Streamer Module

If a web user demands to watch the result on the live video feed, using the Flask web framework (Pallets Projects, 2021) [33] an async program will infuse the object detection and tracking results with the raw frames and stream the output over the local network in real-time.

3.1.5. Classifier Module

Once a node is detected in the camera FoV, the tracker module sends the average scores and corresponding total frame number to the classifier module. As a postprocessing unit, the classifier module rules out weak detections by thresholding the frequency of object appearance in the FoV. If a total frame number of object detection is less than a set threshold, these frames will be filtered out and not considered for further processing. In considering the production environment of the case study, the speed of the conveyor belts and the FoV of the camera, the threshold was set to 30 frames (2 s) as it was determined through trial and error that most 'miss detections' are achieved below this threshold.

Since finding missed welds is the main goal of the framework, class weights were applied to the average score vector to increase the detection probability of the missed-weld class in the classification phase. To implement this technique, the class score related to the Missed-Weld class was multiplied by a factor of 2.5, that was obtained by trial and error (Classes are explained in Section 4.1), and the classifier was carried out according to the network weighted average score.

3.1.6. SQL Inserter Module

Final classification results, stored in an SQL database by an async program, can be retrieved to view the real-time quality control reports on a graphic user interface in addition to the human checker.

3.1.7. Video Writer Module

To visually evaluate the defects detected by the framework, automatically expand the dataset, and decrease the imbalance rate, the captured missed welds are saved in video format. If any missed-weld class is detected, a batch of the previous 300 frames is recorded locally in a different thread, to avoid increasing unnecessary corresponding processing time on the main thread. The frames collected in this way are highly useful to enrich the dataset and decrease the imbalance rate.

3.1.8. User Interface

The user interface retrieves prediction data and the computed relative position of the missed welds from the database and displays them to the users. It thus simultaneously streams the final results on the screen and informs the quality control agents of any missed welds on the production line.

3.2. Modules at a Glance

Table 1 summarizes the characteristics of the developed framework modules.

Table 1. Framework modules.

Module	Role	Type
Video Capture	- Grab image data from cameras	- Async capturing
Detector	- Detect objects in a frame	- YOLOv4
Tracker	- Track detected objects in the video - Assign ID to each object - Average Detector score for each object	- Hybrid (based on IoU & Median Flow)
Streamer	- Stream the predictions in video format	- Async streaming
Classifier	- Filter weak detections - Classify detected objects based on average score	- Appearance thresholding - Weighted voting
Video Writer	- Save video of an object that belongs to a specific class	- Async video writing
SQL Inserter and Database	- Insert classification output into an SQL database - Store the prediction data	- Async inserting
UI Software	- Retrieve prediction data from the database and show them to the user	- Ignition® - Web-based

3.3. Framework Validation Methods

A two-phase validation process is proposed in this research. In the first phase, the performance of the Detector Module is evaluated using two separate validation methods: (1) stratified k-fold cross-validation and (2) train-validate-test split (dividing the images randomly into three sets: training, validation, and test dataset). The metrics for this validation were average precision (AP) over different IoU thresholds (e.g., AP_{50}, average precision over IoU threshold value equal to 50%) and mean average precision (mAP) [34]. Additionally, as the final classification is based on the average detector score on the sequence of frames, our framework as a whole should be validated on the unseen objects as well. Metrics such as precision, recall, and confusion matrix are calculated for each class to assess the overall framework performance.

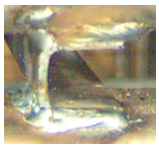
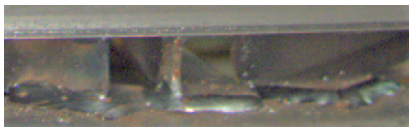
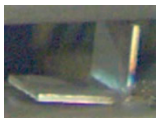
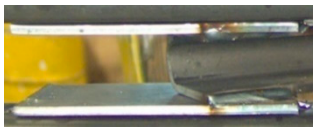
For the second phase, in the case study, the framework was tested in real factory settings for 17 working days in the production phase, where it monitored approximately

3000 joists and a total of 111,492 nodes. The results of missed welds detections are compared with the corresponding quality control reports in Section 6.2.4.

4. Case Study Dataset Preparation

The case study has been performed at one of the production lines of Group Canam Inc., one of the major steel construction companies specializes in designing and fabricating metal components for the North American construction industry. To train the detector module, a dataset was created of 15,356 annotated images containing six object classes (Table 2).

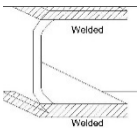
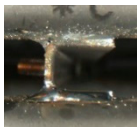
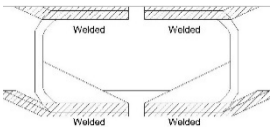
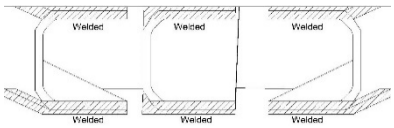
Table 2. Six classes of objects.

Class Name	Image	Note
Node-OK		All the required parts of the node are welded adequately.
Missed-Weld		Some essential parts of the node are missing.
Node-Blocked		The node is partially or fully hidden from the camera.
Spacer		Spacer improves joist structural behavior.
Seat		Bearing seat improves load distribution in the connection area.
Endnode		Marks the beginning and the end of the joists.

4.1. Classes

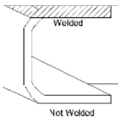
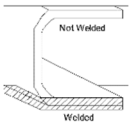
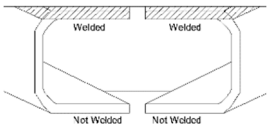
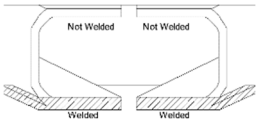
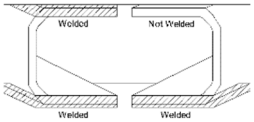
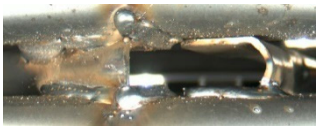
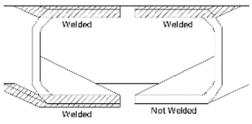
Node-OK: this class refers to fully welded nodes. Each node consists of one to three orthogonal or diagonal welded connections to the main chords. The node welding is considered OK only if all the required steel elements of the node are fully welded. There are three types of fully welded nodes, depending on the number of connected elements and they are shown in (Table 3).

Table 3. Types of fully welded nodes.

Type	Schematic Image	Real Image
One-element nodes		
Two-element nodes		
Three-element nodes		

Missed-Weld: this class refers to a defect in the product, i.e., nodes whose welds are missing a part essential for connecting steel elements. Table 4 demonstrates the most frequent types of missed welds.

Table 4. Frequent types of Missed-Welds.

Type		Schematic Image	Real Image
One-element nodes	Type 1		
	Type 2		
Two-element nodes	Type 3		
	Type 4		
	Type 5		
	Type 6		

Due to the structural design of steel joists, most three-element nodes are on the underside of the joist. It should be noted that factory restrictions in the quality of the

welds prevent creating intentional defects in the joist underside welds. Therefore, partially welded three-element nodes cannot be artificially created.

The third method to identify the dominant types of missed welds is by using the framework itself. By processing the images captured by the cameras, the developed framework can detect possible new missed samples and store them locally for further evaluation. The human annotator verifies the predictions of the framework and includes them in the dataset as the ground truth. This helps increase the accuracy of the framework by completing data samples for the imbalanced types with true-positive samples such as partially welded three-element nodes. Also, it can decrease the detector false-positive rate by including objects previously detected as false-positive in the dataset with accurate annotation.

Moreover, although the precision may decrease, the framework recall can be improved by applying a more significant weight factor to the missed-weld class score in the classifier module. In other words, more missed welds are probably detected, but the rate of false positives will also increase.

Node-Blocked: this refers to nodes being hidden from the camera. Sometimes two joists are stacked to accelerate the welding process. Because the joists vary in length, the camera view of the nodes can become blocked when a bigger joist chord covers the nodes of a smaller one beneath. When this class is detected, the framework notifies the human checker that it is unable to check the welding status due to a blocked view and quality control is done manually.

Spacer: this class refers to small steel members used to prevent buckling in the main chords of steel joists.

Seat: this class refers to the angular steel member typically used where the joist connects to other structural elements, such as girders, to ensure the bearing connection between the joist and the other structural components and to improve load distribution in the connection area.

Endnode: to indicate the position of the missed welds, the framework calculates their relative distance along the defective joist using the detection of the beginning and end of the joists. The detection of the ends of the joists (endnode) can also be used to generate various key performance indicators (KPIs), such as the number of joists produced per day or to calculate the number of spacers in each joist.

Background: this class is added for the detector to detect anything behind the joists as the background.

4.2. Dataset Creation Challenges and Mitigation Scenarios

To create a balanced dataset in our implementation, several plans were used to capture missed welds. The first was to capture the image of nodes during the welding process. As overhead welding is not used to connect steel elements in the factory because of the poor quality it produces, the current welding procedure uses an overhead crane, after the first welding station, to flip the half-welded joists so that the overhead welding position becomes horizontal and the second welding station can complete the weld. In this scenario, the camera can be between two welding stations to capture half-welded nodes.

However, in this method, not all types of missed or partial welds can be captured. Also, the images obtained midpoint have different features than those obtained at the end of the production line (e.g., backgrounds, illumination, camera angle, and distance due to site restrictions) which results in decreasing the effectiveness if the collected samples are included in the dataset.

The second plan classified all possible missed welds and the prominent types were determined intentionally. For that, the images were collected at the end of the production line to ensure homogenous image features.

4.3. Annotation

To create the dataset, the images were annotated using LabelImg (Tzutalin, 2015) [35]. The first 1000 images were labeled manually. An auto-annotating approach was deployed

to automatically generate labels for new images. In this stage, network prediction consisting of class and bounding box information is stored in a text file for each image. Later, the entire raw labels go through quality control by a human agent. The processed batch is added to the evolving dataset, and finally, the network is trained again on the updated dataset to make better predictions for the upcoming batch. Using the above technique, the network assists in the tedious process of labeling, and as the dataset grows, the auto-annotation quality increases.

4.4. Developed Dataset Size

The dataset contains 16,147 annotated images. Node-OK, Missed-Weld, Node-Blocked, Spacer, Seat, and Endnode appear 8873, 8902, 1493, 5973, 2901, and 4477 times, respectively. Its total number of objects is 32,619 and 527 images were included to introduce background elements to the network.

5. Data Collection Station and Hardware Setup

The camera parameters and specifications are shown in Table 5. Compared to Basler acA1920-48gc, Basler a2A1920-51gcPRO can compress frames and downscale frame resolution at the camera.

Table 5. Comparison of Basler acA1920-48gc and Basler a2A1920-51gcPRO.

Property	Basler acA1920-48gc	Basler a2A1920-51gcPRO
Resolution	1920 × 1200	1920 × 1200
Max FPS	50	51
FPS	15	15
Exposure Time (µs)	6800	6800
Gain (dB)	8.21	18
Lens Focal Length (mm)	8.5	6
Sensor Size	2/3″	1/2.3″
Camera-Joist Distance (mm)	300~500	300~500

The virtual machine (VM) runs on an industrial server at the edge of the network to process the live camera feed. The VM consists of 16 virtual processors on Intel® Xeon® Silver 4210R CPU. It has an NVIDIA GRID T4-16Q GPU with 16 GB of dedicated memory, 64 GB of RAM, and SSD storage.

A Lambda Workstation located at ÉTS university is utilized to train the model and analyze the recorded videos. It has an AMD Ryzen Threadripper 3960X CPU, two NVIDIA GEFORCE RTX 3090 as GPUs with 24 GB dedicated memory each, 128 GB of RAM, and SSD storage. Additionally, there are two more servers involved which the framework utilizes to store and visualize data. An SQL server is responsible for storing the prediction data while a server running Ignition®, a web-based industrial automation software, visualizes the results on the user interface.

6. Training and Validation

6.1. Network Hyperparameters

The network hyperparameters are presented in Table 6.

Table 6. Network Hyperparameters.

Network Hyperparameters	
Input size	480 × 480
Optimizer	SGD
Learning rate	0.001
Momentum	0.949
Weight decay	0.0005
Batch	64
Subdivisions	16
Data augmentation	Saturation, exposure, hue, flip horizontally, mosaic

6.2. Training and Validation Metrics

Employing a transfer learning technique, a pre-trained model on the MS COCO dataset [34] was used as the initial weight for our custom object model. K-fold cross-validation and train-validate-test split techniques were used to validate the performance of the detector module.

6.2.1. K-Fold Cross-Validation Results

The dataset was divided into ten folds (k = 10), each fold containing 1615 images. The images were randomly assigned to each fold while maintaining a consistent class distribution and number of node instances in each fold.

Ten models were trained and validated. Each training was done with 16,000 iterations on one NVIDIA GEFORCE RTX 3090, in about 8 h. In each training, the best-performing model based on the AP_{50} was selected. A metric was calculated using the average performance of the ten models. Table 7 summarizes the metric results of AP_{50}, $AP_{75,}$ and mAP as the folds are averaged together.

Table 7. K-fold cross-validation metrics.

Class	AP_{50}	AP_{75}	mAP
Node-OK	0.997	0.980	0.820
Missed-Weld	0.997	0.970	0.794
Node-Blocked	0.986	0.960	0.773
Spacer	0.997	0.977	0.807
Seat	0.998	0.972	0.809
Endnode	0.996	0.843	0.692
Average	**0.995**	**0.950**	**0.783**

The calculated average precision, recall, and F1-score values were 0.92, 0.99, and 0.96, respectively, when the IoU threshold was set to 50%, and the confidence score threshold was considered 0.25. The detector inference time was 0.0083 s (121 FPS).

6.2.2. Train-Validation-Test Results

In the train-validation-test method, the dataset was split into three sets. Sixty percent of the dataset (i.e., 9689 images) was allocated to the training set, while validation and test sets were granted twenty percent (i.e., 3229 images). The training process was done on one NVIDIA GEFORCE RTX 3090, and 15,000 iterations were completed in 7 h and 50 min.

The best-performing model was selected using the validation set during the training process. It was then validated on the test set. Table 8 shows metric results of AP_{50}, $AP_{75,}$ and mAP on the test set.

Table 8. Train-validation-test metrics.

Class	AP_{50}	AP_{75}	mAP
Node-OK	0.994	0.984	0.823
Missed-Weld	0.997	0.971	0.770
Node-Blocked	0.977	0.963	0.733
Spacer	0.995	0.976	0.793
Seat	0.994	0.962	0.802
Endnode	0.990	0.868	0.684
Average	**0.991**	**0.954**	**0.767**

The overall precision, recall, and F1-score were 0.92, 0.99, and 0.95, respectively, when the IoU threshold was set to 50%, and the confidence score threshold was considered 0.25.

6.2.3. Framework Validation

The overall performance of the framework was validated using video samples captured while creating intentionally missed welds, that were unseen by the framework. The framework predictions were compared with the ground truth. The detector confidence score threshold was 0.25, and the non-maximum suppression (NMS) threshold was 0.3. As presented in Table 9, the confusion matrix, precision, and recall were calculated for each class as metrics. The framework inference time was 0.0167 s.

Table 9. Precision and recall.

Class	Precision	Recall
Node-OK	0.994	0.999
Missed-Weld	1	0.992
Node-Blocked	0.981	0.929
Spacer	0.996	0.989
Seat	0.981	1
Endnode	0.972	0.981
Average	**0.9873**	**0.982**

Additionally, the framework ability to track the objects was assessed. If the tracker module could keep track of a node until it reached the last 15 percent of the FoV width, we considered the tracking successful. The tracker module was successful for 6073 nodes and it failed for 34 nodes in 20 h of production. Therefore, the success rate of the tracker module was 0.994.

6.2.4. Performance Assessment in the Production Phase

The performance of the framework was further measured against the actual quality control reports generated in the first stage of visual quality control in the production line. Only the occurrence of missed nodes was reported daily, hence, the comparison was only made on the missed-weld class prediction versus factory reports for three weeks of production.

The total number of detected classes was 103,138. The Node-OK, Missed-Weld, Node-Blocked, Spacer, Seat, and Endnode classes were detected 63,659, 116, 2135, 20,381, 5611, 11,236 times, respectively. Table 10 presents framework defect predictions versus the actual classes.

Table 10. Match Rate of Automated Detection to Reported Missed Welds.

		Actual						
	Class	**Node-OK**	**Missed-Weld**	**Node-Blocked**	**Spacer**	**Seat**	**Endnode**	**Back-Ground**
Prediction	Missed-Weld	9	103	21	1	2	0	0

Regarding the data presented in Table 10, it is important to note that this table is specifically added for assessing the detection accuracy of missed welds. The data acquired from the job site during the first stage of quality assessment in the production phase primarily includes precise figures concerning missed welds, with a scarcity of data for other categories. Therefore, the table is intentionally structured to highlight the accuracy and performance of our model in identifying missed welds. This focus aligns with our research's primary aim and addresses the critical aspect of quality control in welding processes.

The framework counts the defects based on nodes, while missed welds are reported based on the welding path in the shop logs. So, in order to compare the framework output with the shop logs, a conversion was done based on the number of missing welding paths. The framework overall performance is reported in Table 11.

Table 11. Framework overall performance during 17 working days.

Missed-Weld Class	
Predicted number (Node-based)	116
True Positive (Node-based)	103
True Positive (Path-based)	239
False Positive (Node-based)	13
Reported number (Path-based)	88
Precision	0.888
Detection rate	2.71

7. Discussion

The developed framework was trained to detect missed welds and it was validated in the production phase. Although the resulting metrics are promising, there are still limitations that need to be considered. The range of sizes of the joists assembled in the production line is quite broad. In the proposed hardware setup, one camera on either side of the joist was used, but in cases where the joist is very large, the setup cannot provide a complete view of the nodes leading to possible blind spots. Additionally, it is common to stack joists to accelerate the welding process. In such scenarios, one joist may cover part of the other joist node. Currently, this problem is handled by introducing an additional class, i.e., Node-Blocked. In the current production setup, the number of blocked nodes, for which the network cannot predict whether the node is welded or not, is around 3% of the total amount of nodes. To provide a clearer view of the assembly line and nodes, adding cameras with varying angles and FoV to each side of the assembly line could be beneficial. Another issue is that the distance between the camera and the joist is erratic because joists are placed randomly. This can affect the quality of input images as the cameras are manually focused. Employing laser scanners or depth measuring equipment to assess the distance between the camera and the joists, along with cameras that can adjust their focus based on the distance could eliminate this limitation in future. Furthermore, customized joists are assembled in the production line and this may introduce new element types according to the structural design needs. In the conditions of the production phase, the metrics drop compared to the previously applied validation methods. Another reason for the difference between the accuracy of our metric on the test dataset versus the production phase is that balancing classes is considered in the dataset creation while in reality, the input data is severely imbalanced. In the production phase, missed-weld class distribution, compared to its counterpart Node-OK, is 1 to 500. The mentioned fact would put a lot of pressure on the

minority class metrics. Even if the framework performance is satisfactory for the majority class, a tiny percent of edge cases in the majority can generate significant false positives for the minority class, decreasing its precision metrics.

Finally, the process developed in this research identifies missing welds on top and bottom chords of joists. It cannot be used to identify missing welds on the other components of the joist, such as wrapping webs, spacers, and bridging, as they are not within the FoV of the camera. For such components, alternatives methods need to be investigated.

8. Conclusions and Recommendations

We developed a deep-learning-based framework that leverages an average-based classification over a sequence of frames to identify missed welds on a steel joist assembly line. Previous approaches monitor welding quality using different data sources, yet the proposed framework can detect the absence of a necessary weld within the joist connections using live RGB data.

The framework consists of multiple modules to detect, track, and classify the nodes of the joists and it utilizes postprocessing techniques to remove weak predictions from the final classification. Our input image source is the live camera feed, and the framework results are both stored in an SQL database and displayed on a web-based user interface in real-time.

To train the framework, a dataset of 16,147 annotated images was created with six classes of objects. Several sampling techniques were used to adjust the severely unbalanced class data, including intentionally making different configurations of missed welds. An extra class was added to the dataset for a partially or fully blocked view of a node to notify the human quality controller.

Dataset validations show accurate results with mAP of 0.783 and AP_{50} of 0.995 applying the k-fold cross-validation technique. The detector inference time was 0.0083 s and the framework time was 0.0167 s. After framework implementation in the production phase, a comparison was made between framework outputs and the factory logs for the first stage of visual quality control, leading to a precision and recall of 0.884. The detection rate of the framework versus human checkers was 2.71, meaning the developed system outperformed humans in finding weld defects.

In future work, it is recommended that extra cameras be used to address the camera blind spot issue. Also, other sources of data, such as depth and thermal cameras, should be investigated.

Even if we created the main types of missed weld, the dataset still lacks some minor types. This can be corrected by intentionally creating other types of missed welds. Alternatively, techniques such as data augmentation and data synthesis can be used to generate the absent types in the dataset artificially.

In the case study, joists are welded while moving, yet there are other lines in the factory where the joists remain static in the welding process. Future work could focus on redesigning the current hardware and software settings to enable the system to adapt to the specifications of other lines.

The framework inference time could be further decreased if the wrapper were removed from the framework. This would require that the framework architecture only use one programming language.

To further improve the precision of minor classes (missed-weld), it is recommended to utilize a deep-learning-based binary classification network to confirm the existence of missed welds in the objects previously detected as missed-weld by our developed framework. The mentioned reevaluation could decrease false positives in the severely unbalanced data input.

Author Contributions: H.R.: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing—original draft, Writing—review & editing. A.S.: Validation, Visualization, Writing—original draft preparation, Writing—review & editing. D.B.: Funding acquisition, Investigation, Project administration, Resources, Supervision, Valida-

tion, Writing—original draft preparation. A.M.: Conceptualization, Methodology, Funding acquisition, Investigation, Project administration, Resources, Supervision, Writing—original draft preparation, Writing—review & editing. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by Mitacs Acceleration Internship Program grants IT18375 and IT25884.

Data Availability Statement: The datasets presented in this article are not readily available because of data ownership restrictions. Requests to access the datasets should be directed to Groupe CANAM.

Acknowledgments: The authors would like to thank Marc Larochelle and Pierre Bourque from Group CANAM for their invaluable help in realizing the case study. The support provided by Mathieu Fokwa Soh of Group Canam is highly appreciated.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yang, J.; Li, S.; Wang, Z.; Dong, H.; Wang, J.; Tang, S. Using Deep Learning to Detect Defects in Manufacturing: A Comprehensive Survey and Current Challenges. *Materials* **2020**, *13*, 5755. [CrossRef] [PubMed]
2. Jiao, W.; Wang, Q.; Cheng, Y.; Zhang, Y. End-to-End Prediction of Weld Penetration: A Deep Learning and Transfer Learning Based Method. *J. Manuf. Process.* **2021**, *63*, 191–197. [CrossRef]
3. Zhao, R.; Yan, R.; Chen, Z.; Mao, K.; Wang, P.; Gao, R.X. Deep Learning and Its Applications to Machine Health Monitoring. *Mech. Syst. Signal Process.* **2019**, *115*, 213–237. [CrossRef]
4. Tang, X.; Zhong, P.; Zhang, L.; Gu, J.; Liu, Z.; Gao, Y.; Hu, H.; Yang, X. A New Method to Assess Fiber Laser Welding Quality of Stainless Steel 304 Based on Machine Vision and Hidden Markov Models. *IEEE Access* **2020**, *8*, 130633–130646. [CrossRef]
5. Xiong, J.; Zou, S. Active Vision Sensing and Feedback Control of Back Penetration for Thin Sheet Aluminum Alloy in Pulsed MIG Suspension Welding. *J. Process Control* **2019**, *77*, 89–96. [CrossRef]
6. Peng, G.; Gao, Y.; Tian, Z.; Yang, Z.; Xue, B.; Hong, Y.; Du, D. Penetration Control of GTAW Process for Aluminum Alloy Using Vision Sensing. *J. Phys. Conf. Ser.* **2019**, *1303*, 012139. [CrossRef]
7. Soares, L.B.; Weis, A.A.; Rodrigues, R.N.; Botelho, S.S. A Robotic Passive Vision System for Texture Analysis in Weld Beads. In Proceedings of the 2019 IEEE 17th International Conference on Industrial Informatics (INDIN), Helsinki, Finland, 22–25 July 2019; pp. 535–540.
8. Han, Y.; Fan, J.; Yang, X. A Structured Light Vision Sensor for On-Line Weld Bead Measurement and Weld Quality Inspection. *Int. J. Adv. Manuf. Technol* **2020**, *106*, 2065–2078. [CrossRef]
9. Chu, H.-H.; Wang, Z.-Y. A Study on Welding Quality Inspection System for Shell-Tube Heat Exchanger Based on Machine Vision. *Int. J. Precis. Eng. Manuf.* **2017**, *18*, 825–834. [CrossRef]
10. Pratt, W.K. *Digital Image Processing*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2007; ISBN 978-0-470-09743-4.
11. Otsu, N. A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [CrossRef]
12. Chu, H.-H.; Wang, Z.-Y. A Vision-Based System for Post-Welding Quality Measurement and Defect Detection. *Int. J. Adv. Manuf. Technol.* **2016**, *86*, 3007–3014. [CrossRef]
13. Sun, J.; Li, C.; Wu, X.-J.; Palade, V.; Fang, W. An Effective Method of Weld Defect Detection and Classification Based on Machine Vision. *IEEE Trans. Ind. Inf.* **2019**, *15*, 6322–6333. [CrossRef]
14. Hartl, R.; Landgraf, J.; Spahl, J.; Bachmann, A.; Zaeh, M.F. Automated Visual Inspection of Friction Stir Welds: A Deep Learning Approach. In *Multimodal Sensing: Technologies and Applications*; Negahdaripour, S., Stella, E., Ceglarek, D., Möller, C., Eds.; SPIE: Munich, Germany, 2019; p. 8.
15. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
16. Huang, G.; Liu, Z.; van der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. *arXiv* **2016**, arXiv:1608.06993.
17. Cruz, Y.J.; Rivas, M.; Quiza, R.; Beruvides, G.; Haber, R.E. Computer Vision System for Welding Inspection of Liquefied Petroleum Gas Pressure Vessels Based on Combined Digital Image Processing and Deep Learning Techniques. *Sensors* **2020**, *20*, 4505. [CrossRef] [PubMed]
18. Muniategui, A.; García de la Yedra, A.; del Barrio, J.A.; Masenlle, M.; Angulo, X.; Moreno, R. Mass Production Quality Control of Welds Based on Image Processing and Deep Learning in Safety Components Industry. In *Fourteenth International Conference on Quality Control by Artificial Vision*; Cudel, C., Bazeille, S., Verrier, N., Eds.; SPIE: Mulhouse, France, 2019; p. 13.
19. Haffner, O.; Kucera, E.; Bachurikova, M. Proposal of Weld Inspection System with Single-Board Computer and Android Smartphone. In Proceedings of the 2016 Cybernetics & Informatics (K&I), Levoca, Slovakia, 2–5 February 2016; pp. 1–5.
20. Spruck, A.; Seiler, J.; Roll, M.; Dudziak, T.; Eckstein, J.; Kaup, A. Quality Assurance of Weld Seams Using Laser Triangulation Imaging and Deep Neural Networks. In Proceedings of the 2020 IEEE International Workshop on Metrology for Industry 4.0 & IoT, Roma, Italy, 3–5 June 2020; pp. 407–412.

21. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*; IEEE: Piscataway, NJ, USA, 2016; pp. 2818–2826.
22. Xue, B.; Ma, S.; Chu, H.; Liu, K.; Jiao, S.; Meng, Q. Research on Weld Quality Detection Method Based on Machine Vision and Computer Image Processing. *IOP Conf. Ser. Mater. Sci. Eng.* **2019**, *631*, 052031. [CrossRef]
23. Dong, X.; Taylor, C.J.; Cootes, T.F. Automatic Aerospace Weld Inspection Using Unsupervised Local Deep Feature Learning. *Knowl.-Based Syst.* **2021**, *221*, 106892. [CrossRef]
24. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, 5–9 October 2015*; Springer International Publishing: Cham, Switzerland, 2015; pp. 234–241.
25. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
26. Bochinski, E.; Eiselein, V.; Sikora, T. High-Speed Tracking-by-Detection without Using Image Information. In Proceedings of the 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Lecce, Italy, 29 August–1 September 2017; pp. 1–6.
27. MySQL Documentation. Available online: https://Dev.Mysql.Com/Doc. (accessed on 1 November 2021).
28. Ignition Documentation. Available online: https://Docs.Inductiveautomation.Com/Display/DOC81/Welcome (accessed on 1 January 2021).
29. Pypylon (1.7.2). Available online: https://Github.Com/Basler/Pypylon (accessed on 1 November 2021).
30. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*; IEEE: Piscataway, NJ, USA, 2014; pp. 580–587.
31. Girshick, R. Fast R-CNN. *arXiv* **2015**, arXiv:1504.08083.
32. Jiao, L.; Zhang, F.; Liu, F.; Yang, S.; Li, L.; Feng, Z.; Qu, R. A Survey of Deep Learning-Based Object Detection. *IEEE Access* **2019**, *7*, 128837–128868. [CrossRef]
33. Flask (2.0.1). Available online: https://Flask.Palletsprojects.Com/En/2.0.x/ (accessed on 6 January 2020).
34. Lin, T.-Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C.L.; Dollár, P. Microsoft COCO: Common Objects in Context. *arXiv* **2015**, arXiv:1405.0312.
35. Git Code, T.L. 2015. Available online: https://Github.Com/Tzutalin/labelImg (accessed on 6 January 2020).